# *Y ahora…. ¿Cómo vuelvo?*

Javier Ávila Nieto

"Trabajo Final de Carrera"

07 – Enero – 2013

# Contenido

# 1. Descripción del proyecto

No es necesario decir cómo nos ha cambiado internet en la forma en que las personas planean las vacaciones o acceden a las mejores ofertas turísticas. Desde la preparación, las reservas de billetes, hoteles o restaurantes, hasta la localización en un mapa en el teléfono del lugar en el que pasaremos nuestras próximas vacaciones. La tecnología web nos suministra herramientas que nos ayudan a tener a la mano mucha información que necesitaremos en el viaje. Atrás quedan las guías de papel, cuando se pueden descargar guía virtuales con geolocalización de museos, bares, restaurantes y demás lugares de interés.

Las nuevas herramientas nos facilitan mucho las cosas a la hora de conocer lugares nuevos, por eso esta aplicación quiere aportar un complemento más para aquellos que quieren disfrutar del viaje sin pérdidas, ni de tiempo ni de localización.

Para poder realizar este trabajo, el primer paso necesario ha sido la investigación del funcionamiento del sistema operativo Android y de todas las posibilidades de las características que ofrece a partir de las APIs disponibles. Se ha elegido desarrollar la aplicación sobre este sistema operativo debido principalmente al auge que está viviendo, ya que es el que más se utiliza actualmente y en el que existe una comunidad de desarrolladores de aplicaciones mucho mayor.

El origen de este proyecto surge de la necesidad de viajeros que hacen viajes, especialmente de corta duración, llegan a su alojamiento temporal y, o bien porque no tienen mucho tiempo o porque tienen prisa por conocer la ciudad, en la mayoría de los casos, a pesar de saber dónde está el hotel y hacia donde se van a dirigir, olvidan anotar previamente cómo se llama el hotel o la dirección donde se ubica. Con esta aplicación ya no necesitaran recordar dónde está su hotel, su calle, ni su ubicación exacta en el mapa para volver. La aplicación almacenará su posición exacta al salir de cualquier lugar, de modo que cuando quieran regresar al punto de origen la aplicación les mostrará la ruta de vuelta, tanto si es a pie como si es en coche.

## 2. Justificación del proyecto

El objetivo de este Trabajo Final de Carrera es la realización del proyecto de un sistema que permita a los usuarios de dispositivos móviles, con sistema operativo Android, guardar lugares en la memoria del teléfono para encontrar el camino de vuelta sin ningún problema, esté donde esté.

Esta aplicación debe ser capaz de localizar y almacenar la posición de un usuario de teléfono móvil gracias a las coordenadas obtenidas por el GPS del mismo. Con ello, la aplicación será capaz de trazar la ruta de vuelta a una posición.

Actualmente, existen aplicaciones que permiten a un usuario ver mapas, ver dónde se encuentra e incluso guardar la posición actual. Esta aplicación no es simplemente un localizador GPS de la posición actual o una guía para llegar a un punto que fijemos, la aplicación va servirnos para localizar la ruta de vuelta y no perdernos.

En esta aplicación se puede observar cómo se aprovechan las capacidades que nos dan las API de Google Maps. Gracias a éstas, el sistema será capaz de localizar al usuario en un mapa y mostrárselo en el terminal móvil. Del mismo modo que Google Maps nos ofrece la posibilidad de elegir un trayecto de “x” a “y”, con esta aplicación podremos volver al punto “x”; el origen que nosotros queramos, aunque no hayamos programado previamente en el dispositivo móvil ningún trayecto de “x” a “y”, es decir, no es necesario marcar el itinerario inicial, la aplicación lo almacena y después haríamos el recorrido inverso, de “y” a “x”. Esto nos permite no tener que saber obligatoriamente la posición exacta desde la que partiremos a la vuelta.

De manera que esta aplicación facilita una información valiosa para el viajero que salió a caminar o conocer la ciudad sin un destino fijo y necesita valorar las posibles alternativas de vuelta a su alojamiento, que previamente a almacenado en la memoria de su móvil, a través de la aplicación.

# 3. Usuarios y contextos de uso

## 3.1. Perfil del usuario

La aplicación está enfocada para que pueda ser utilizada por cualquier persona, ya sea una persona con bajos conocimientos o poca experiencia con las aplicaciones de dispositivos móviles hasta personas con mayor destreza en el uso de los dispositivos.

Para alcanzar el mayor número de usuarios posibles el fundamental el diseño de la aplicación. Para ello debe ser de fácil manejo, creando menús muy intuitivos y donde la información esté bien organizada siguiendo los criterios de usabilidad.

## 3.2. Usuarios encuestados

Se realiza una encuesta con seis personas seleccionadas al azar, para determinar si existe relación en la edad y nivel estudios con el uso de las tecnologías, los dispositivos móviles y el acceso a internet a través de los mismos. Después se hará una entrevista a esos mismos encuestados, para conocer mejor las capacidades que tienen para manejar los nuevos dispositivos móviles y la facilidad o dificultad en el uso de un prototipo similar al que estamos desarrollando en este proyecto.

| Nº | Sexo | Edad | Nivel cultural | Uso internet | Manejo móviles | Móvil con acceso internet |
|---|---|---|---|---|---|---|
| 1 | Mujer | 60 | Medio | Básico | Si | No |
| 2 | Hombre | 61 | Medio | Medio | Si | Si |
| 3 | Mujer | 40 | Alto | Básico | Si | No |
| 4 | Mujer | 36 | Alto | Alto | Si | Si |
| 5 | Hombre | 32 | Alto | Alto | Si | Si |
| 6 | Hombre | 15 | Medio | Alto | Si | Si |

## 3.3. Resultados

Tras el análisis de los datos obtenidos con la encuesta se puede determinar que no existe una correlación entre edad, nivel estudios y uso de las nuevas tecnologías; hay personas jóvenes, con alto nivel cultural o de estudios, hacen un uso básico de las nuevas tecnologías, y personas muy jóvenes con nivel cultural bajo que no tiene dificultad para el manejo de nuevas tecnologías, en el soporte que sea.

Para la entrevista utilizamos un prototipo similar al que estamos desarrollando en este proyecto, se entrega a los encuestados una serie de tareas para que realicen con la aplicación “¿Cómo vuelvo?”.

Observamos cómo para las personas más familiarizadas con los móviles con acceso a internet es más fácil el uso de este prototipo. Aquellos que utilizan aplicaciones de geolocalización en su móvil no tienen grandes dificultades para realizar acciones con el prototipo del proyecto. Aunque en la encuesta declaran haber tenido alguna dificultad para encontrar la acción de exportar datos.

Para aquellos que están habituados al manejo de internet, no con el móvil, no es muy complicado llegar a realizar una tarea sencilla con el prototipo, guardar un destino, aunque es más difícil para ellos encontrar la forma de volver al punto guardado.
En el caso de los usuarios que tienen menos habilidades en el manejo de internet les resulta muy complicado realizar alguna de las acciones que les entregamos al principio del a entrevista con este prototipo de la aplicación.

En cuanto a las propuestas de los entrevistados para mejorar el acceso y uso de la aplicación, recogemos las siguientes:

- Diseño más sencillo, una sola pantalla que muestre todas las opciones.
- Prefieren que no haya menús emergentes.
- Navegación intuitiva.
- Acciones básicas en una sola pantalla.
- Se añade una nueva funcionalidad que nos comentan que pueden ser de gran utilidad, un buscador de nuevos lugares…

### 3.4. Contexto de uso

La aplicación está enfocada para viajeros que visitan ciudades nuevas. Su principal diferencia frente a otras aplicaciones del mercado es que muestra el camino de vuelta a un punto visitado anteriormente, para lo cual, el diseño de la aplicación debe satisfacer las siguientes funcionalidades:

- Debe ser capaz de localizar la posición del usuario del dispositivo móvil haciendo uso del GPS del mismo.
- Localizar esa posición y mostrarlo en el mapa utilizando para ello las API's de Google Maps.
- Posibilidad de almacenar la posiciones en una base de datos interna haciendo uso de SQLLite. Mostrar la ruta de vuelta desde la posición actual a cualquiera de los puntos almacenados en la base de datos.

### 3.5. Análisis de tareas

- Mapa de posición
  El usuario, gracias al sistema de geoposicionamiento del dispositivo móvil, podrá ver en cualquier momento su posición en un mapa. Este sistema ayudará al usuario a situarse en caso de pérdida o le ayudará a guiarse al poder ver las calles adyacentes.

- Almacén de puntos de interés
  La aplicación dispondrá de una base de datos (que se almacenará de forma local en el dispositivo) que le permitirá almacenar puntos de interés. Estos puntos de interés podrán almacenarse buscando una dirección o guardando posiciones donde ha estado previamente el usuario.

- Buscador de lugares
  La aplicación dará al usuario, mediante un buscador con una caja de texto, la posibilidad de buscar lugares de interés o direcciones para poder localizarlas en el mapa. Adicionalmente se le ofrecerá también la posibilidad de guardar el lugar en la base de datos para poder recuperarlo más tarde.

- Exportar datos
  Los datos guardados por el usuario podrán ser exportados fácilmente a otras aplicaciones como Google Earth donde podrá ver las posiciones de los lugares almacenados en el dispositivo. Los datos pueden ser exportados en formato gpx, kmz y kml, lo que permitirá al usuario importarlos con cualquier otra aplicación que acepte dichas extensiones de archivo.

# 4. Diseño Técnico

En este apartado se va a tratar de dar respuesta a cómo debe funcionar la aplicación. Antes de nada es necesario explicar que una interfaz gráfica y la interactuación entre el usuario y el sistema se realizará por medio de Activities, mediante extensión de la clase Activity. Se puede decir que una Activity es una pantalla de la aplicación en la que el usuario puede interactuar con el sistema.

El núcleo principal del sistema está la clase MostrarPosicionActual.class, que realmente extiende una Activity especial: MapActivity. Tiene asignadas dos tareas básicas: construcción del mapa en la pantalla y la creación un localizador de posición del dispositivo en el mapa. En esta misma pantalla se muestra una brújula y un botón con el que el usuario podrá salvar la posición en la que se encuentra en la base de datos, de modo que luego pueda encontrar el camino de vuelta a ese punto sin la más mínima complicación. Además, está previsto para futuras versiones, añadir un botón con el que el usuario pueda cambiar el modo de vista en el mapa.
La principal función de esta clase es asignar el listener que se encargará de actualizar periódicamente la posición del usuario y realizar sobre el mapa las operaciones necesarias. Una vez captada la posición del usuario, las coordenadas geográficas serán convertidas en una dirección real, que es la encargada de hacer la consulta adecuada a la API Geocode. La dirección obtenida será mostrada por pantalla gracias a la clase MyOverlay.class, que extiende a ItemizedOverlay. Esta clase situará en la posición correcta del mapa un icono que representará a usuario.

Actualmente la aplicación sólo cuenta con un idioma, pero está preparada para que en futuras versiones se puedan añadir más idiomas de manera sencilla, ya que todas las cadenas de texto se han guardado en el archivo strings.xml de la carpeta values.

## 4.1. Escenarios de uso

- Escenario para mostrar la posición del usuario

Rocío es una joven viajera que está realizando un viaje por Asia. Acaba de llegar a su hotel en Beijing tras 9 horas de vuelo, pero como ha llegado al medio día decide salir a conocer la ciudad. Tiene miedo a perderse, y no encontrar el camino de vuelta, ya que todas las casas y calles le parecen iguales. Para evitar no encontrar el camino de vuelta decide abrir la aplicación “¿Cómo vuelvo?” de su teléfono móvil y guardar la posición donde se encuentra actualmente (el hotel) para saber dónde está situado exactamente.

- Escenario para buscar lugares en el mapa

Una vez llegado al hotel, la joven viajera Rocío, se encuentra con una pareja de viajeros que lleva ya 7 días paseando por Beijing. Durante la charla le recomiendan a Rocío ciertos lugares que no puede perderse bajo ningún concepto, ya que son sitios con un encanto especial que no aparecen en la mayoría de las guías. Mientras apunta los nombres en su cuaderno de viaje, abre la aplicación y utiliza el buscador para ver en qué zona está cada uno de ellos y le va saliendo el mapa para poder situarlos.

- Escenario para almacenar lugares

Como no va a acordarse de las indicaciones que le dan para llegar a esos lugares, cada vez que buscan uno de los lugares que le comentan, lo busca en el mapa y acto seguido lo almacena en la base de datos del dispositivo.

- Escenario para encontrar el camino de ida / vuelta

Como era previsible, después de todo el día recorriendo la ciudad, andando de un lado para el otro, Rocío no encuentra el camino de vuelta. Tiene un mapa que de poco le vale porque no aparecen los nombres de las calles y preguntar a la gente se convierte en una dura tarea, ya que no es sencillo encontrar a alguien que hable inglés. Así que se decide a abrir la aplicación "¿Cómo vuelvo?". Rocío puede acceder al listado de lugares almacenados, entre los que está el hotel (cuya posición había almacenado por la mañana) y lo selecciona. Automáticamente el móvil comprueba la posición actual de Rocío y muestra la ruta de vuelta al hotel.

