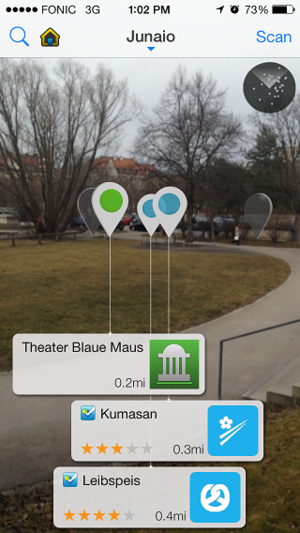
**Location Based AR de Metaio**

Los algoritmos usados en este trabajo final de grado corresponden a la seccion llamada "Location Based AR" del navegador de RA "Metaio". Location Based AR ó Realidad Aumentada basada en la localización es una tecnología de seguimiento no óptica en donde se debe de cargar POIs a travez de un formato de coordenadas que definen las posiciones geográficas en el mundo, llamado "LLA". Éstas coordenadas definen la latitud, longitud y altitud del lugar exacto en que se encuentra posicionado un POI, donde la altitud se mide en metros sobre el nivel del mar, la latitud y la longitud se miden en grados. Además cuenta con un radar en el cual se puede observar donde quedan los puntos de interes que están alrededor del usuario.



Aplicacion basada en la localizacion

En la imagen se puede observar la interfaz proporcionada por el SDK de metaio llamada "IAnnotatedGeometriesGroup", en los SDKs anteriores eran utilizadas las interfaces "IBillboardGroup". Estas interfaces contienen informaciones como el nombre, la distancia, el tipo de un lugar de un punto de interés.

Location Based AR puede ser programado a través de dos maneras: con código Andriod o usando AREL, que es lo que usamos en el presente trabajo. Para trabajar con AREL debe de incluirse la carpeta Assets en el proyecto Android de la aplicación a crear, dicha carpeta se encuentra en el sitio de metaio en el proyecto "Example".

Lo primero que hay que hacer para usar la RA basada en la localización es establecer la configuración de seguimiento por parte del móvil a través del GPS. En AREL se puede realizar esto con el siguiente código: arel.Scene.setTrackingConfiguration(arel.Tracking.GPS); y luego ya se puede crear la escena con varios POIs y establecer su ubicación. Se debe de crear una instancia de la clase LLACoordinate para poder establecer la ubicación del POI y luego crear el respectivo POI. Esto se puede hacer con las siguientes funciones proporcionadas por AREL:

...

**var** ubicacionUNI = **new** arel.LLA(51.50661, -0.130463, 0);

createPOI("2", "UNI", ubicacionUNI);

...

*//aqui se crea un nuevo poi*

**function** createPOI(id, nombre, ubicacion)

{

**var** newPOI = **new** arel.Object.POI();

newPOI.setID(id);

newPOI.setTitle(nombre);

newPOI.setLocation(location);

newPOI.setThumbnail("");

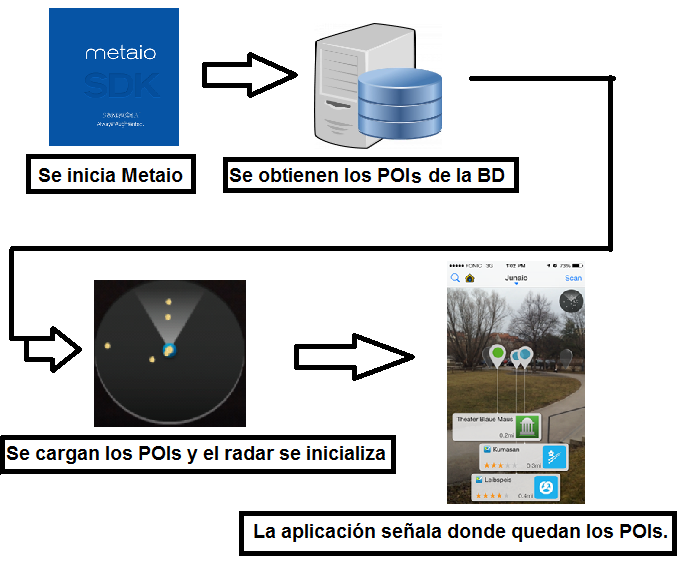
newPOI.setIcon("");

newPOI.setVisibility(**true**,**false**,**true**);

arel.Scene.addObject(newPOI);

}

En Arel existe una función llamada "setLLAObjectRenderingLimits" que sirve para establecer los límites desde donde se pueden ver los puntos de interés en la aplicación. También se puede agregar miniatura con la función " setThumbnail " y un icono con la función " setIcon ", también se puede definir una ventana emergente. Además el radar que contiene los puntos con sus coordenadas se añade automáticamente para la escena.



Al iniciar el recorrido, la aplicación carga la librería de metaio, éste inicia la cámara y crea los puntos de interés verificando la posición del usuario con el fin de deinir si le deben de aparecer los POIs de acuerdo a los límites establecidos en el código con uno de los métodos de metaio mencionados anteriormente. Para crear los POIs la aplicación utiliza los métodos mencionados anteriormente y se conecta a la base de datos en el servidor a través de métodos AJAX que se conectan a archivos PHP en el servidor para realizar la consulta a la base de datos obteniendo así los datos para crear cada POI. Cuando todos los POIs son cargados entonces el radar aparece en la pantalla y dentro de él los puntos de interés. Desde ese momento, el usuario puede realizar el recorrido donde la aplicación le indicará hacia donde quedan los POIs a través de un icono en forma de gota que señalará en la pantalla del usuario donde queda cada POI. Este icono contendrá el nombre del POI, una imagen en miniatura del POI y la distancia que se encuentra el usuario al POI, al acercarse al POI la se hará más grande.

