MÓDULO DE CONTROL DE INVENTARIO DE AMBULANCIAS PARA LA APLICACIÓN MÓVIL SAPHCA

Varela, Erick.

Departamento de Electrónica e Informática, Universidad Centroamericana “José Simeón Cañas”, El Salvador evarela@uca.edu.sv

***Abstract―*El manejo de datos siempre ha sido un gran problema para las personas, ya que todos los datos por lo general siempre se han llevado en registros físicos que ocupan un espacio muy amplio en oficinas, empresas, entre otros. Todo ello ha cambiado gracias al auge de la tecnología, específicamente la informática, ya que ahora se puede almacenar millones de registros en una sola computadora, que ocupa un espacio mucho más reducido que una habitación con muchos archiveros, además los beneficios son mucho más, ya que por ejemplo para buscar un registro era necesario mantener un orden estricto en los lugares donde se almacenaban, y solo lo podía consultar una persona a la vez, en cambio con las nuevas tecnologías los datos pueden ser vistos desde una o más computadoras al mismo tiempo, además de poder ser visualizados en dispositivos móviles, todo ellos a través de conexiones de red local o inalámbrica. Muchas empresas e instituciones gubernamentales están implementando estas tecnologías con el fin de facilitar el manejo de datos, entre ellos el Sistema de Emergencias Médicas del Ministerio de Salud, que posee un dispositivo móvil en cada ambulancia que tiene una aplicación móvil que realiza la función de coordinar,**

*Palabras clave*―

# INTRODUCCIÓN

El avance tecnológico que se ha logrado en los últimos años, encamina a una era digital en la que las antiguas prácticas están quedando cada día más obsoletas, como por ejemplo para la reservación de una cita al momento de tramitar algún documento, se hacía de manera telefónica o en el peor de los casos de manera personal, hoy en día para reservar una cita basta con tener una computadora o algún dispositivo móvil conectado a una red local o inalámbrica para poder realizar dicha cita, al hacer esto los datos son tomados de manera digital y el usuario puede verificar que los datos sean correctos, y estar seguro que no se suscitara algún error, como por ejemplo la persona encargada de tomar los datos podía tener algún error de escritura y este provocar la pérdida de la cita del usuario, además todos estos registros al estar de forma física durante unos diez años, se acumulaba una inmensa cantidad de estos, con la implementación de sistemas que permiten almacenar dichos registros de manera digital, el espacio ocupado por los registros físicos se vio muy reducido.

Otro ejemplo sería el control de inventario de cierta ambulancia, el caso hipotético de una ambulancia que cuenta con un inventario asignado mensualmente, al atender una emergencia se utilizan medicamentos e insumos del inventario, una vez se finalizaba de atender la emergencia, se registraba de manera física el uso de los medicamentos o insumos, estos registros tenían que ser guardados para poder saber la cantidad de medicamento o insumo que quedaban en el inventario, y así con estos reportes se tenían que pasar muchas horas para la elaboración de reportes mensuales. El cambio se presentó cuando la tecnología se empezó a utilizar para estos casos, ya que con la implementación de un sistema que permitiera el control del inventario de forma digital en las ambulancias, estos procesos de minutos se verían reducidos a segundos, con la implementación de herramientas que permiten el desarrollo de estas aplicaciones en dispositivos móviles, ya que ahora el registro del inventario se hace en la aplicación, y en vez de llenar el formulario con los medicamentos e insumos utilizados en las emergencias, ahora solo basta con ingresar a la aplicación, seleccionar del inventario actual los medicamentos e insumos utilizados y listo, al mismo tiempo que se actualiza el inventario y se ve de manera inmediata la cantidad de inventario que ha quedado. Con respecto al tiempo utilizado para la elaboración de reportes se ve reducida de horas a minutos, ya que al llevar un registro de los medicamentos e insumos utilizados en un mes, solamente bastaría con seleccionar en la aplicación el periodo de tiempo, en este caso un mes y se mostrarían en el dispositivo el inventario utilizado en dicho periodo de tiempo, con esto bastaría que la aplicación tenga implementada una manera de imprimir los resultados mostrados en pantalla para tener un reporte de los medicamentos o insumos utilizados en un mes, todo ello en cuestión de un par de minutos.