- Escenario para exportar los lugares almacenados

Al final del viaje, de vuelta en casa, a Rocío le gustaría poder ver en el ordenador en el Google Earth todos los lugares que tiene guardados en el móvil para poder verlos mejor en la pantalla del ordenador. Así que, abre la aplicación y selecciona la opción de exportar datos. El programa genera un fichero kmz que le permitirá importar todos los datos guardados en su ordenador cómodamente.

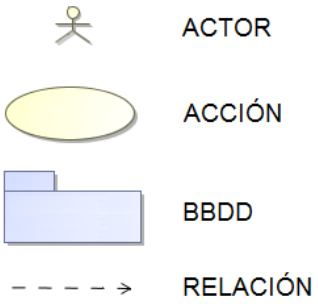
## 4.2. Casos de uso

"Un caso de uso es una descripción de los pasos o las actividades que deberán realizarse para llevar a cabo algún proceso. Los personajes o entidades que participarán en un caso de uso se denominan actores. En el contexto de ingeniería del software, un caso de uso es una secuencia de interacciones que se desarrollarán entre un sistema y sus actores en respuesta a un evento que inicia un actor principal sobre el propio sistema. Los diagramas de casos de uso sirven para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios y/u otros sistemas." [1]

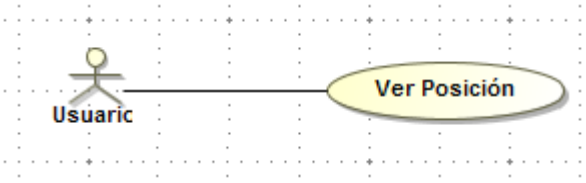
Para la realización de los distintos diagramas de casos de uso realizados en este documento se ha utilizado el software “MagicDraw UML”. Este software está especialmente diseñado para el desarrollo de todo tipo de diagramas UML”.

Cada diagrama de los que se muestran a continuación, muestran las posibles interacciones del usuario con el dispositivo móvil.

**Leyenda**

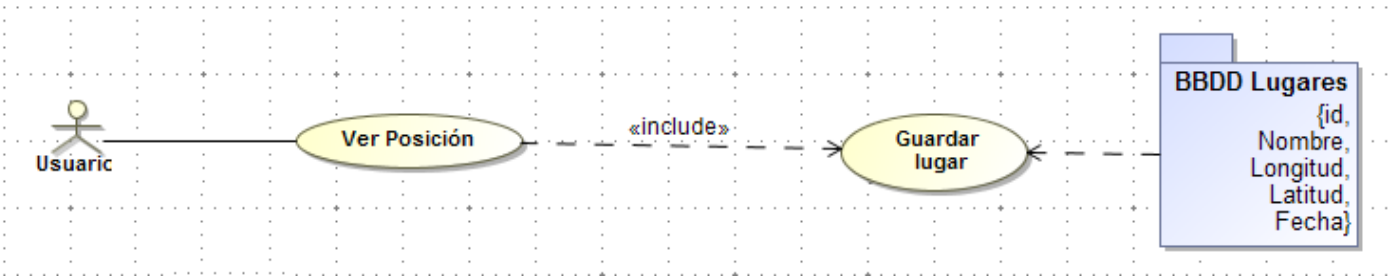



- Diagrama caso de uso: “Mostrar la posición del usuario”



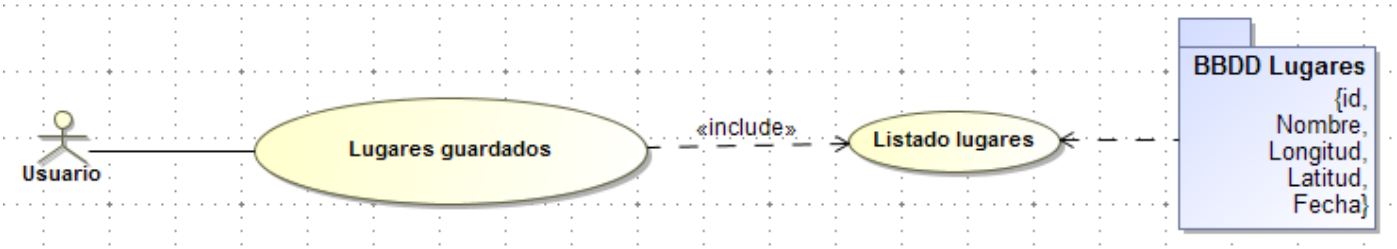

| | |
|---|---|
| **Nombre** | Mostrar la posición del usuario |
| **Actor** | Usuario |
| **Flujo** | 1. El actor abre la aplicación en el dispositivo móvil.<br>2. El actor selección en el menú inicial la opción “Ver posición actual”<br>3. El sistema muestra por pantalla al actor un mapa de la zona donde se encuentra con una marca que le indica su posición exacta. |

- Diagrama caso de uso: “Guardar la posición del usuario en la BBDD”



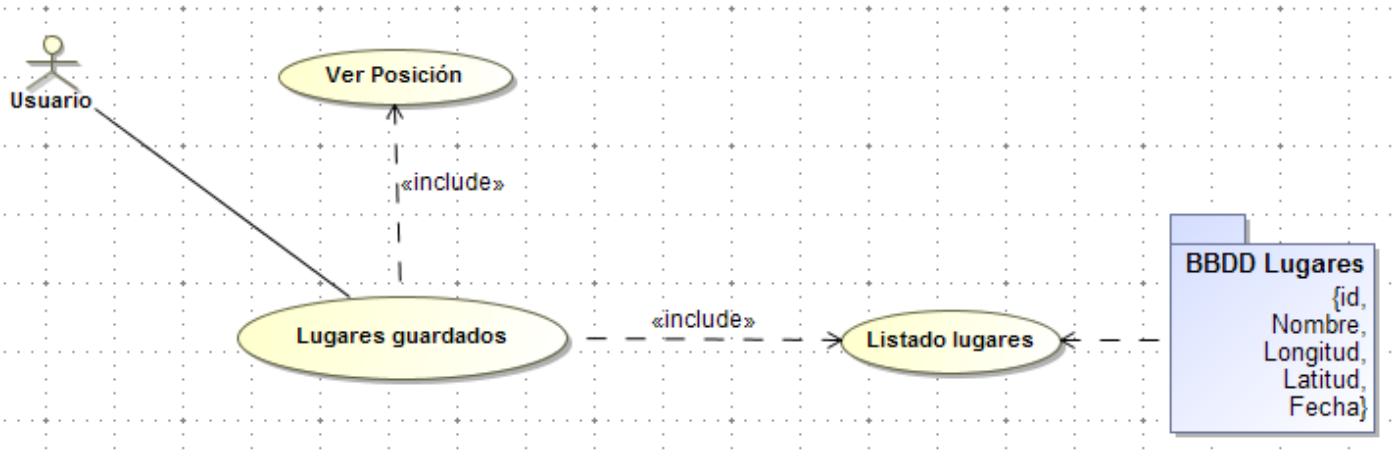

| **Nombre** | Guardar la posición del usuario en la BBDD |
|---|---|
| **Actor** | Usuario |
| **Flujo** | 1. El actor abre la aplicación en el dispositivo móvil.<br>2. El actor selección en el menú inicial la opción “Ver posición actual”<br>3. El sistema muestra por pantalla al actor un mapa de la zona donde se encuentra con una marca que le indica su posición exacta.<br>4. El actor pulsa el botón “Guardar posición”<br>5. El sistema muestra una ventana al actor para que ingrese el nombre del lugar (para poder encontrarlo más tarde).<br>6. El usuario añade un nombre y pulsa en el Botón “Guardar”<br>7. El sistema añade una entrada en la BBDD lugares, con el nombre puesto por el actor, la longitud y la latitud de la posición seleccionadas. |

- Diagrama caso de uso: ”Listar los lugares almacenados en la BBDD”



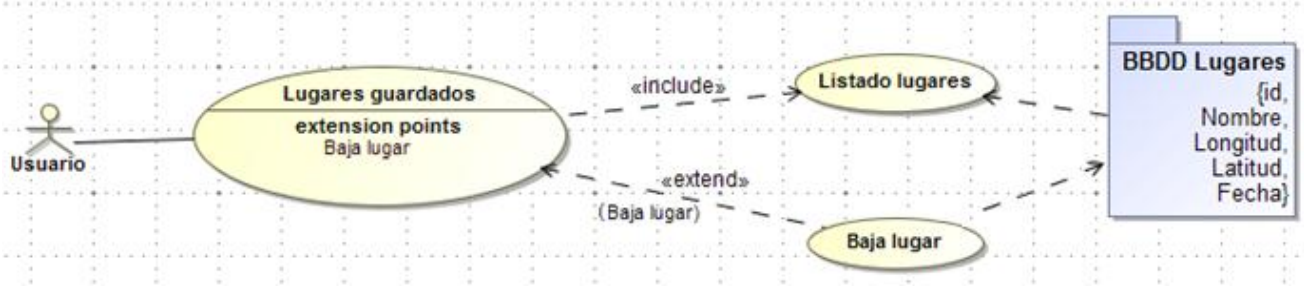

| **Nombre** | Listar los lugares almacenados en la BBDD |
|---|---|
| **Actor** | Usuario |
| **Flujo** | 1. El actor abre la aplicación en el dispositivo móvil.<br>2. El actor selección en el menú inicial la opción “Ver lugares guardados”<br>3. El sistema muestra por pantalla al actor un listado con todos los lugares guardados en la BBDD lugares ordenados por fecha de inclusión en el sistema. |

- Diagrama caso de uso: ”Mostrar en el mapa la posición de un elemento de la lista de lugares almacenados en la BBDD”



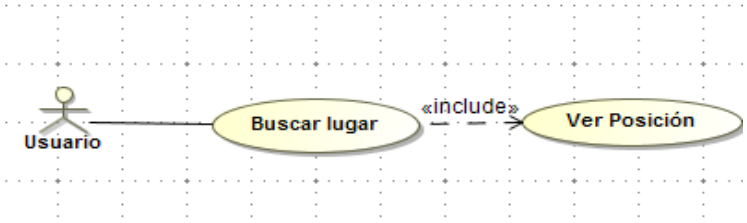

| **Nombre** | Mostrar en el mapa la posición de un elemento de la lista de lugares almacenados en la BBDD |
|---|---|
| **Actor** | Usuario |
| **Flujo** | 1. El actor abre la aplicación en el dispositivo móvil.<br>2. El actor selección en el menú inicial la opción “Ver lugares guardados”<br>3. El sistema muestra por pantalla al actor un listado con todos los lugares guardados en la BBDD lugares ordenados por fecha de inclusión en el sistema.<br>4. El actor selecciona uno de los lugares del listado.<br>5. El sistema muestra al usuario una ventana de opciones.<br>6. El usuario selecciona la opción “Mostrar posición en el mapa”<br>7. El sistema muestra por pantalla al actor un mapa de la zona donde se encuentra con una marca que le indica su posición exacta. |

- Diagrama caso de uso: ”Eliminar un elemento de la lista de lugares almacenados en la BBDD”



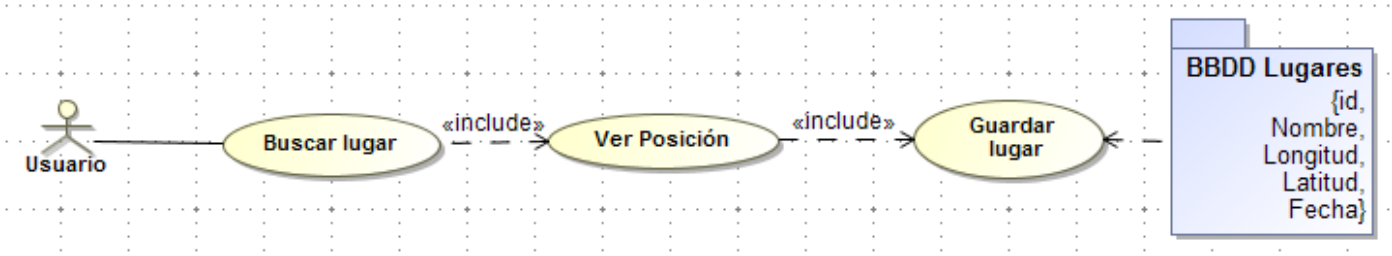

| **Nombre** | Eliminar un elemento de la lista de lugares almacenados en la BBDD |
|---|---|
| **Actor** | Usuario |
| **Flujo** | 1. El actor abre la aplicación en el dispositivo móvil.<br>2. El actor selección en el menú inicial la opción “Ver lugares guardados”<br>3. El sistema muestra por pantalla al actor un listado con todos los lugares guardados en la BBDD lugares ordenados por fecha de inclusión en el sistema.<br>4. El actor selecciona uno de los lugares del listado.<br>5. El sistema muestra al usuario una ventana de opciones.<br>6. El usuario selecciona la opción “Eliminar lugar”<br>7. El sistema elimina de la BBDD lugares el registro seleccionado por el usuario.<br>8. El sistema muestra por pantalla al actor un listado con todos los lugares guardados en la BBDD lugares ordenados por fecha de inclusión en el sistema. |

