Globile / ATM-Branch Locator / Services API

A. Vieiro (antonio.vieiro@ciberexperis.es)

Table of Contents

Introduction . . . 1
Definitions . . . 1
The Find Service . . . 4
- The Find Service URL . . . 4
  - The base URL . . . 4
  - The view . . . 4
- The Find Service input data . . . 4
  - The config parameter . . . 4
  - The northEast parameter . . . 4
  - The southWest parameter . . . 5
  - The country parameter . . . 5
  - The customer parameter . . . 5
  - The filterType parameter . . . 5
  - The filterSubtype parameter . . . 5
- Coordinate description . . . 6
- Full URL example . . . 6
- The Find Service output data . . . 6

Appendix I: Sample Code . . . 10

Introduction

This document describes the Globile ATM-Branch Locator backend services, the endpoint, its input and its output.

This is the 1.0 version of this document, released in July, 2018.

Definitions

Some definitions used along this document:

POI

A Point of Interest, this can be either an ATM, a Branch or a third-party object (such a store).

POI Type

The type of a POI. This is a `String` returned in the `objectType.code` JSON property, and has one of the following values:

- `ATM`: The POI is an ATM.
- `BRANCH`: The POI is a Branch. See POI SubType to see what kind of branch this is.
- `CORRESPONSALES`: The POI is not an ATM nor a Branch. See POI SubType to see what kind of POI this is.

POI SubType

Non-ATM POIs can be of different kinds. You must use the `subType` JSON property to check what the specific type of POI is. Available subtypes are:

- When the POI type is `BRANCH` then:
  - `SELECT`, some types of branches that were available in Embassies.
  - `BANCAPRIVADA` a branch for customers in the private-banking segment.
  - `PYME` branch for small companies.
  - `WORKCAFE` Coffee shop and spaces of coworking.
  - `EMPRESAS` branches for big companies.
  - `UNIVERSIDADES` branches for universities.
  - `CLIENTES_POPULAR` branches for customers of Banco Popular.
  - `CLIENTES_PASTOR` branches for customers of Banco Pastor.
  - `CLIENTES_BANEFE` branches for customers of Banco Banefe.
  - `RESIDENTES` branches of Santander in a foreign office.
  - `GRANDES_SUPERFICIES` branches that are located in malls.
  - `AG_COLABORADORES` branches being handled by Santander collaborators.
  - `AG_FINANCIEROS` branches being handled by Santander financial agent.

- For POIs not of type `ATM` nor `BRANCH`:
  - `OXXO`: A POI handled by OXXO https://www.oxxo.com/ .
  - `ELEVEN`: A POI handled by seven eleven https://www.7-eleven.com.mx/ .
  - `CIRCLE_K`: A POI handled by Circle K https://www.circlek.com/ .
  - `TIENDA_EXTRA`: A POI handled by "Tienda Extra" https://es.wikipedia.org/wiki/Tiendas_Extra .
  - `TIENDA_K`: A POI handled by "Tienda K" https://www.circlek.com.mx/ .
  - `TELECOMM`: A POI handled by telecomm.

POI Code

A POI Code is a unique identifier of a POI inside a Country. For instance `B29` uniquely identifies an ATM in Spain.

This value is returned in the `poicode` JSON property.

POI Primary Key

A POI Primary Key is a unique identifier of a POI inside the Global Locator, all around the world. For instance, the `Santander_UK_UK_B29` uniquely identifies ATM `B29` in Santander UK. the POI Primary Key is available as the `code` JSON property.

POI Entity Code

A POI belongs to a Bank or another entity, such as "Santander UK", "Santander España" or "Seven Eleven". This value is available in the `entityCode` JSON property.