En la figura 1, se muestra un ejemplo sencillo de un control de inventario en una aplicación diseñada para un dispositivo móvil, en ella se puede observar la identificación de la ambulancia, para llevar un registro por separado para cada una, la descripción del medicamento, acompañado de la fecha de vencimiento y la cantidad disponible, con dos opciones muy necesarias como por ejemplo el ajuste del inventario, ya que se puede suscitar el caso que el medicamento sufra algún incidente por lo que sea necesario dar de baja el mismo, la otra opción aún más vital como lo es el registro de consumo de medicamentos, en la que el usuario registrará la salida de medicamento utilizado en una emergencia, como se puede observar todo esto parece muy sencillo, pero el desarrollo de dicha aplicación también conlleva cierto tiempo de elaboración, después un período de tiempo en el que debe ser probada por los desarrolladores de la misma que intentarán buscar problemas en su funcionamiento con el fin de corregir fallas y que sea lo más confiable posible. Además, se requiere de software especializado para el desarrollo de estas, y herramientas que por ejemplo permitan que los registros que se almacenan en los dispositivos móviles pasen a una computadora central donde estos registros son almacenados, y al obtener estos registros en un solo punto se podrían elaborar registros comparativos del consumo de inventario de todas las ambulancias.

Una de las herramientas para la consumación de este proceso es Rest API, que permite la transferencia de esta información desde un dispositivo móvil a una computadora central haciendo uso de conexión a internet. Toda esta información es manipulada a través de otra herramienta muy importante como lo son las bases de datos, que permite mantener un control centralizado de los datos, además de ofrecer seguridad a los datos y un mejor manejo de los mismos.



*Fig. 1 Ejemplo de un control de inventario de una ambulancia.*

# MATERIALES Y MÉTODOS

1. *Requerimientos*

Para el desarrollo de un TileProvide para dispositivos Android es necesario tener instalado:

* Java developer Kit 8(JDK)
* Android Studio
* Android SDK
* Google Maps API

1. *Pasos para el desarrollo*

Para la creación e implantación de un TileProvider se siguieron los siguientes pasos:

* Crear un proyecto nuevo en Android Studio.
* Agregar al archivo Gradle las dependencias para poder usar Google Maps API, a la fecha las dependencias son:
  + com.google.android.gms:play-services-maps:10.2.4
  + com.google.maps.android:android-maps-utils:0.5
* Obtener la clave de Google Maps API:
  + Copiar el enlace proporcionado en el archivo google\_maps\_api.xml y pégalo en el navegador. Este accederá a la **Google API Console** y dará la información requerida sobre el proyecto a través de los parámetros de la dirección URL.
  + Seguir las indicaciones para crear un nuevo proyecto en la **Google API Console** o selecciona un proyecto previamente existente.
  + Crear una clave de la API de Google Maps SIN NINGUNA restricción para el proyecto.
  + Copiar la clave de la API de Google Maps obtenida, y pega la clave en el elemento <string> del archivo google\_maps\_api.xml en el proyecto de Android Studio.
* En el proyecto crear una clase nueva llama GoogleMapOfflineTileProvider que implemente la interfaz com.google.android.gms.maps.model.TileProvider
* En la clase GoogleMapOfflineTileProvider sobrescribir el método public Tile getTile(int x, int y, int z) e implementar un sistema de descarga de imágenes de Tile correspondiente a los valores de X, Y, Z. Esta implantación deberá considerar que, si el Tile está en cache, no deberá utilizar internet.
* Crear una clase con el nombre que desee que implemente la clase OnMapReadyCallback y con un atributo de tipo GoogleMap y nombre mMap.
* En la clase anterior sobrescribir el método onMapReady (GoogleMap googleMap) y guardar en mMap el parámetro recibido.
* A la variable mMap añadirle un el TileProvider creado en los puntos anteriores con la siguiente línea de código:

|  |
| --- |
| mMap.addTileOverlay(**new** TileOverlayOptions().tileProvider(**new** GoogleMapOfflineTileProvider(**this**)).zIndex(-100)) |

# RESULTADOS Y DISCUSIÓN

Dependencias añadidas al archivo **build.gradle** se pueden observar en la Fig. 4.



Fig. 4. Dependencias que se deben añadir al archivo gradle en Android Studio.