- Diagrama caso de uso: ”Localizar un lugar nuevo en el mapa”



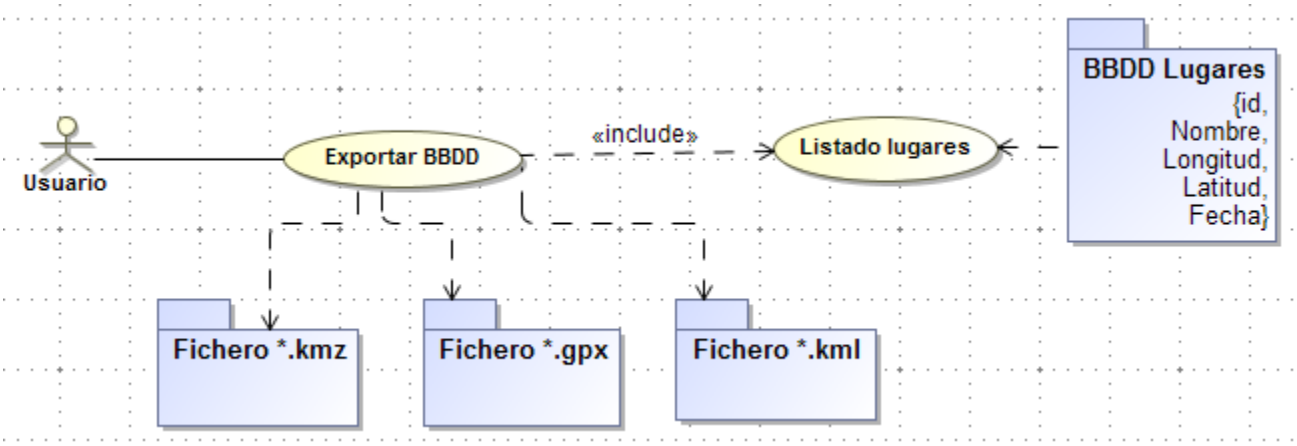

| **Nombre** | Localizar un lugar nuevo en el mapa |
|---|---|
| **Actor** | Usuario |
| **Flujo** | 1. El actor abre la aplicación en el dispositivo móvil.<br>2. El actor selección en el menú inicial la opción “Buscar lugar”<br>3. El sistema muestra por pantalla al actor una caja de texto para que introduzca una dirección o el nombre del lugar que quiere buscar.<br>4. El actor escribe en la caja de texto el nombre del lugar que quiere encontrar y pulsa en el botón “Buscar”.<br>5. El sistema se conecta a internet y busca la posición de la dirección escrita por el usuario.<br>6. Una vez localizada la posición, el sistema muestra por pantalla al actor un mapa de la zona donde se encuentra con una marca que le indica su posición exacta. |

- Diagrama caso de uso: ”Guardar en la BBDD lugares la posición de un lugar buscado por el usuario”



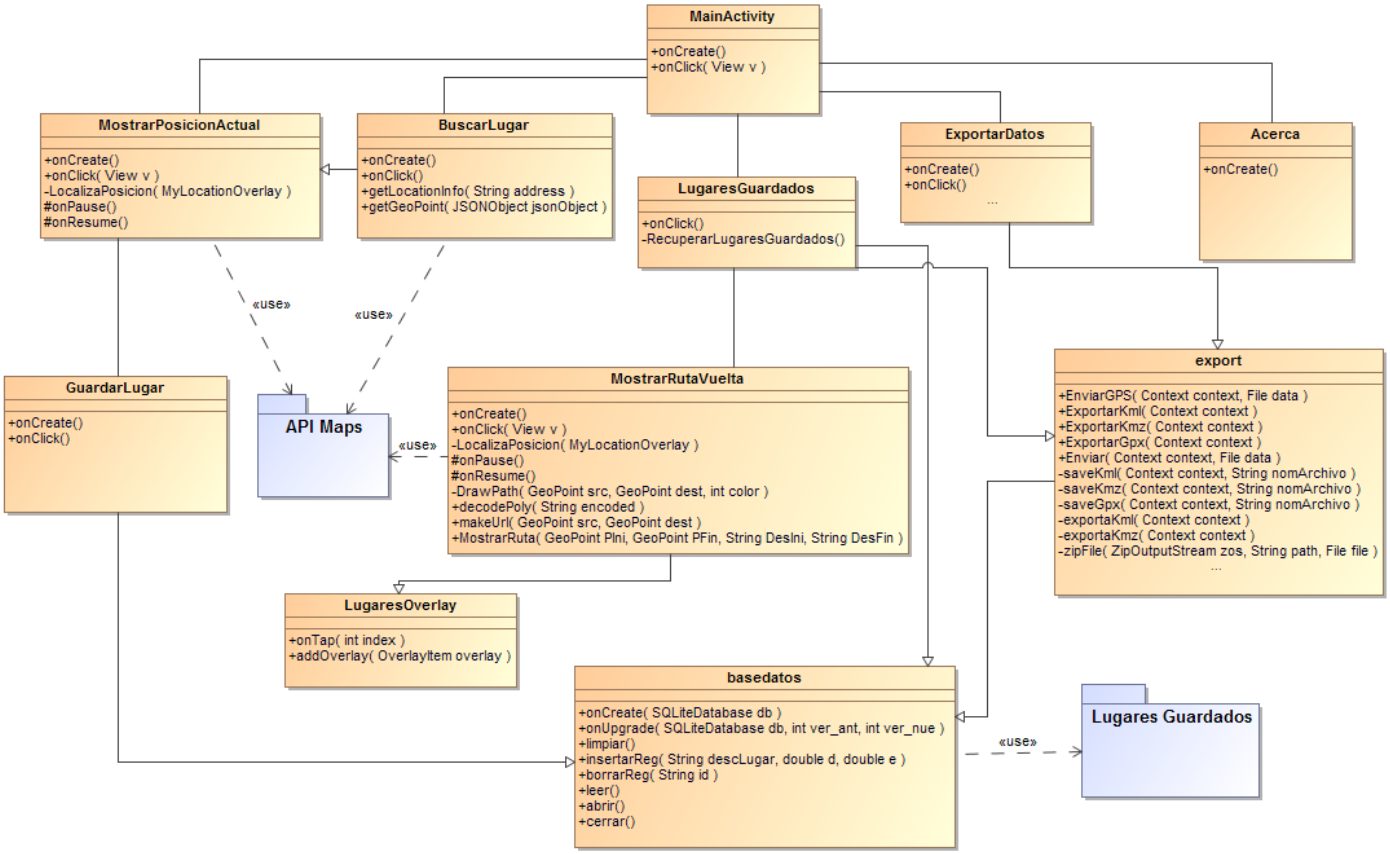

| | |
|---|---|
| **Nombre** | Guardar en la BBDD lugares la posición de un lugar buscado por el usuario |
| **Actor** | Usuario |
| **Flujo** | 1. El actor abre la aplicación en el dispositivo móvil.<br>2. El actor selección en el menú inicial la opción “Buscar lugar”<br>3. El sistema muestra por pantalla al actor una caja de texto para que introduzca una dirección o el nombre del lugar que quiere buscar.<br>4. El actor escribe en la caja de texto el nombre del lugar que quiere encontrar y pulsa en el botón “Buscar”.<br>5. El sistema se conecta a internet y busca la posición de la dirección escrita por el usuario.<br>6. Una vez localizada la posición, el sistema muestra por pantalla al actor un mapa de la zona donde se encuentra con una marca que le indica su posición exacta.<br>7. Sobre el mapa que el sistema muestra al actor, hay un botón “Guardar lugar” (Ver caso de uso “Guardar la posición del usuario en la BBDD”) |

- Diagrama caso de uso: ”Exportar la base de datos de lugares almacenados”

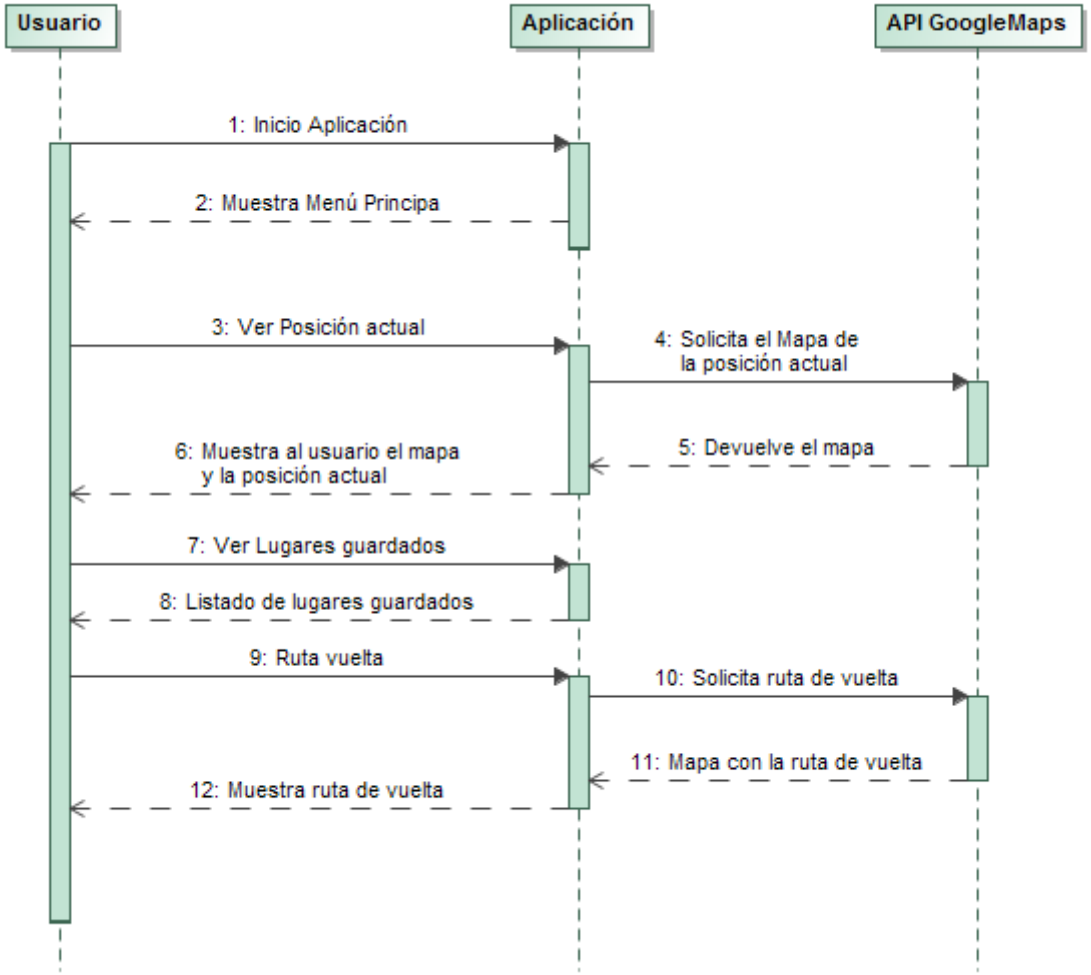



| | |
|---|---|
| **Nombre** | Exportar la base de datos de lugares almacenados |
| **Actor** | Usuario |
| **Flujo** | 1. El actor abre la aplicación en el dispositivo móvil.<br>2. El actor selección en el menú inicial la opción “Exportar datos”<br>3. El sistema muestra por pantalla al actor las opciones de exportación de la aplicación.<br>4. El actor selecciona uno entre los posibles formatos de archivos en los que puede exportar la aplicación.<br>5. El sistema genera un archivo con los datos almacenados en la base de datos en el formato que ha seleccionado el actor previamente.<br>6. El sistema muestra al usuario las opciones en las que puede enviar el archivo generado. |