Possible entity codes are, currently:

- Santander_Totta: ??
- Santander_ESP: Santander España
- Santander_Private_Banking: ??
- Santander_OpenBank: ??
- Santander_BRA: Santander Brazil
- Santander_UK : Santander UK
- Santander_MEX : Santander México
- Santander_CHI: Santander Chile
- Santander_Río_ARG: ??
- Santander_ALE : Santander Germany
- Santander_US: Santander USA
- Bank_Zachodni_WBK: Santander Poland
- Santander_Peru: Santander Perú
- Santander_PR : ??
- Santander_URU: Santander Uruguay

POI Attributes

A POI may have an array of attributes, i.e., different capabilities the POI provides. These attributes are included in the `attrib` JSON Array, each element in the array is a JSON Object, and each object has a `code` attribute. Possible values are:

- ACCESIBILITY: The POI has accesibility features.
- APPOINTMENT: The POI supports requesting an appointment.
- ATM_INSIDE: The POI has an interior ATM.
- CONTACTLESS: The POI has support for contactless cards.
- EMBOSADORA: The POI has a card issuer.
- MEETING_ROOMS: The POI has meeting rooms.
- MULTICAJERO: The POI has multiple ATMs.
- OPEN_SATURDAY: The POI opens on saturdays.
- PARKING: The POI has parking facilities.
- RETIRO_CON_CODIGO: Withdrawal without a card.
- WIFI: The POI provides wi-fi.

Schedule Format

POIs have opening hours that are returned in the `schedule.workingDay` for working days and `schedule.specialDay` for special days. The exact format is as follows:

```
"schedule":
  {
    "workingDay":{
      "WEDNESDAY":["10:00-17:00","18:00-21:00"],
      "MONDAY":["10:00-17:00","18:00-21:00"],
      "THURSDAY":["10:00-17:00","18:00-21:00"],
      "SUNDAY":[],
      "TUESDAY":["10:00-17:00","18:00-21:00"],
      "FRIDAY":["10:00-17:00","18:00-21:00"],
      "SATURDAY":["09:00-16:00"]},
  },
  "specialDay":
    [
      {"date":"05-15","time":["09:00-12:00"]},{"date":"01-01","time":[]}
    ]
}
```

The object schedule is composed of two parts:

- workingDay: In this object you can see day of the week followed by the opening hours. It can be one or two.
- specialDay: In this object you can see a element formed by date and time. `date` is the day of the year when you have the special schedule and `time`is the special schedule. If time is empty

means that this day the POI is closed.

The Find Service

The "find" service is responsible for returning a number of POIs inside a geographical rectangle.

The Find Service URL

The base URL

The basic URL of the service is https://back-scus.azurewebsites.net/branch-locator/find/

For the **development environment** the URL of the service is https://back.branchlocatorsb.p.azurewebsites.net/branch-locator/find/

This basic URL must be configurable in the Globile ATM-Branch Locator.

The view

A `view` is a String that is used internally in the backend to limit the number of POIs returned, and other settings. This string **must** be configurable in the mobile phones, as it may be used to differentiate between countries.

The `view` parameter is added to the basic URL to conform a new URL. During the development phase we will be using the `devaultView` view, so the URL will be: https://back-scus.azurewebsites.net/branch-locator/find/defaultView

The Find Service input data

The "find" service expects all input data as query parameters.

The config parameter

The `config` query parameter is mandatory, and it must be a JSON object. The JSON object must have a property named `coords` with a value of an array of doubles specifying the center of the search. The radius of search is stored internally in the configuration of the "view".

For example:

```
config={"coords":[40.4625872,-3.8103279]}
```

The northEast parameter

The `northEast` query parameter is optional, it is used when the `config.coords` parameter is missing, and it must be two numbers separated by commas, like so:

```
northEast=40.47899946763356,-3.749092528031042
```

This parameter defines the top-left coordinates of the query rectangle.

The southWest parameter

The `southWest` query parameter is optional, it is used when the `config.coords` parameter is missing, and it must be two numbers separated by commas, like so:

```
southWest=40.47899946763356,-3.749092528031042
```

This parameter defines the bottom-right coordinates of the query rectangle.

The country parameter

The `country` parameter is mandatory, and it specifies the country there the mobile application is to be delivered. So, for instance, for an application delivered in Santander España this parameter will equal the two letter country code "ES".

```
country=ES
```

The customer parameter

The `customer` parameter is optional, and it specifies if the customer has been logged in in the mobile application. If this parameter has not been set, or if it's value is different from `true`, then the user is not considered a Santander customer. If the value of this parameter is `true` then the customer has been logged in into the mobile application and is considered a Santander customer. This affects the way the fees are computed.

```
customer=false
```

The filterType parameter

The `filterType` parameter is optional, and it specifies an array of POI Type values to find.