Creación de la clase que implementa la interfaz TileProvider y la sobre escritura del método getTile como se observa en la Fig. 5. En los parámetros de este método se recibe el Tile que se desea descargar con las coordenadas X, Y y un valor Z que representa el nivel de Zoom. Con esta información se debe acceder a un repositorio de Tiles que puede ser personalizado y así poder añadir diferentes capas de visualización o seguir utilizando los servicios de Google a través de su API de JavaScript con la URL:

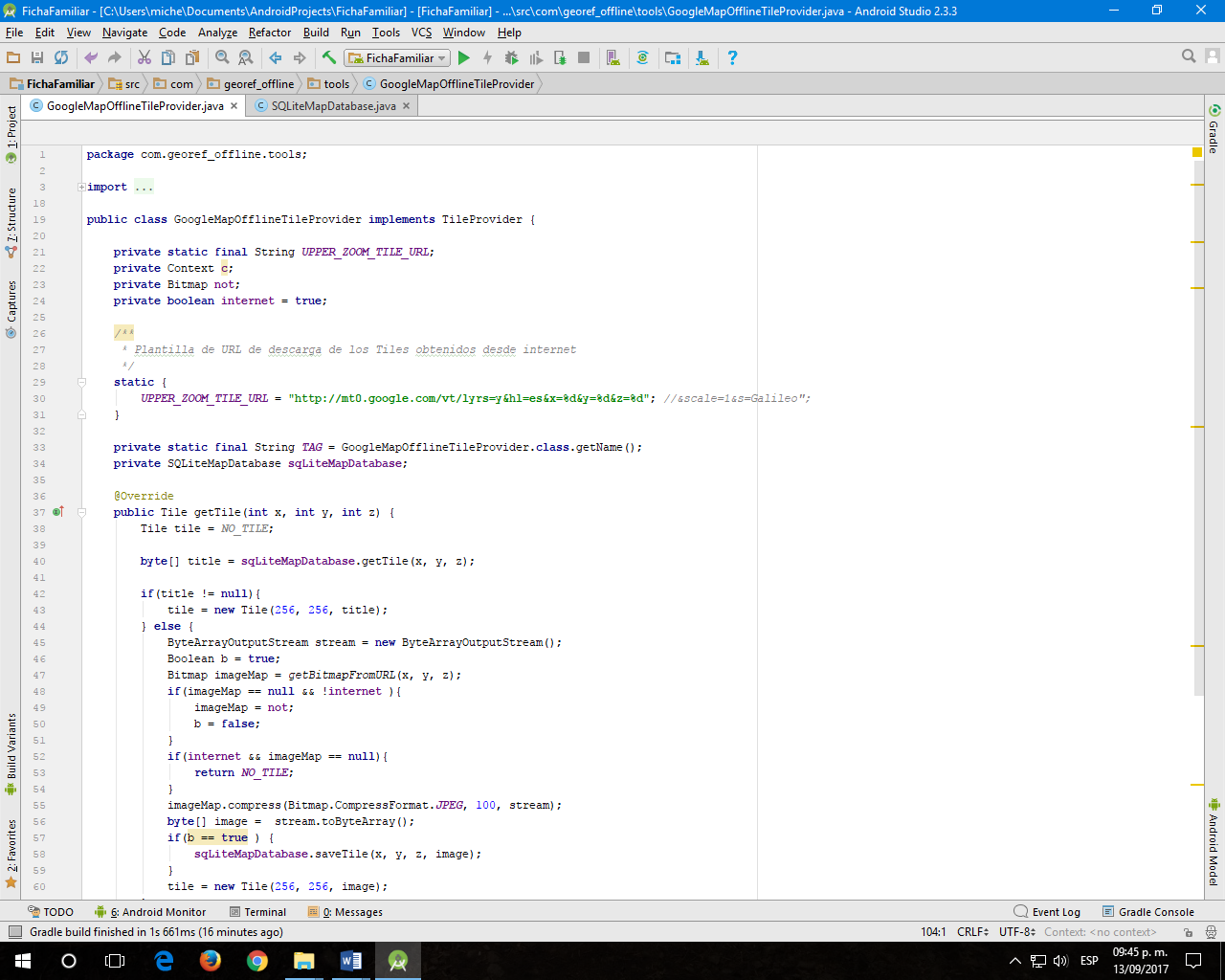
|  |
| --- |
| http://mt0.google.com/vt/lyrs=y&hl=es&x=%d&y=%d&z=%d |

donde %d en x=%d y=%d y z=%d debe ser cambiado por los valores X, Y y Z recibidos en el método getTile. Existen cuatro servidores de Google, cambiando en la URL el mt0, mt1, mt2 y mt3, del inicio.