## 4.3. Diagrama de clases



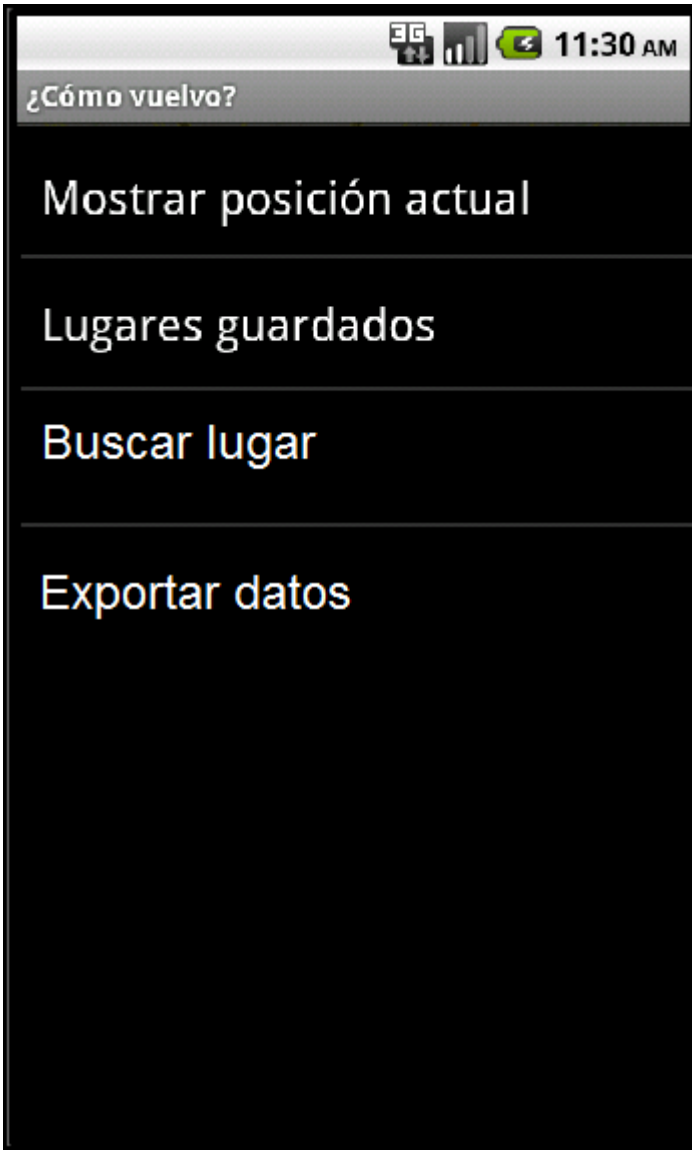
## 4.4. Diagrama de secuencias



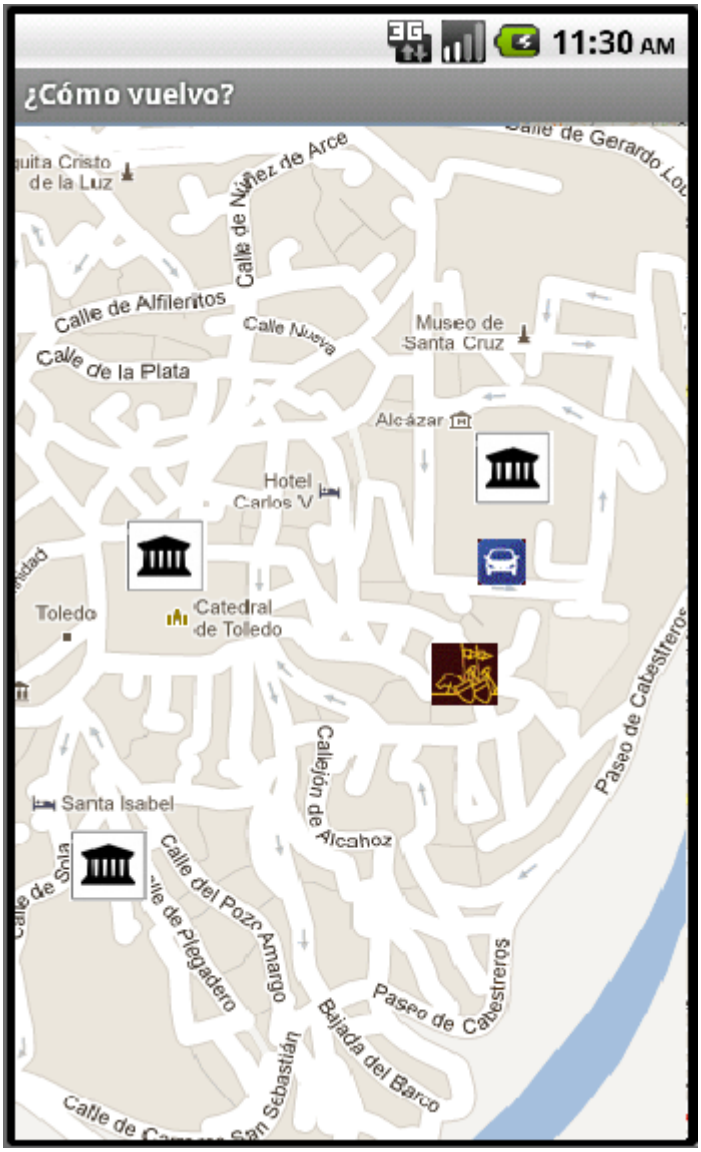
## 5. Prototipo

En el diseño del prototipo de la aplicación se trabajado tanto en el modelo de prototipo de baja fidelidad como en el de alta fidelidad. Para poder realizar encuestas a los usuarios se realizó un prototipo de baja fidelidad, que aunque era muy básico y no tenía un diseño muy elaborado, permitió que en un corto espacio de tiempo se pudiese comenzar las encuestas a los usuarios. Así les permitió hacerse una idea de cómo sería la aplicación y las tareas que realizaría.
Una vez realizadas las encuestas y vistas las posibles mejoras en el diseño de la aplicación (siguiendo el modelo de Diseño Centrado en el Usuario), se realizó este prototipo de alta fidelidad, con pantallas mucho más elaboradas y donde el usuario podrá ver el aspecto final de la aplicación.