Example:

```
filterType=ATM
```

The filterSubtype parameter

The `filterSubtype` parameter is optional, and it specifies an array of POI SubType values to find.

Example:

```
filterSubtype=NON_SANTANDER_ATM
```

Coordinate description

The following diagram explains the query parameters

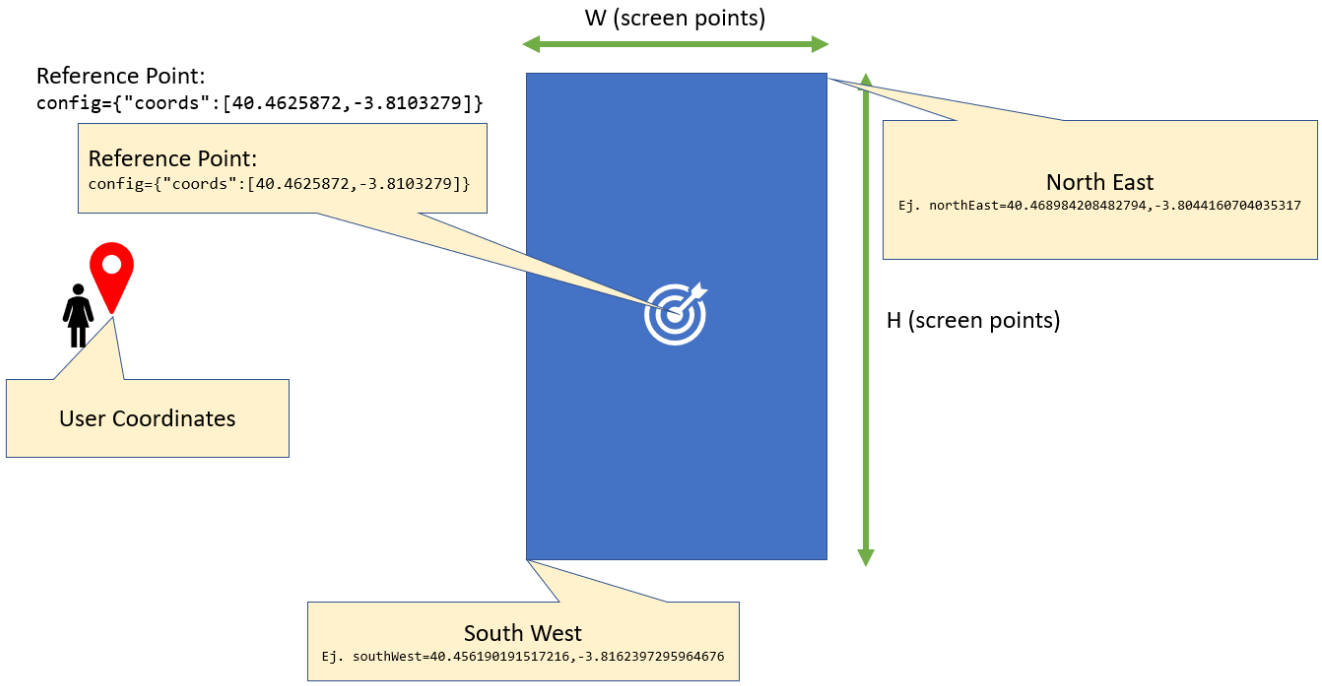

Figure 1. Coordinate description

See This example code for Java code used to compute coordinates.

Full URL example

As an example a final URL could be:

```
https://back-scus.azurewebsites.net/branch-
locator/find/defaultView?config={"coords":[40.46259,-
3.8102959999999997]}&country=ES&customer=false
```

You can click here to run this service

The Find Service output data

The service will return a **JSON ARRAY** of [POI]s. Each [POI] is represented as a **JSON OBJECT**, with many attributes. The Globile ATM-Branch locator **can only use** the documented attributes below