**Funciones en AREL**

Existen algunas funciones de AREL que utilizamos en este trabajo final de grado que explicamos a continuación:

* arel.sceneReady(startCallback, activateDebugging, useInBrowser)

Este función se produce una vez que todo el contenido de la página principal está cargada. Aquí se crean los POIs y se establece los objetos para la escena.

Parámetros:

- {función} startCallback: La función de devolución de llamada que se llamará una vez que se cargan todos los objetos estáticos y la escena esté lista.

- {Booleano} activateDebugging: Se establece en true, si la consola de depuración se debe utilizar.

- {Booleano} useInBrowser: Se establece en true si el comando se utiliza en el primer navegador.

* arel.LLA(\_lat, \_lng, \_alt, \_acc)

Establece la nueva geolocalización. El autor es Metaio GmbH.

Parámetros:

- {Number} \_lat: La latitud para esta geolocalización. Ejemplo: 48.14243

- {Number} \_lng: La longitud para esta geolocalización.

- {Number} \_alt: La altitud para la geolocalización en metros.

- {Number} \_acc: El valor de la exactitud del sensor de posición actual (GPS) en metros.

* arel.Scene.addObject(\_object)

Esta función sirve para agregar un POI o un objeto a la escena.

Parámetro:

- {arel.Objet} \_object: la variable de tipo objeto para ser agregado.

* arel.Scene.getObject(\_objectID)

Esta función retorna un arel.Object desde la escena dada.

Parámetro:

- {String} \_objectID: El id del objeto que se desea obtener.

* arel.Scene.setLLAObjectRenderingLimits(\_near, \_far)

Esta función establece los limites desde donde y hasta donde el usuario puede percibir los POIs con coordenadas LLA. Para desactivar esta función se puede ajustar los limites con 0.

Parámetros:

- {Number} \_near: El limite más próximo en metros.

- {Number} \_far: El limite lejano en metros.

* Object.setLocation(\_location)

Esta funcion setea la informacion de la ubicacion para objetos basados en la localizacion solamente.

Parametros:

- {[arel.LLA](https://dev.metaio.com/fileadmin/documentationareljs/symbols/arel.lla.html)} \_location: una variable de tipo LLA donde se establece sus coordenadas como latitud, longitud y altitud.

* Object.setVisibility(\_liveview, \_maplist, \_radar)

Establece la visibilidad de este objeto para el mapeo 3D, la lista de POIs y el radar. Los dos últimos parámetros son relevantes para canales basados en la localización como el caso del presente trabajo.

Parámetros:

- {boolean} \_liveview: Se establece en true si este objeto se debe mostrar en la escena.

- {boolean} \_maplist: Se establece en true si este objeto se debe mostrar en el mapa y en la lista.

- {boolean} \_radar: Se establece en true si este objeto debe ser mostrado en el radar.

* Object.setLLALimitsEnabled(\_enable)

Establece si el objeto se moverá dentro del rango de distancia de los límites establecidos.

Parámetros:

- {bool} \_enable: true para activar la iluminación.

* POI.setMinDistance(\_mindistance)

Ajusta la distancia desde el cual el objeto se puede mostrar, sólo para canales basados en la localización.

Parámetros:

- {int} \_mindistance: Distancia mínima en metros.

* POI.setMaxDistance(\_maxdistance)

Ajusta la distancia máxima desde el cual el objeto se puede mostrar, sólo para canales basados en la localización.

Parámetros:

- {int} \_maxdistance: Distancia máxima en metros.

* POI.setID(\_id)

Establece el id para el objeto, su identificacion es alfanumerico y único para cada objeto que luego no se puede cambiar. El objeto debe de tener su ID antes de añadirlo a la escena.

Parámetro:

- {string} \_id: identificador único alfanumerico para el objeto.

* POI.setTitle(\_title)

Esta función establece el titulo para el objeto

Parámetro:

- {String} \_title: titulo para el objeto.

* POI.setLocation(\_location)

Establece la information de la ubicacion del objeto para canales basados en la localización.

Parámetro:

- {arel.LLA} \_location: una variable de tipo LLA donde se establece sus coordenadas como latitud, longitud y altitud.

* POI.setThumbnail(\_thumbnailPath)

Establece la imagen en miniatura que se muestra en cada POI para objetos basados en la localización.

Parámetro:

- {string} \_ thumbnailPath: path, ruta donde se encuentra la imagen.

* POI.setIcon(\_iconPath)

Establece el icono del objeto que se muestra en cada POI para objetos basados en la localización.

Parámetro:

- {string} \_ iconPath: path, ruta donde se encuentra la imagen.

* POI.setPopup(\_popup)

Establece la informacion que se mostrará en el emergente del POI. El Popup aparecerá cuando el usuario hace click en el objeto.

Parámetros:

- {arel.Popup} \_popup: El objeto de tipo popup que contendrá la información del POI.

* arel.Events.Object.ONTOUCHSTARTED

En este evento se puede una función que ocurrirá cuando el objeto es presionado. En el este trabajo final de grado usamos este método para que al presionar un POI se muestre una ventana emergente con la descripción del POI pulsado.

* arel.Media.speak(\_text)

Función a través del cual la aplicación lee lo que se le pasa en el parámetro en nuestro caso la descripción del POI usando el idioma actual del dispositivo móvil.

Parámetro:

- {string} \_text: texto para leer.

Fuente:

https://dev.metaio.com/arel/javascript-reference/index.html

https://dev.metaio.com/storage-folder/tutorials/location-based-ar/