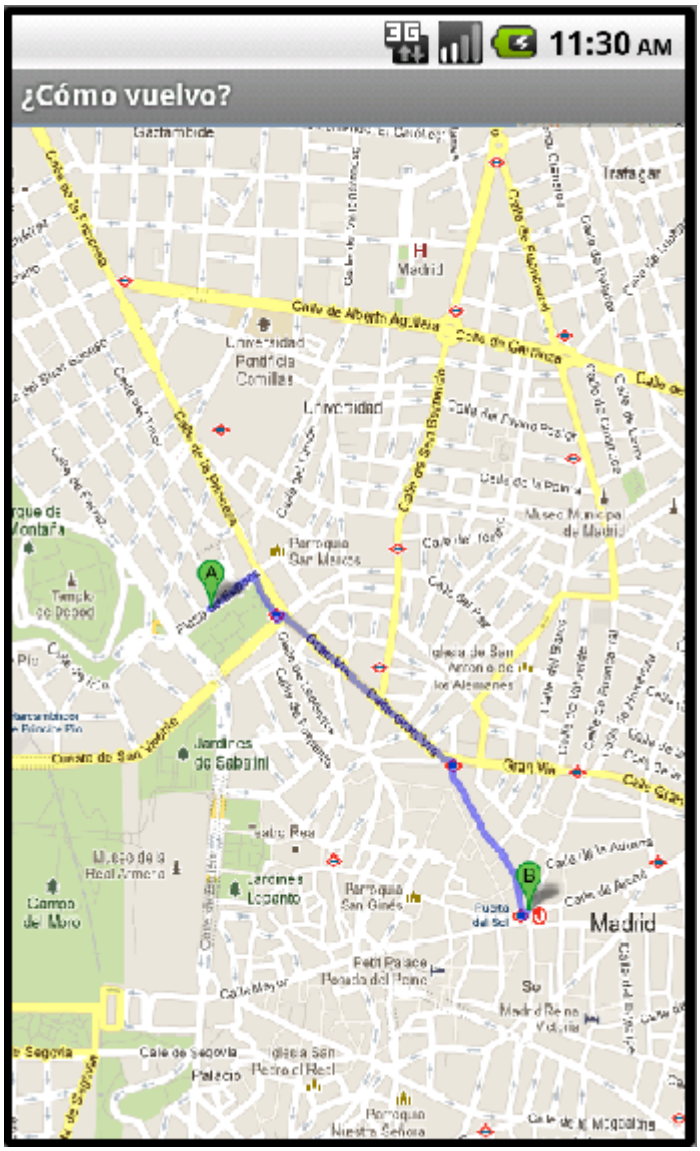
| Inicio | Posición actual |
|---|---|



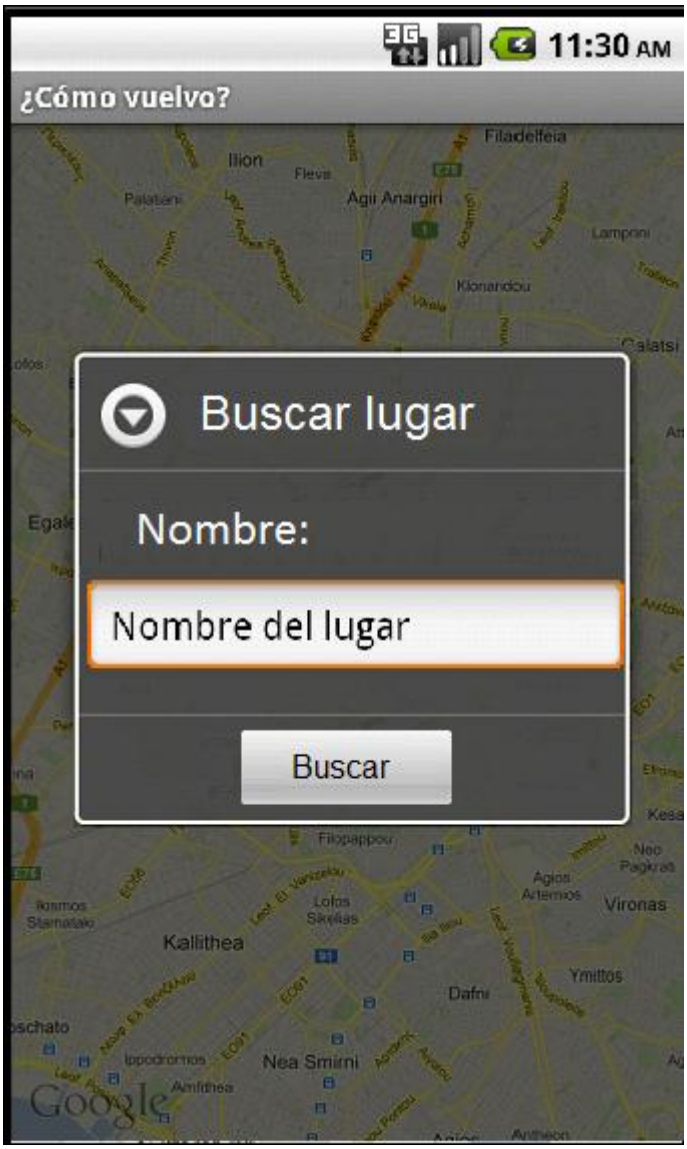

| Guardar lugar | Lugares guardados |
| --- | --- |

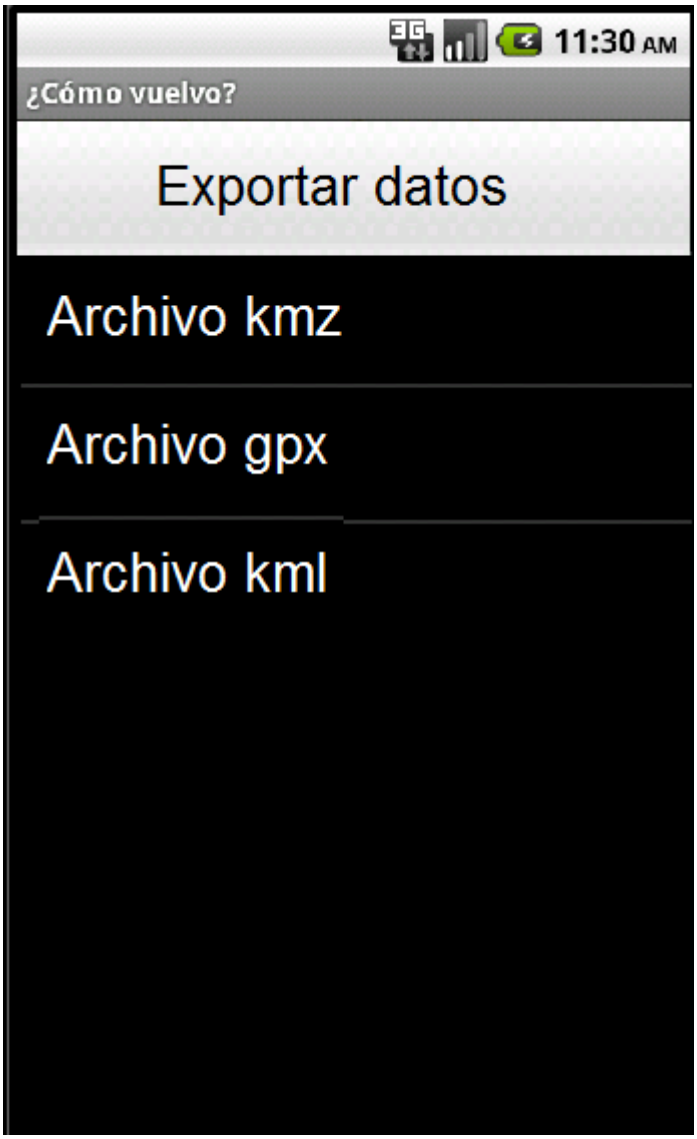



| Lugar seleccionado | Camino de vuelta |
| --- | --- |

| Buscar lugar | Exportar datos |
| --- | --- |

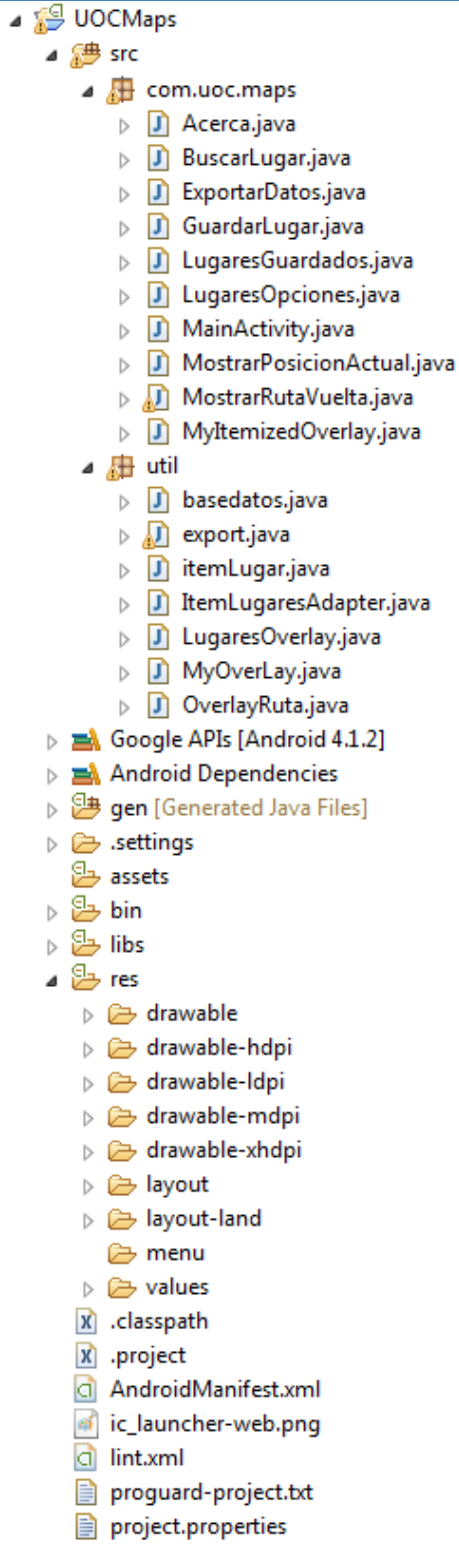


## 6. Implementación

En esta sección se detalla el código utilizado para la implementación de la aplicación y se comenta las partes más destacables del código fuente desarrollado.

**Esquema del proyecto**




Este proyecto (al igual que el resto de los proyectos desarrollados para Andorid) cuenta con la siguiente estructura de carpetas:

- src: Esta es por defecto la carpeta donde se depositará el código fuente Java. Todo el código que pongamos en las carpetas fuente será compilado cuando se requiera. Al igual que en los proyectos tradicionales de java el código se organizará en carpetas que resultaran en paquetes.
- gen: Esta es una carpeta de código fuente que va a contener archivos de Java como la carpeta src, pero no deberemos agregar o modificar los archivos contenidos ya que son generados automáticamente por el plugin de Android a partir del contenido de la carpeta res. Aquí encontramos el archivo R que es el archivo de java que servirá para indexar e identificar a todos y cada uno de los recursos de la carpeta res
- res: Esta carpeta es compleja, ya que se subdivide en diferentes directorios:
  drawable: para guardar las imágenes.
  layout: para los ficheros de configuración gráfica.
  values: donde se crean diccionarios de variables, por ejemplo Strings.
- AndroidManifest.xml: es el archivo de configuración de nuestra aplicación. En él definiremos “qué puede hacer nuestra aplicación”, es decir, informaremos al sistema de que nuestra aplicación puede abrir archivos, abrir enlaces http o que es capaz de manejar las llamadas telefónicas que recibamos. En este archivo también indicaremos las actividades que ejecutará nuestra aplicación y los permisos especiales que necesita utilizar en el caso de que quiera acceder a recursos compartidos del sistema, como el acceso a la lista de contactos, uso del GPS, o la posibilidad de enviar mensajes SMS.

En cada actividad, además de documentar su clase principal, se referencian también cada una de las demás clases usadas, así como los ficheros layout necesarios. La aplicación también está preparada para responder a los cambios de la posición del teléfono, creando layouts tanto horizontales como verticales para un perfecto acople a las dimensiones de la pantalla.

### 6.1. Clase MainActivity.class

Es la clase principal de la aplicación, extiende a Activity. En primer lugar, llama al método onCreate() de la clase, que supone el punto de entrada en el programa, y con el parámetro Bundle se crea la interfaz de la aplicación.

setContentView es un método esencial, cuya función es establecer el layout en la actividad, haciendo visibles los elementos definidos en él (se detallará más adelante).

**MainActivity.class**

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    View button1 = findViewById(R.id.BtnPosicionActual);
    button1.setOnClickListener(this);
    View button2 = findViewById(R.id.BtnLugaresGuardados);
    button2.setOnClickListener(this);
    View button3 = findViewById(R.id.BtnBuscarLugar);
    button3.setOnClickListener(this);
    View button4 = findViewById(R.id.BtnExportarDatos);
    button4.setOnClickListener(this);
    View button5 = findViewById(R.id.main_acerca_button);
    button5.setOnClickListener(this);
}
```

Una vez mostrada la actividad, el contenido visible será el especificado en el layout establecido. Para cada elemento se especifica su tamaño. Un atributo muy importante es el id, que sirve como identificador para controlar el elemento en tiempo de ejecución. Como hemos comentado anteriormente, esta clase tiene preparados dos layouts, uno en la carpeta “layout” y otro en la carpeta “layout-land”, donde el sistema aplicará uno u otro dependiendo de la posición horizontal o vertical del teléfono.

**activity_main.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/home_dashboard"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:baselineAligned="false"
    android:orientation="vertical">
    <LinearLayout android:id="@+id/home_dashboard_row1"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:gravity="center_vertical" >
        <Button android:id="@+id/BtnPosicionActual"
            android:layout_width="173dp"
            android:layout_height="wrap_content"
            android:background="@null"
            android:drawableTop="@drawable/mapa"
            android:gravity="center"
            android:text="@string/BtnPosicionActual" />
        <Button android:id="@+id/BtnBuscarLugar"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:background="@null"
            android:drawableTop="@drawable/buscar"
            android:gravity="center"
            android:text="@string/BtnBuscarLugar" />
    </LinearLayout>
    <LinearLayout android:id="@+id/home_dashboard_row2"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:gravity="center_vertical" >
        <Button android:id="@+id/BtnLugaresGuardados"
            android:layout_width="171dp"
            android:layout_height="wrap_content"
            android:drawableTop="@drawable/favoritos"
            android:gravity="center"
            android:text="@string/BtnLugaresGuardados" />
        <Button android:id="@+id/BtnExportarDatos"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:background="@null"
            android:drawableTop="@drawable/exportar"
            android:gravity="center"
            android:text="@string/BtnExportarDatos" />
    </LinearLayout>
    <LinearLayout android:id="@+id/home_dashboard_row3"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:gravity="center_vertical" >
        <Button android:id="@+id/main_acerca_button"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:background="@null"
            android:drawableTop="@drawable/acerca"
            android:gravity="center"
            android:text="@string/acerca_de" />
    </LinearLayout>
</LinearLayout>
```

### 6.2. Clase MostrarPosicionActual.class

Esta clase es donde se mostrará el mapa y donde se desarrollan los principales procesos.

MapView es la representación del mapa, MapController permite su control, mOverlays es el conjunto de los elementos pintados sobre el mapa y mOverlaylocation es el escuchador del GPS y gestor de toda la actividad.

**MostrarPosicionActual.class**

```
public class MostrarPosicionActual extends MapActivity implements OnClickListener {
            private MapView mapView;
            private MapController myMapController;
            private MyLocationOverlay mOverlayLocation;
            private List<Overlay> mOverlays;
```

Tras establecer el layout de la clase, el método onCreate() inicializa los elementos.

En primer lugar obtiene el objeto desde el layout mediante findViewById y posteriormente establece el controlador de zoom e inicializa las demás variables creadas anteriormente. Además, carga las preferencias por defecto, como mostrar el mapa en versión satélite o mostrar el control de la brújula en la pantalla.

**MostrarPosicionActual.class**

```
        public void onCreate(Bundle savedInstanceState) {

            super.onCreate(savedInstanceState);
            setContentView(R.layout.mapview);

            mapView = (MapView) findViewById(R.id.mapview);
            mapView.setBuiltInZoomControls(true);
            myMapController = mapView.getController();
            mOverlays = mapView.getOverlays();
            mOverlayLocation = new MyLocationOverlay(this, mapView);

            mOverlayLocation.enableMyLocation();
            mOverlayLocation.enableCompass();

            mapView.setSatellite(true);
```

Tenemos dos métodos importantes en esta clase, el método onPause tiene la función contraria a onResume, ya que se ejecuta cuando la actividad pasa a estado inactivo. En ese momento la aplicación va a dejar de ser utilizada, por lo que conviene liberar recursos como el GPS, teniendo en cuenta el importante consumo del dispositivo. El GPS dejará de recibir señal y de consumir batería innecesariamente. Esto no supone ningún problema, puesto que cuando la aplicación vuelva a estar activa se volverá a llamar a onResume,

**MostrarPosicionActual.class**

```
protected void onPause() {
	mOverlayLocation.disableMyLocation();
	mOverlayLocation.disableCompass();
	super.onPause();
}
protected void onResume() {
	mOverlayLocation.enableMyLocation();
	mOverlayLocation.enableCompass();
	super.onResume();
}
```

En el layout se representan los items que deben ser mostrados. A cada uno se le asigna un id, usado en los métodos anteriores.

**mapview.xml**

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >
    <com.google.android.maps.MapView
        android:id="@+id/mapview"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:apiKey="@string/maps_api_key"
        android:clickable="true"
        android:enabled="true">
    </com.google.android.maps.MapView>
    <Button
        android:id="@+id/btnGuardar"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_alignParentTop="true"
        android:text="@string/BtnGuardar" />
</RelativeLayout>
```

Es importante comentar la necesidad de generar una key que permita usar la API Maps. Esa key se inserta en Layout de la actividad, donde se instancia el MapView, contenido en un LinearLayout, sobre el que se inserta el mapa. `android:apiKey="@string/maps_api_key"`. Esta key es la que permitirá a Google saber la identidad de la aplicación que está accediendo a sus mapas, y sin la cual nos denegaría el acceso a los mismos.

En nuestro caso, hemos almacenado el valor de la key en “maps_api_key.xml” de la carpeta values.

### 6.3. Clase itemLugar.class

Se ha creado la clase itemLugar que nos ayudará a representar un lugar de una forma manejable para la aplicación. Su constructor recibe el valor de esos parámetros y desde ese momento el lugar queda completamente definido. Así en el caso de futuras versiones y querer ampliar la información almacenada sobre un lugar será mucho más sencillo su manejo.

El resto de los métodos de la clase han sido omitidos, puesto que corresponden a getters y setters para cada uno de sus campos.

**itemLugar.class**

```
public class itemLugar {
        protected long id;
        protected String nombre;
        protected int latitud;
        protected int longitud;

        public itemLugar(long id, String nombre, int lat, int lon) {
              this.setId(id);
              this.setNombre(nombre);
              this.setLatitud(lon);
              this.setLongitud(lat);
        }
```

### 6.4. Clase basedatos.class

La clase basedatos se ha creado con el fin de tener todos los métodos de las acciones sobre la base de datos en una misma clase. Al igual que la clase anterior, esto nos permitirá (además de su posible utilización en futuros proyectos, la facilidad en el caso de querer implementar nuevos métodos en la misma, dando a la aplicación un mayor número de funcionalidades.

**basedatos.class**

```
public class basedatos extends SQLiteOpenHelper{

        public basedatos (Context ctx)
        {
              super (ctx, "lugares", null, 1);
        }

        public void onCreate(SQLiteDatabase db) {
              String query = "CREATE TABLE lugares (" + _ID + " INTEGER PRIMARY KEY” +
                        + ” AUTOINCREMENT, descLugar TEXT, latitud int, longitud int);";
              db.execSQL(query);
        }

        public void onUpgrade(SQLiteDatabase db, int version_ant, int version_nue) {
              db.execSQL("DROP TABLE IF EXISTS lugares");
              onCreate(db);
        }
```