En la implementación realizada se creó una clase llamada SQLiteMapDatabase que se encarga de almacenar en un base de datos, de una sola tabla con campo X, Y, Z y fecha de guardado de las imágenes descargadas. Así cuando se le solicite a getTile, un Tile especifico hará uso de esta clase para buscarla en la base de datos, de no ser capaz de encontrarla intentara descargar y almacenar el Tile buscado.

SQLiteMapDatabase se asegura de no violar los acuerdos de licencia con Google ya que no permite al usuario guardar permanentemente las imágenes de mapa, manteniéndolas en cache un máximo de 30 días.

Para poder utilizar una imagen con Tile es necesario convertir el archivo descargado a un arreglo de Bytes, ya que la clase Tile provista por Google trabaja con ese tipo de dato.



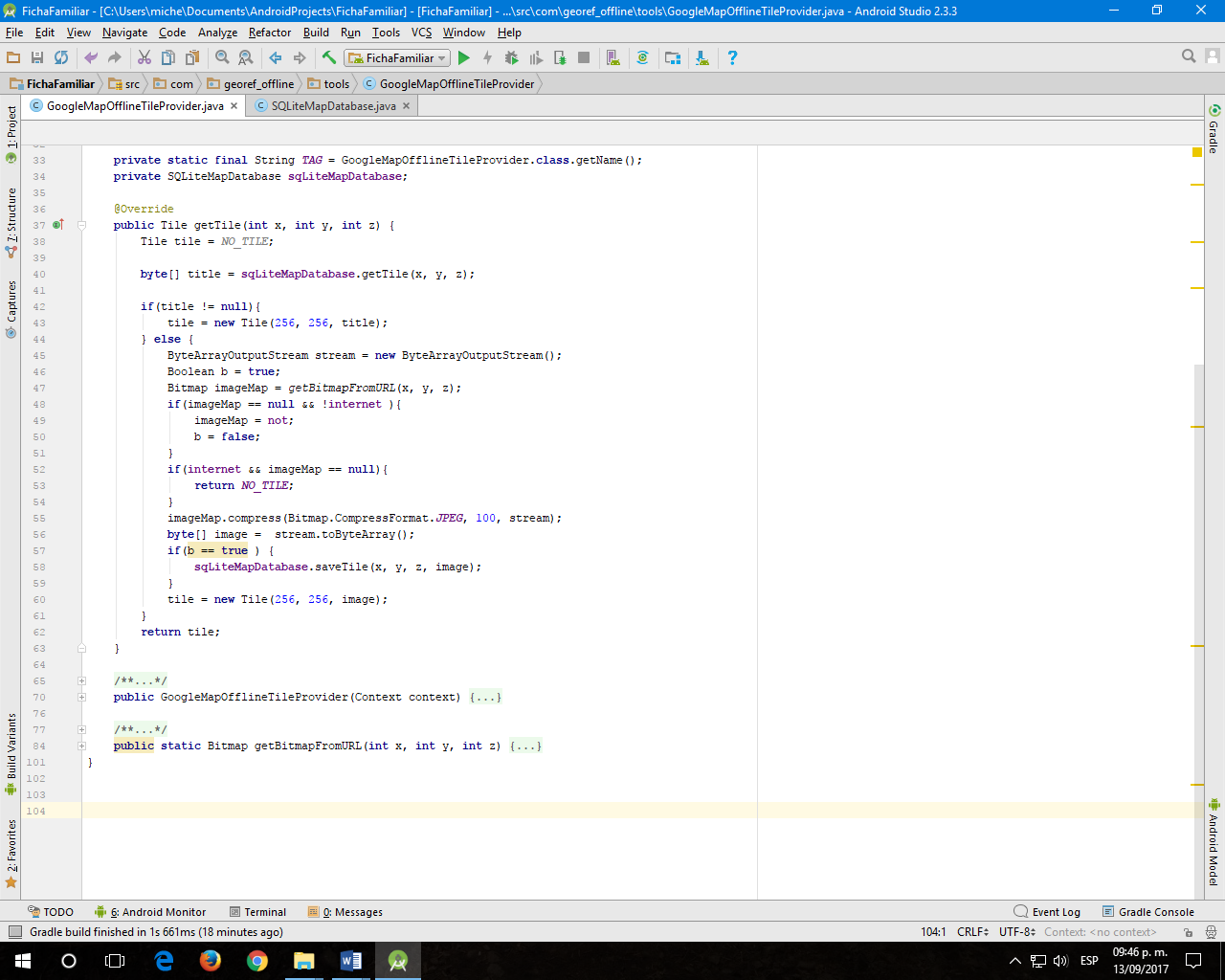
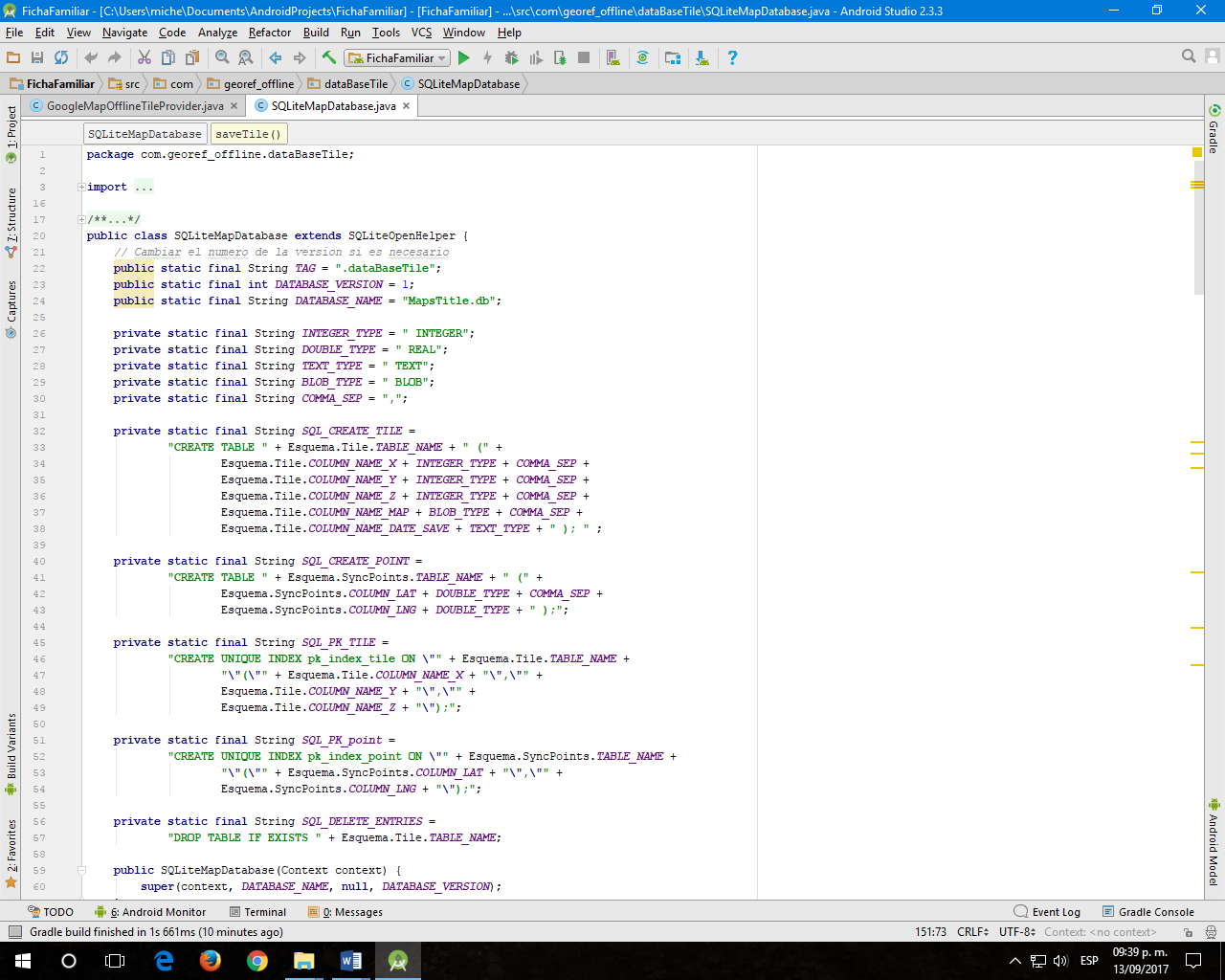