(the rest of attributes are subject to change and may dissappear in future revisions of the service):


```
{
    "action": null,
    "appointment": {
        "branchAppointment": "https://www.santander.co.uk/uk/book-an-appointment", ①
        "waitingTimeSpecialist": null,
        "waitingTimeTeller": null
    },
    "attrib": [ ③
        {
            "code": "WIFI",
            "multi": {
                "default": "YES",
                "en": "YES"
            }
        },
        {
            "code": "ATM_INSIDE",
            "multi": {
                "default": "EXTERNAL ATM",
                "en": "EXTERNAL ATM"
            }
        },
        {
            "code": "APPOINTMENT",
            "multi": {
                "default": "APPOINTMENT",
                "en": "APPOINTMENT"
            }
        }
    ],
    "banner": null,
    "code": "Santander_UK_UK_B29",  ②
    "comercialProducts": [
        "..."
    ],
    "contactData": null,
    "description": null,
    "dialogAttribute": {
        "WIFI": true ③
    },
    "distanceInKm": 0.15921668827597277, ④
    "distanceInMiles": 0.25474670124155646, ④
    "entityCode": "Santander_UK", ⑤
    "events": null,
    "id": "5b3cd746ff3b8bdcffa259de",
    "location": { ⑥
        "address": "11, Ludgate Circus, Ludgate Circus, London, EC4M 7LQ",
        "city": "Ludgate Circus",
```

```
        "coordinates": [
            -0.104049255,
            51.51431829
        ],
        "country": "UK",
        "descriptionPhoto": null,
        "geoCoords": {
            "latitude": 51.51431829,
            "longitude": -0.104049255
        },
        "locationDetails": null,
        "parking": null,
        "type": "Point",
        "urlPhoto": null,
        "zipcode": "EC4M 7LQ"
    },
    "name": "Ludgate Circus", ⑦
    "objectType": { ⑧
        "code": "BRANCH",
        "multi": {
            "default": "BRANCH",
            "en": "BRANCH"
        }
    },
    "people": null,
    "poiStatus": "ACTIVE", ⑨
    "poicode": "B29", ⑩
    "schedule": { ⑪
        "specialDay": [],
        "workingDay": {
            "FRIDAY": [
                "08:30-16:30"
            ],
            "MONDAY": [
                "08:30-16:30"
            ],
            "SATURDAY": [],
            "SUNDAY": [],
            "THURSDAY": [
                "08:30-16:30"
            ],
            "TUESDAY": [
                "08:30-16:30"
            ],
            "WEDNESDAY": [
                "09:30-16:30"
            ]
        }
    },
    "socialData": { ⑫
        "facebookLink": "https://www.facebook.com/santanderuk/",
```

```
        "googleLink": null,
        "instagramLink": null,
        "linkedinLink": "https://www.linkedin.com/company/santander-uk-corporate-&-
commercial",
        "twitterLink": "https://twitter.com/santanderuk",
        "youtubeLink": "https://www.youtube.com/user/UKSantander"
    },
    "specialType": null,
    "spokenlanguages": [ ⑬
        "EN"
    ],
    "status": null,
    "store": "..."
    "subType": { ⑭
        "code": "SELECT",
        "multi": {}
    },
    "urlDetailPage": null ⑮
}
```

① The `appointment.branchAppointment` property, if present, specifies an URL to request an appointment in the POI (usually a branch).

② The POI Primary Key

③ The `attrib` property is an array representing some of the attributes of the POI. Each attriute value in the array is an object where the key `code` defines the attribute. For instance, if `attrib[0].code == 'WIFI'` then the POI supports WIFI. Some possible attribute values include `WIFI` (WI-FI network is available), `ATM_INSIDE` (there is an ATM inside the building), or `APPOINTMENT` (the POI supports requesting an appointment).

④ The `distanceInKm` and `distanceInMiles` return the distance (in Km. and in miles) between the coordinates specified in the `config` input parameter and the POI.

⑤ The POI Entity Code

⑥ The `location` attribute is an object representing the location of the POI.

⑦ The `name` attribute is the name of the POI.

⑧ The `objectType` represents the type of the POI. See POI Type

⑨ The `poiStatus` property indicates if the POI is active, when active the value of this property is `ACTIVE`. This field must not be used. For ATMs with cash there will be another field to be defined. For Branches there will be another field to be defined.

⑩ The `poicode` property indicates the POI Code which is unique in a given country.

⑪ The `schedule` property indicates the ranges of hours in the working days where the POI is open see Schedule Format for details.

⑫ The `socialData` property contains links to social networks.

⑬ The `spokenLanguages` property contains an array of spoken languages in the POI.

⑭ The `subType.code` property contains the subtype of the POI. See POI SubType for more information.