```
	public void limpiar(){
		this.getWritableDatabase().delete("lugares", null, null);
	}

	public void insertarReg(String descLugar, double d, double e){
		ContentValues valores = new ContentValues();
		valores.put("descLugar", descLugar);
		valores.put("latitud", d);
		valores.put("longitud", e);
		this.getWritableDatabase().insert("lugares", null, valores);
	}

	public void borrarReg(String id){
		String StringSQL = "DELETE FROM lugares WHERE " + _ID + "=" + id;
	    getWritableDatabase().execSQL(StringSQL);
	}

	public String[][] leer(){
		String columnas[]={_ID,"descLugar","latitud","longitud"};
		Cursor c = this.getReadableDatabase().query("lugares", columnas, null,
				null, null, null, null);
		String result[][]=new String[c.getCount()][4];

		int id,idDesc,idLati,idLong;
		id = c.getColumnIndex(_ID);
		idDesc = c.getColumnIndex("descLugar");
		idLati = c.getColumnIndex("latitud");
		idLong = c.getColumnIndex("longitud");

		int contador=0;
		for (c.moveToFirst(); !c.isAfterLast(); c.moveToNext())
		{
			result[contador][0]=c.getString(id);
			result[contador][1]=c.getString(idDesc);
			result[contador][2]=c.getString(idLati);
			result[contador][3]=c.getString(idLong);
			contador++;
		}
		c.close();
		return result;
	}

	public void abrir(){
		this.getWritableDatabase();
	}

	public void cerrar(){
		this.close();
	}
}
```

### 6.5. Clase export.class

La clase export ha sido creada con el fin de proporcionar los métodos necesarios para que la aplicación pueda exportar los datos de los lugares almacenados en la base de datos en diferentes formatos para su posterior reutilización en diferentes aplicaciones, ya sea GoogleMaps o cualquier otra que permita leer los formatos a los que exporta la aplicación.

Actualmente la aplicación tiene implementada la funcionalidad para exportar a ficheros GPX, KML y KMZ y puede enviarlos en cualquier forma que tenga configurado el dispositivo móvil, ya sea sms o email.

**export.class**

```
public class export {

	public void EnviarGPS(Context context, File data, int lat, int lon){
		String message =
			"http://maps.google.com/maps?q=” + lat + “,” + lon + “&iwloc=A";

		Intent sendIntent = new Intent(Intent.ACTION_SEND);
		sendIntent.putExtra(Intent.EXTRA_TEXT, message);
		sendIntent.setType("text/plain");
		context.startActivity(Intent.createChooser(sendIntent, "Enviar a..."));
	}

	public void ExportarKml(Context context){
		File data=null;
		data = saveKml(context, "como_vuelvo");
		Enviar (context, data);
	}

	public void ExportarKmz(Context context){
		File data=null;
		data = saveKmz(context, "como_vuelvo");
		Enviar (context, data);
	}

	public void ExportarGpx(Context context){
		File data=null;
		data = saveGpx(context, "como_vuelvo");
		Enviar (context, data);
	}

	private void Enviar(Context context, File data){

		Intent sendIntent = new Intent();
		sendIntent.setAction(Intent.ACTION_SEND);

		sendIntent.putExtra(Intent.EXTRA_STREAM, Uri.fromFile(data));
		sendIntent.setType("text/plain");
		context.startActivity(Intent.createChooser(sendIntent, "Enviar a..."));
	}

	private File saveKml(Context context, String nombreArchivo)
	{
		File data = null;
		try
		{
```

```
                data = File.createTempFile(nombreArchivo, ".kml");
                FileWriter out = new FileWriter(data);
                out.append(exportaKml(context));
                out.flush();
            }
            catch (Exception ex)
            {
                Log.e("Ficheros", "Error al escribir fichero a memoria interna");
            }
            return data;
    }

    private File saveKmz(Context context, String nombreArchivo)
    {
            File data = null;
            File dataZip = null;
        System.out.println("Packaging to " + nombreArchivo);

        data=saveKml (context, nombreArchivo);
        ZipOutputStream zipOut = null;

        try {
           dataZip = File.createTempFile(nombreArchivo, ".kmz");
           FileOutputStream aa = new FileOutputStream(dataZip);
             zipOut = new ZipOutputStream(aa);
             zipOut.setLevel(Deflater.DEFAULT_COMPRESSION);

                zipFile(zipOut, "", data);

             zipOut.flush();
             zipOut.close();

           } catch (IOException e) {
                e.printStackTrace();
           }

        System.out.println("Done");
        return dataZip;
    }

    private File saveGpx(Context context, String nombreArchivo)
    {
            File data = null;
            try
            {
                data = File.createTempFile(nombreArchivo, ".gpx");
                FileWriter out = new FileWriter(data);
                out.append(exportaGpx(context));
                out.flush();
            }
            catch (Exception ex)
            {
                Log.e("Ficheros", "Error al escribir fichero a memoria interna");
            }
            return data;
    }
```

### 6.6. Clase LugaresOverlay.class

LugaresOverlay, que extiende a la clase ItemizedOverlay, es la que permite situar al usuario sobre el mapa. El principal elemento de la clase es mapOverlays, que será donde se guarde el elemento a posicionar. El método addOverlay es el que nos permitirá añadir nuevos elementos a esa lista. La llamada populate() es la que se encarga de pintar los puntos sobre el mapa.

**LugaresOverlay.class**

```
public class LugaresOverlay extends ItemizedOverlay<OverlayItem> {

   private ArrayList<OverlayItem> mapOverlays = new ArrayList<OverlayItem>();
   private Context context;

   public LugaresOverlay(Drawable defaultMarker) {
        super(boundCenterBottom(defaultMarker));
   }

   public LugaresOverlay(Drawable defaultMarker, Context context) {
        this(defaultMarker);
        this.context = context;
   }

   protected OverlayItem createItem(int i) {
      return mapOverlays.get(i);
   }

   public int size() {
      return mapOverlays.size();
   }

   protected boolean onTap(int index) {
      OverlayItem item = mapOverlays.get(index);
      AlertDialog.Builder dialog = new AlertDialog.Builder(context);
      dialog.setTitle(item.getTitle());
      dialog.setMessage(item.getSnippet());
      dialog.show();
      return true;
   }

   public void addOverlay(OverlayItem overlay) {
      mapOverlays.add(overlay);
       this.populate();
   }
```

El método onTap, que se ejecuta cuando el usuario toca sobre la posición del Overlay en el mapa, mostrará un mensaje con la información del lugar.

### 6.7. Strings.xml

Strings.xml es el fichero donde se incluyen los valores de cada tipo de variables. En esta versión de la aplicación sólo está disponible en castellano, pero para futuras versiones no hay más que crear ficheros con string_idioma.xml con las distintas definiciones en el correspondiente idioma y decirle a la aplicación que cargue ese fichero. Incluso, se podría añadir una opción de preferencias a la aplicación donde el usuario pudiese modificar el idioma en tiempo de ejecución.

**strings.xml**

```
<resources xmlns:xliff="urn:oasis:names:tc:xliff:document:1.2">
    <string name="app_name">¿Cómo vuelvo?</string>
    <string name="menu_settings">Salir</string>
    <string name="lblBuscar">Escriba el lugar a buscar:</string>
    <string name="acerca_de">Acerca de</string>
    <string name="lblGuardar">Escriba el lugar a guardar:</string>
    <string name="btnBuscar">Buscar</string>
    <string name="BtnPosicionActual">Mostrar Posición Actual</string>
    <string name="BtnLugaresGuardados">Lugares Guardados</string>
    <string name="BtnBuscarLugar">Buscar Lugar</string>
    <string name="BtnExportarDatos">Exportar Datos</string>
    <string name="BtnGuardar">Guardar Posición</string>
    <string name="btnEnviarPosicion">Enviar Posición</string>
    <string name="btnMostrarPosicion">Mostrar Posición</string>
    <string name="btnRutaPosicionCoche">Mostrar ruta de vuelta en coche</string>
    <string name="btnRutaPosicionPie">Mostrar ruta de vuelta a pie</string>
    <string name="btnEliminar">Eliminar</string>
    <string name="title_activity_mapview">Mapview</string>
    <string name="title_activity_main">MainActivity</string>
    <string name="lista_vacia">No hay lugares guardados</string>
    <string name="BtnExportarKml">Exportar como KML</string>
    <string name="BtnExportarKmz">Exportar como KMZ</string>
    <string name="BtnExportarGpx">Exportar como GPX</string>
</resources>
```

### 6.8. AndroidManifest.xml

El Maniesto es un archivo imprescindible en toda aplicación Android, debe tener exactamente este nombre y estar situado en el directorio raíz. Es el archivo de configuración de nuestra aplicación. En él definiremos "qué puede hacer nuestra aplicación", es decir, informaremos al sistema de que nuestra aplicación puede abrir archivos, abrir enlaces http o que es capaz de manejar las llamadas telefónicas que recibamos. En este archivo también indicaremos las actividades o servicios que ejecutará nuestra aplicación y los permisos especiales que necesita utilizar en el caso de que quiera acceder a recursos compartidos del sistema, como el acceso a la lista de contactos, uso del GPS, o la posibilidad de enviar mensajes SMS.

De este fichero se extrae información esencial para una correcta ejecución. En nuestro caso, la información que contiene es la siguiente:

- Package: "com.uoc.maps". Sirve de identificador único para la aplicación.
- VersionCode: 1. VersionName: 1.0 Nombre de la versión.
- minSdkVersion: 8. Versión mínima del SDK requerida para el correcto funcionamiento.
- Permisos: INTERNET, ACCESS_COARSE_LOCATION, FINE LOCATION y WRITE_EXTERNAL_STORAGE.
- Logo de la aplicación: @drawable/ic_launcher
- Actividades con sus correspondientes etiquetas.
- Librerías: com.google.android.maps

**LugaresOverlay.class**

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.uoc.maps" android:versionCode="1" android:versionName="1.0" >

    <uses-sdk android:minSdkVersion="8" android:targetSdkVersion="15" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"></uses-
permission>
    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name" android:theme="@style/AppTheme" >
        <uses-library android:name="com.google.android.maps" />
        <activity android:name=".MainActivity" android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".BuscarLugar" />
        <activity android:name=".ExportarDatos" />
        <activity android:name=".MostrarPosicionActual" />
        <activity android:name=".MostrarRutaVuelta" />
        <activity android:name=".GuardarLugar" />
        <activity android:name=".LugaresGuardados" />
        <activity android:name=".LugaresOpciones" />
        <activity android:name=".Acerca" />
        <uses-library android:name="com.google.android.maps" />
      </application>
</manifest>
```

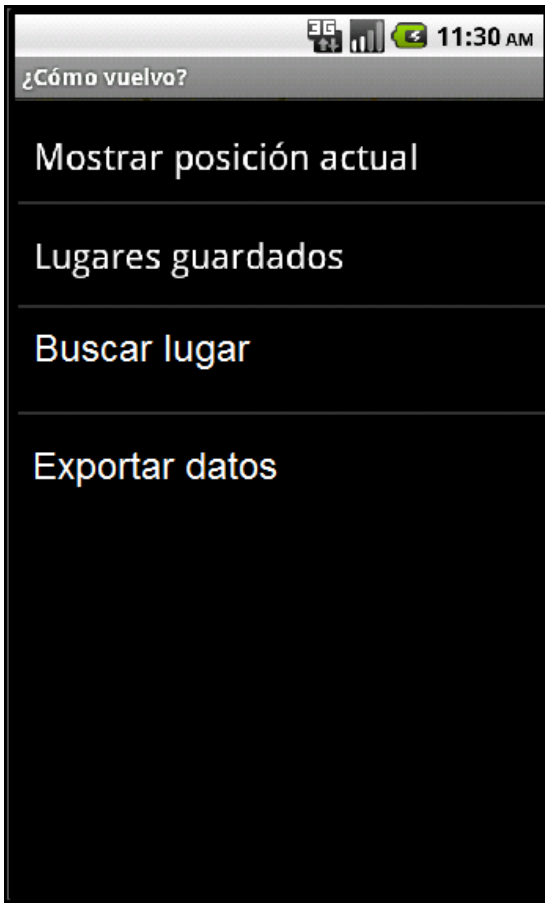
# 7. Diseño final de la aplicación

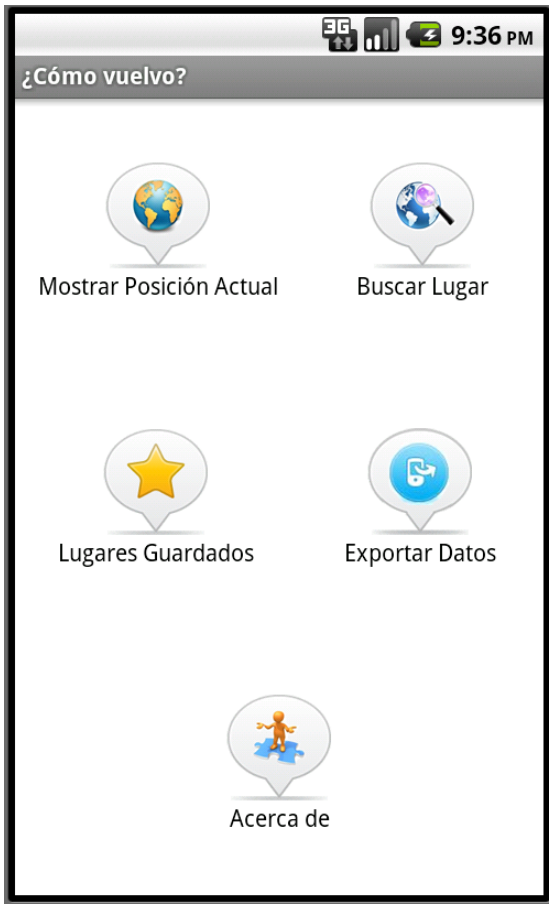
El diseño de la aplicación que había en el prototipo ha sido modificado por completo y para ello se ha seguido las pautas marcadas del modelo DCU (Diseño Centrado en el Usuario).

## 7.1. Menú principal

- En primer lugar se ha modificado el icono de la aplicación para que sea reconocible en el menú del teléfono a simple vista y el usuario pueda localizarla sin problemas.
- Se ha modificado el menú inicial con una lista de opciones por un diseño más gráfico, con iconos descriptivos de la acción y que resulten más intuitivos al usuario. Todos los iconos son animados y cuando se pulsa sobre ellos cambian de color para que el usuario pueda saber en todo momento cuál es el botón que está pulsando.
- Se ha añadido una opción más respecto al diseño del prototipo, “Acerca de”, donde el usuario podrá acceder a los datos descriptivos tanto de la aplicación como del creador.

| Diseño Prototipo | Diseño Final |
|---|---|

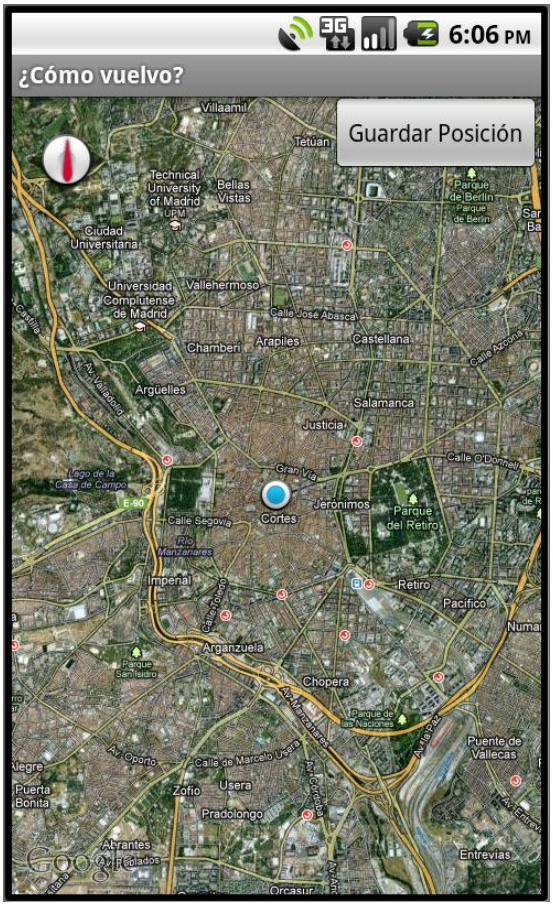


## 7.2. Mostrar posición actual

- Se ha añadido la opción de la brújula que no estaba contemplada en el prototipo para ayudar al usuario a orientarse.
- Se ha modificado el mapa para que sea en modo satélite que da un mayor realismo y ayuda más al usuario a localizarse en el mapa.
- El tamaño del botón de "Guardar posición" se ha reducido para dejar espacio a nuevas opciones previstas para las nuevas versiones:
    - Previsto: sustituir el botón "guardar posición" por un icono descriptivo y que no ocupe media pantalla.
    - Previsto: añadir un botón que permita modificar la vista del MapView, de vista satélite a vista en mapa.

| Diseño Prototipo | Diseño Final |
|---|---|

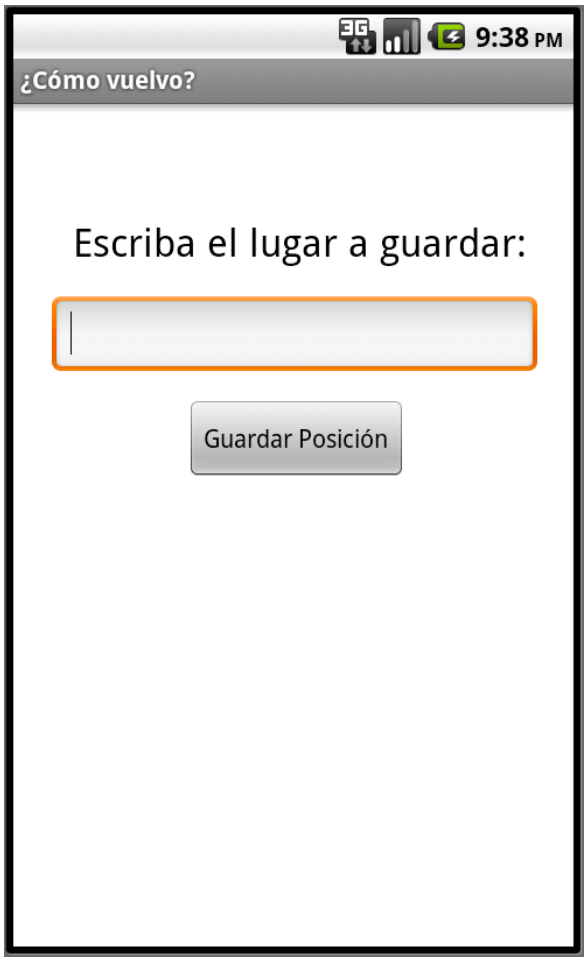



## Localización GPS


En esta nueva versión, al abrir una pantalla en la que sea necesaria la localización del terminal móvil del usuario, se oscurece la pantalla y se muestra el mensaje “Localizando posición por el GPS”. De esta manera el usuario sabe que la aplicación está ocupada tratando de localizar su posición en el mapa.

Una vez localizado, el mensaje desaparece y deja ver el mapa claramente y señala la posición en la que se encuentra el usuario.

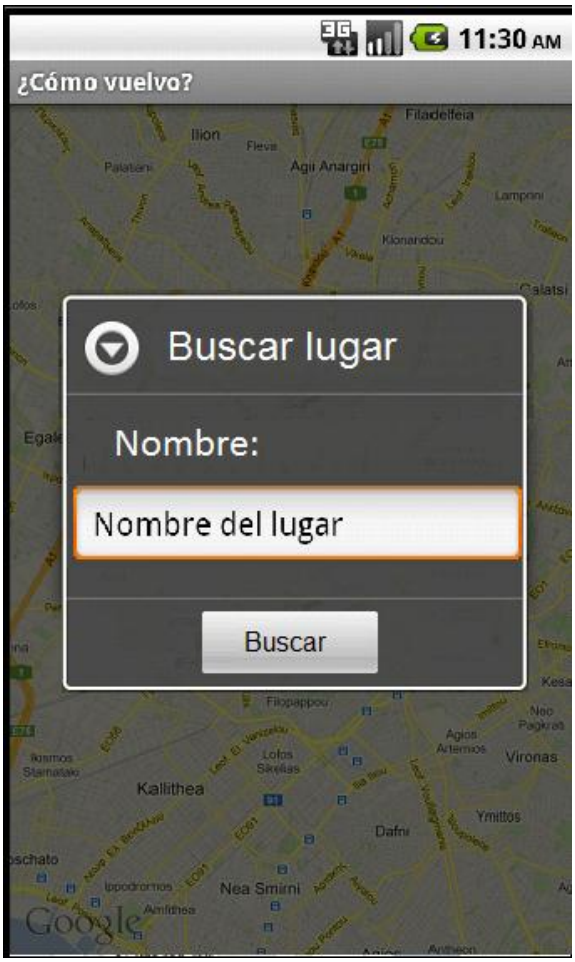
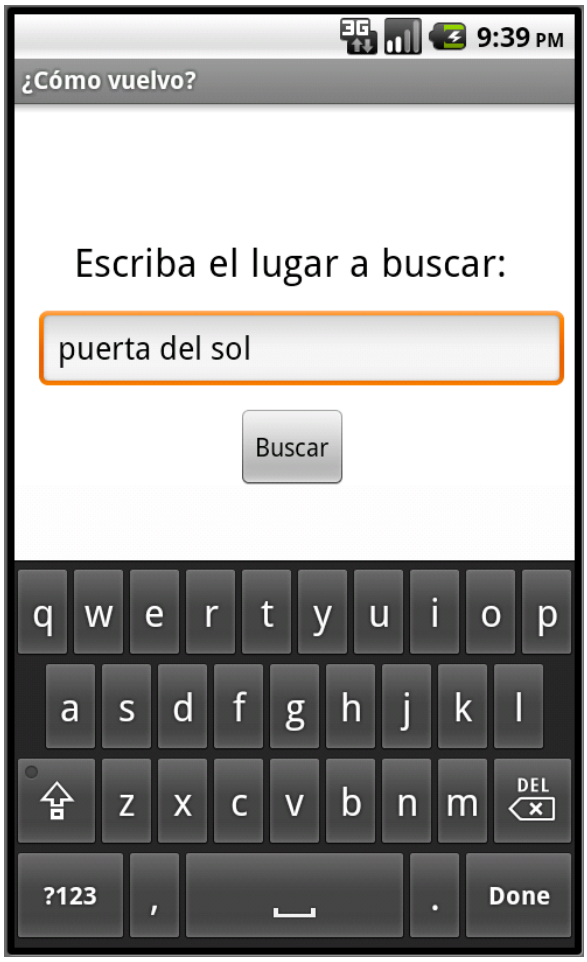
### 7.3. Guardar lugar en la base de datos

| Diseño Prototipo | Diseño Final |
| --- | --- |
|  |  |



La aplicación muestra ahora una nueva ventana donde el usuario puede escribir el nombre con el que quiere guardar la posición que tiene en el mapa. Como se puede apreciar, en el prototipo se mantenía la imagen de fondo del mapa de Google. Tras varios ensayos de percepción visual se decide eliminar el mapa en el fondo de la pantalla para que tenga mucha más claridad y evite que usuarios con problemas de visión no puedan ver correctamente el formulario.

### 7.4. Buscar un lugar

| Diseño Prototipo | Diseño Final |
| --- | --- |
|  |  |



Al igual que en el formulario de guardar lugares, y siguiendo el mismo criterio se ha eliminado el mapa del fondo de la pantalla para que tenga mucha más claridad y evite que usuarios con problemas de visión puedan no ver correctamente el formulario.

## 7.5. Lugares guardados

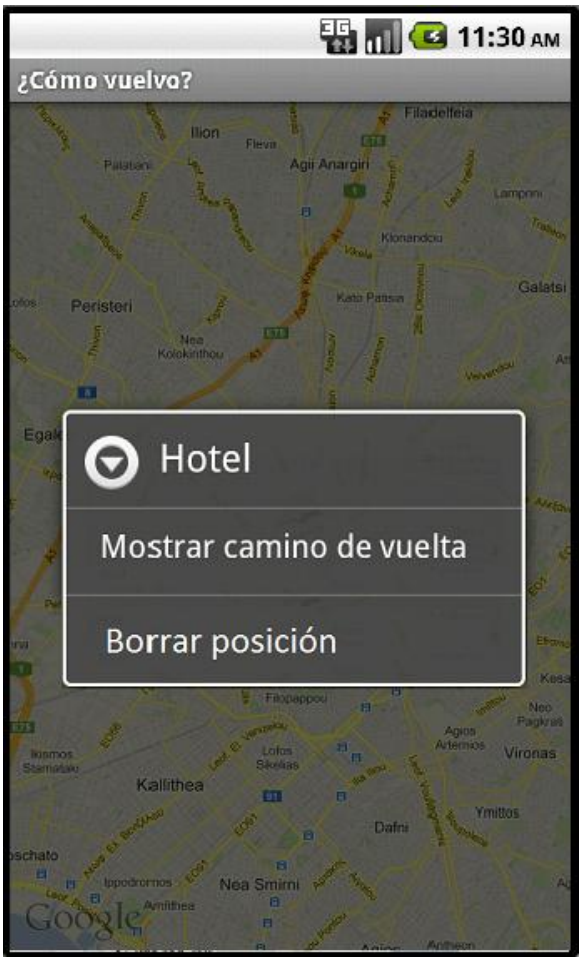
**Diseño Final**


Se ha creado una nueva pantalla donde el usuario va a poder consultar todos los lugares que tenga almacenados en la base de datos de su dispositivo móvil.

Se ha diseñado de forma que se vea claramente el nombre del lugar guardado y, como datos adicionales, se le muestra al usuario la latitud y la longitud de la posición que ha almacenado en la base de datos.

Se podrán almacenar tantas posiciones como el usuario el necesite, para tener toda la información de su viaje al alcance, los puntos frecuentes o habituales de partida y regreso, de manera rápida y sencilla.