⑮ The `urlDetailPage` proprety may contain an URL specific to the POI.

Appendix I: Sample Code

Sample code used to compute the bounding box given a reference point.

Also available here: GEOUtils.java

```
package com.santander.globile.utils;


/**
 * Some utilitites to compute geographical distances.
 */
public final class GEOUtils {

        /* Semieje mayor del radio de la tierra (WGS84_a), en metros. */
        private static final double WGS84_a = 6378137.0;
        /* Semieje menor del radio de la tierra (WGS84_a), en metros. */
        private static final double WGS84_b = 6356752.3;

        /**
         * Calcula las coordenadas de consulta al servicio de Globile ATM-Branch
Locator, dados un
         * punto de referencia (donde se quiere centrar la vista), la anchura
horizontal en metros que
         * se quiere mostrar, y las dimensiones de la vista (en puntos).
         * @param latitudeDegrees la latitud del punto donde se quiere centrar la
consulta, en grados (ej. -40.4625872).
         * @param longitudeDegrees la longitud del punto donde se quiere centrar la
consulta, en grados (ej. -3.8103279).
         * @param widthInMeters anchura de la consulta, en metros (ej. 500).
         * @param screenWidth la anchura del MapView/fragment, en puntos (ej. 720).
         * @param screenHeight la altura del MapView/fragment, en puntos (ej. 1024).
         * @return Devuelve un array con cuatro doubles: [northEast latitude,
northEast longitude, southWest longitude, southWest latitude].
         * Por ejemplo: [40.468984208482794,-3.8044160704035317,40.456190191517216,-
3.8162397295964676]
         */
        public static double [] getBoundingBox(double latitudeDegrees,
                        double longitudeDegrees,
                        double widthInMeters,
                        int screenWidth,
                        int screenHeight)
        {
                double degrees2Radians = Math.PI / 180.0;
                double latitude = degrees2Radians * latitudeDegrees;
                double longitude = degrees2Radians * longitudeDegrees;

                double radiusOfEarthInLatitude = getRadioDeLaTierraEnLatitud(latitude
```

```
);
                double radiusOfParallelInLatitude = radiusOfEarthInLatitude * Math.
cos(latitude);

                double heightInMeters = widthInMeters * screenHeight / screenWidth;

                double latDelta = heightInMeters / radiusOfEarthInLatitude;
                double longDelta = widthInMeters / radiusOfParallelInLatitude;

                double latitudeMin = (latitude - latDelta) / degrees2Radians;
                double latitudeMax = (latitude + latDelta) / degrees2Radians;

                double longitudeMin = (longitude - longDelta) / degrees2Radians;
                double longitudeMax = (longitude + longDelta) / degrees2Radians;

                return new double [] {
                        // northEastLatitude
                        latitudeMax,
                        // northEastLongitude
                        longitudeMax,
                        // southWestLatitude
                        latitudeMin,
                        // southWestLongitude
                        longitudeMin
                };

        }

        /**
         * Calcula el radio de la tierra en la latitud indicada.
         **/
        private static double getRadioDeLaTierraEnLatitud(double lat)
        {
                double clat = Math.cos(lat);
                double slat = Math.sin(lat);
                // http://en.wikipedia.org/wiki/Earth_radius
                double An = WGS84_a * WGS84_a * clat;
                double Bn = WGS84_b * WGS84_b * slat;
                double Ad = WGS84_a * clat;
                double Bd = WGS84_b * slat;
                return Math.sqrt((An*An + Bn*Bn) / (Ad*Ad + Bd*Bd));
        }

        /**
         * Ejemplo.
         */
        public static void main(String [] args) throws Exception {
                double [] bbox = getBoundingBox(
                                40.4625872, // Latitud del punto de referencia
                                -3.8103279, // Longitud del punto de referencia
                                500,  // Distancia en metros de la anchura (500m.)
```

```
                                720,  // Anchura de la pantalla (720 puntos)
                                1024); // Altura de la pantalla (1024 puntos)
                // Coordenadas "northEast" para la consulta
                System.out.println("northEast=" + bbox[0] + "," + bbox[1]);
                // Coordenadas "southWest" para la consulta
                System.out.println("southWest=" + bbox[2] + "," + bbox[3]);
        }
}
```