### 7.6. Menú – Lugar guardado

| Diseño Prototipo | Diseño Final |
|---|---|
|  |  |



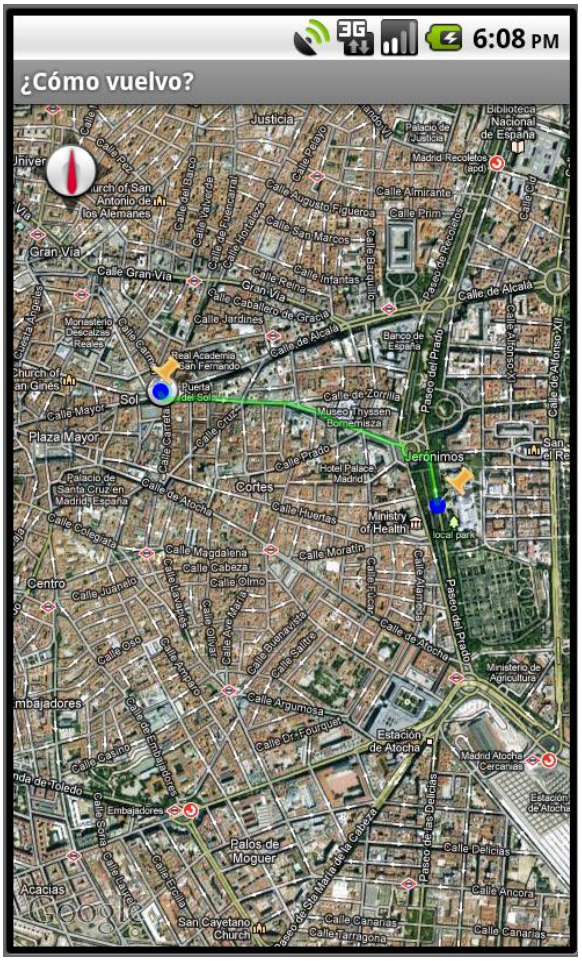
Este es uno de los formularios que más variación ha sufrido con respecto del diseño del prototipo. Al igual que en la pantalla inicial, se ha modificado el listado de opciones por iconos con imágenes descriptivas de la acción que realiza el botón (no se elimina el texto para que no haya dudas en lo que realiza cada acción).

Y como en los otros caso también se elimina el mapa de fondo que pude molestar para tener una visión nítida de las opciones.

Anteriormente sólo existían dos opciones: mostrar la ruta de vuelta y eliminar el lugar guardado. En esta versión se ha dividido "mostrar ruta de vuelta" por dos nuevas opciones, "mostrar ruta de vuelta a pie" y "mostrar ruta de vuelta en coche", donde se le mostrará al usuario el camino de regreso más corto dependiendo de la forma de volver. También se ha añadido la opción de enviar los datos por email u otro medio que soporte el dispositivo móvil.

## 7.7. Ruta a pie / Ruta en coche

**Mostrar Ruta a pie / coche**

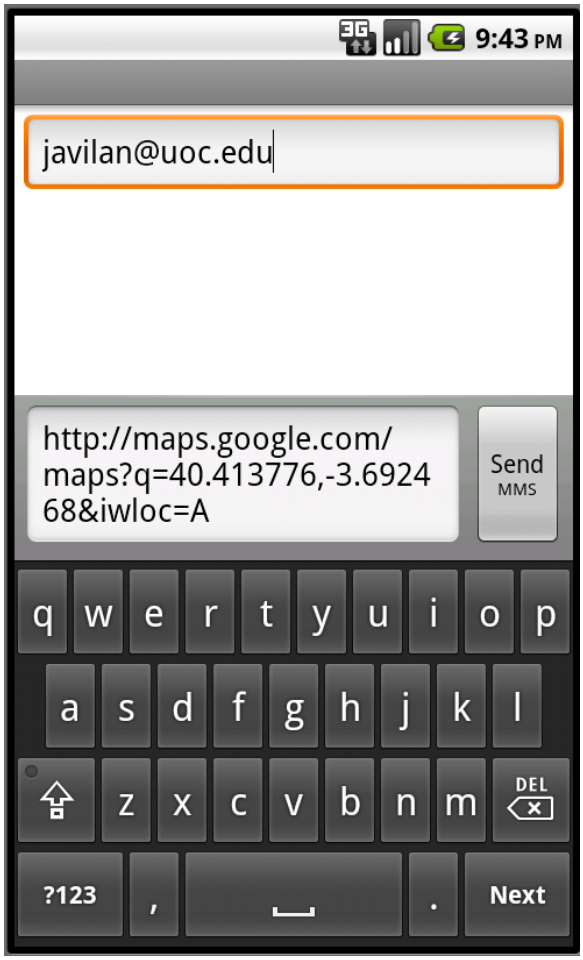


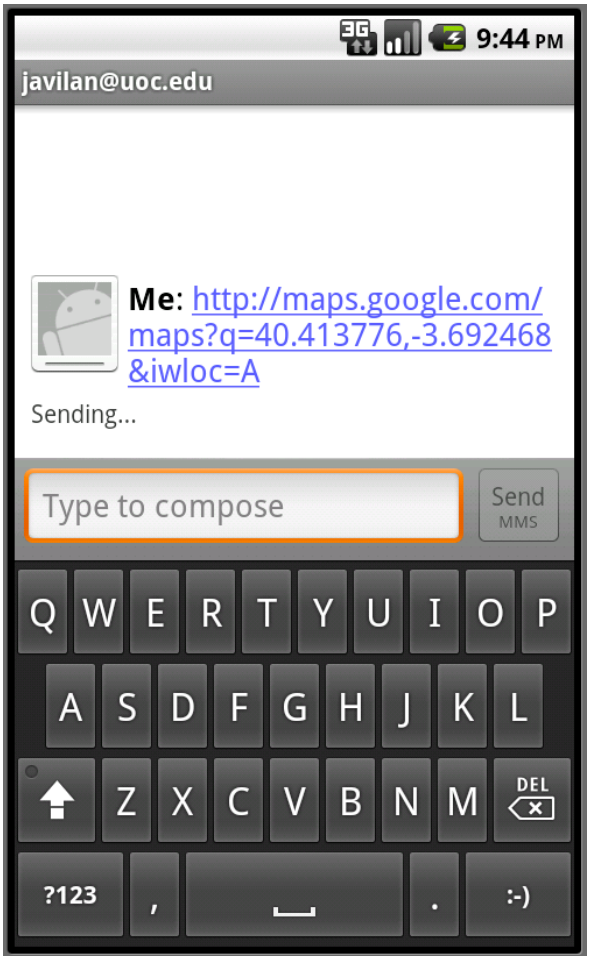

Esta pantalla se ha añadido en el diseño final con la intención de poder ofrecer alternativas a la ruta seleccionada, de manera que el usuario pueda valorar la distancia a la que tiene su destino siguiendo la ruta a pie o en vehículo privado, ya que en ocasiones estas pueden variar de forma notable.

Con esta nueva pantalla se añade otro factor más a la aplicación, además de resolver el problema de cómo volver a un punto previamente seleccionado, sin necesidad de anotar la dirección exacta de partida, el usuario también podrá valorar la manera de volver que más le interese, y de esta manera puede ahorrar tiempo y dinero en caso de no tener vehículo propio.

## 7.8. Compartir un lugar

**Compartir lugar**

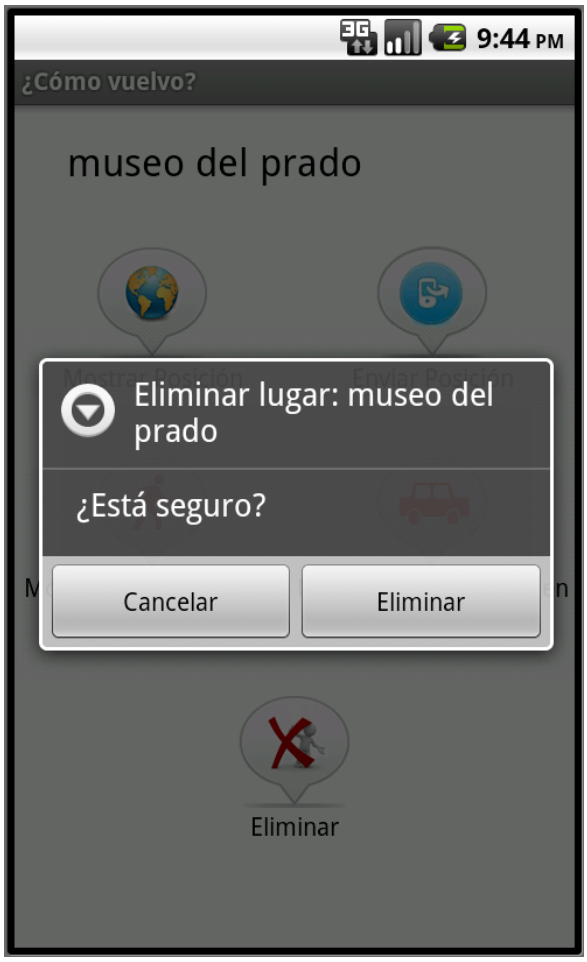



La función de enviar la posición desde el dispositivo móvil permite, no solo almacenar la ruta para el mismo usuario de la aplicación en su propio correo electrónico, sino también poder compartirla con otros usuarios que necesiten tener información de los mismos recorridos o que compartan viaje y quieran guardar las mismas posiciones. De esta forma no es necesario que todos tengan conectado el GPS en el momento de localizar el punto de partida, bastará con que abran el enlace de su dispositivo en el momento de querer volver.

## 7.9. Eliminar un lugar de la base de datos

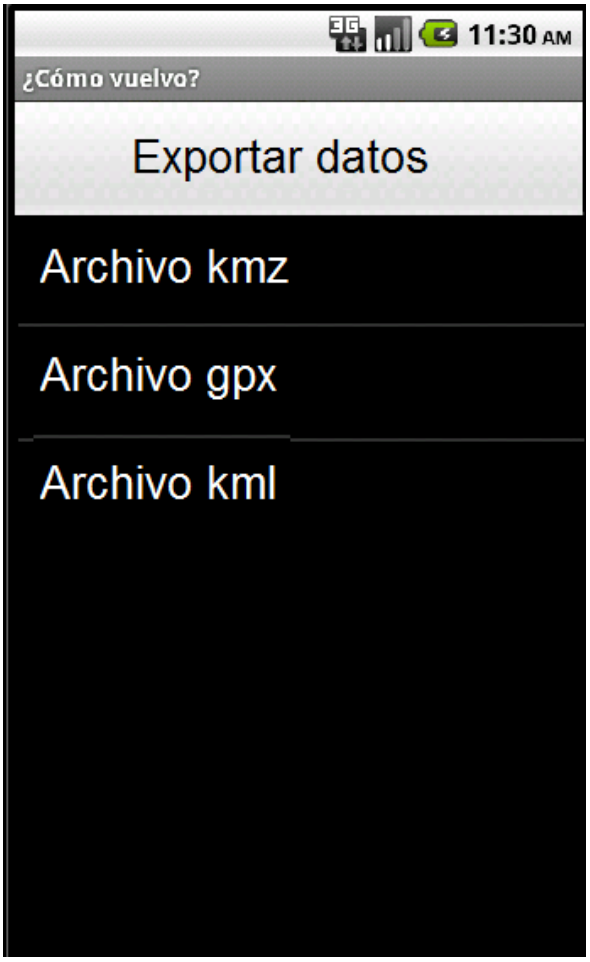
**Eliminar un lugar**




En esta nueva versión, al pulsar sobre el icono de eliminar, se abrirá una ventana donde se le mostrará al usuario claramente el nombre del elemento que está a punto de borrar. Además, se le pedirá confirmación de la acción, puesto que no tiene marcha atrás y una vez borrado el elemento no se podrá recuperar.

De esta forma puede tener la posibilidad de no almacenar lugares a los que ya no va a volver y que pueden confundirle en otras búsquedas, siempre intentando evitar la pérdida involuntaria de puntos de referencia.

### 7.10. Exportar listado de lugares guardados

| Diseño Prototipo | Diseño Final |
|---|---|
|  |  |





La función de exportar los datos en esta aplicación tiene un papel importante, ya que una de las principales características era poder enviar los datos para que fueran accesibles por otras aplicaciones. Actualmente el sistema exporta en los tres formatos más utilizados de geolocalización: KML, KMZ y GPX, pero está abierto a que en futuras versiones se puedan añadir nuevos formatos.

### 7.11. Acerca de…

**Acerca de**


Esta pantalla tiene carácter meramente informativo. Se crear para dar información al usuario de los datos generales de la misma.
En un futuro se utilizará para informar a los usuarios de las diferentes modificaciones de las versiones que vayan surgiendo.

Para ayudar a dar a conocer la aplicación, así como todo la información que haya sobre la misma o incluso un foro donde los usuarios puedan exponer sus dudas, problemas sobre el manejo de la misma, se ha creado la siguiente web:

http://como-vuelvo.blogspot.com.es/

## 8. Bibliografía


- Android, el sistema operativo de Google
  www.android.com
- Página oficial para desarrolladores de Google
  https://developers.google.com
- http://www.paratuandroid.com/Table/Desarrollo/
- Foro desarrolladores de aplicaciones Android
  http://stackoverflow.com/questions/tagged/android
- http://www.elandroidelibre.com
- Foro de programación
  http://stackoverflow.com