Fig. 5. Código fuente de la clase GoogleMapsOfflineTilePrivider en Android

La implantación de SQLiteMapDatabase es mostrada en la Fig. 6, hace uso de **SQLite** extendiendo la clase SQLiteOpenHelper de Android. Se sobrescribe los métodos onCreate, onUpgrade, para permitir la creación de la base de datos al vuelo, cuando sea necesario. Además, se añade el método getTile (ver Fig. 7) que busca en la base de datos un Tile especifico y de encontrarlo devuelve un arreglo de Bytes que contiene la imagen del mosaico (Tile), además se encarga de validar que esta imagen solo pueda ser usada si tiene un tiempo máximo de vida en el dispositivo de 30 días; para poder lograrlo hace uso de la fecha en que se guarda y la fecha actual del dispositivo.



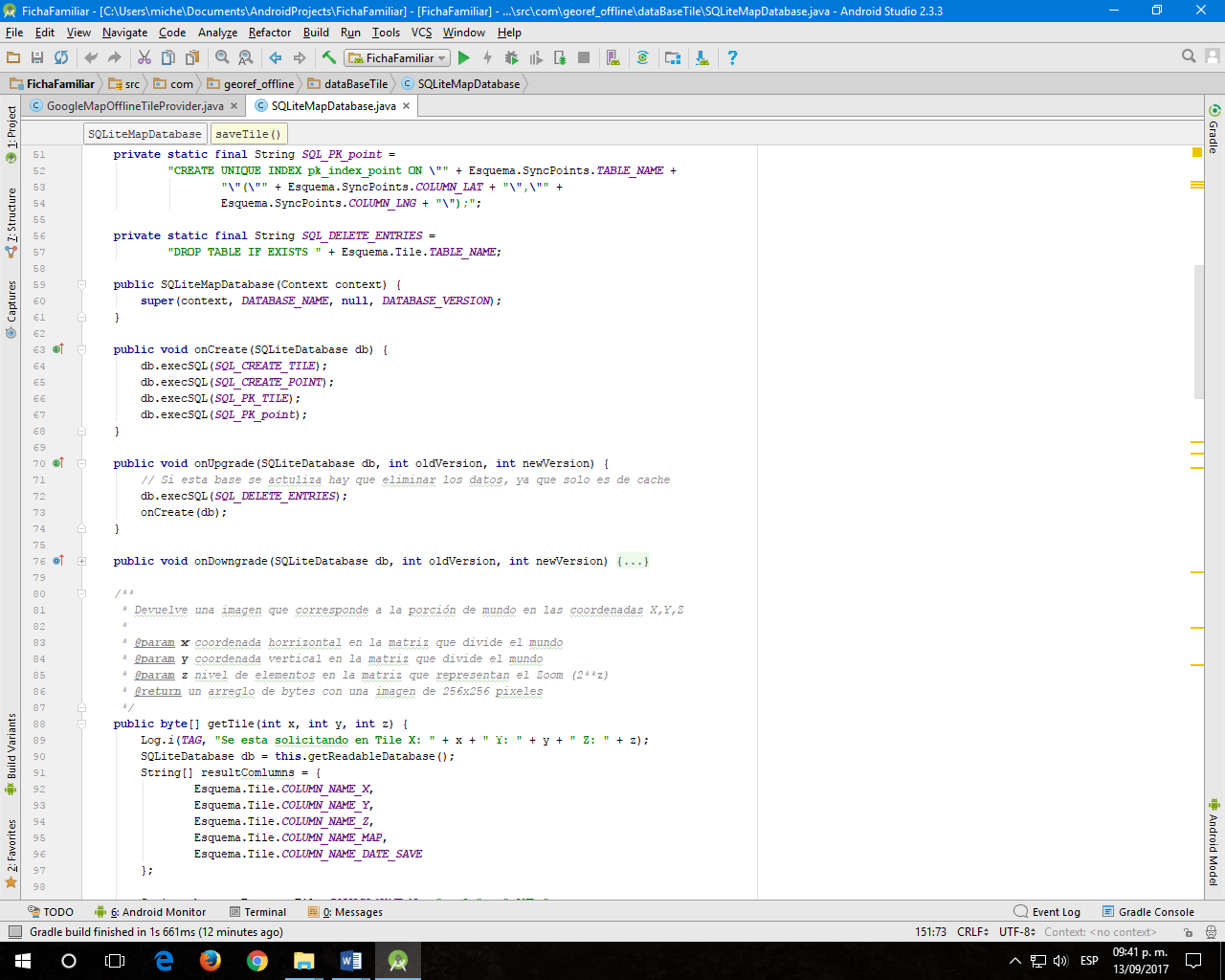
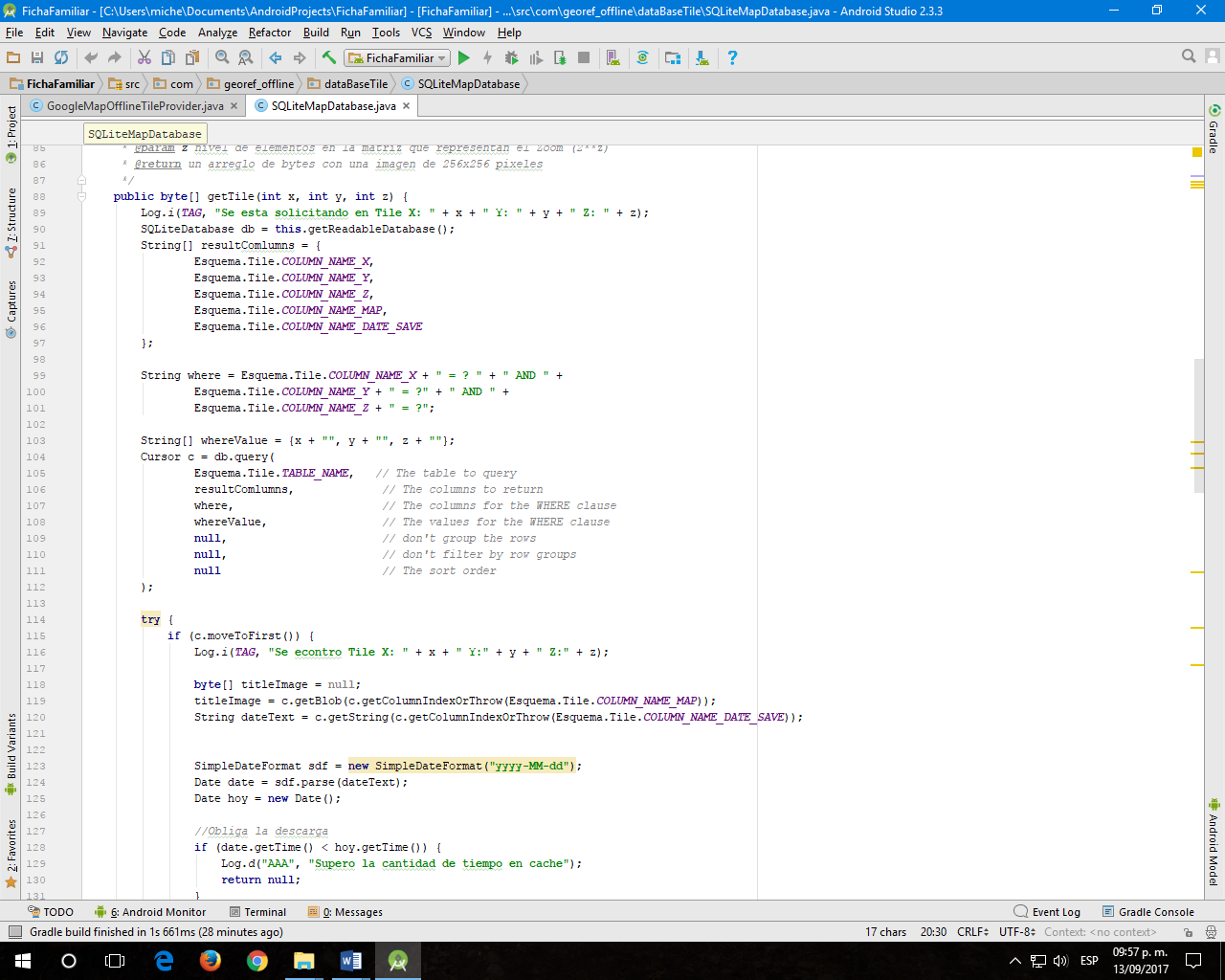
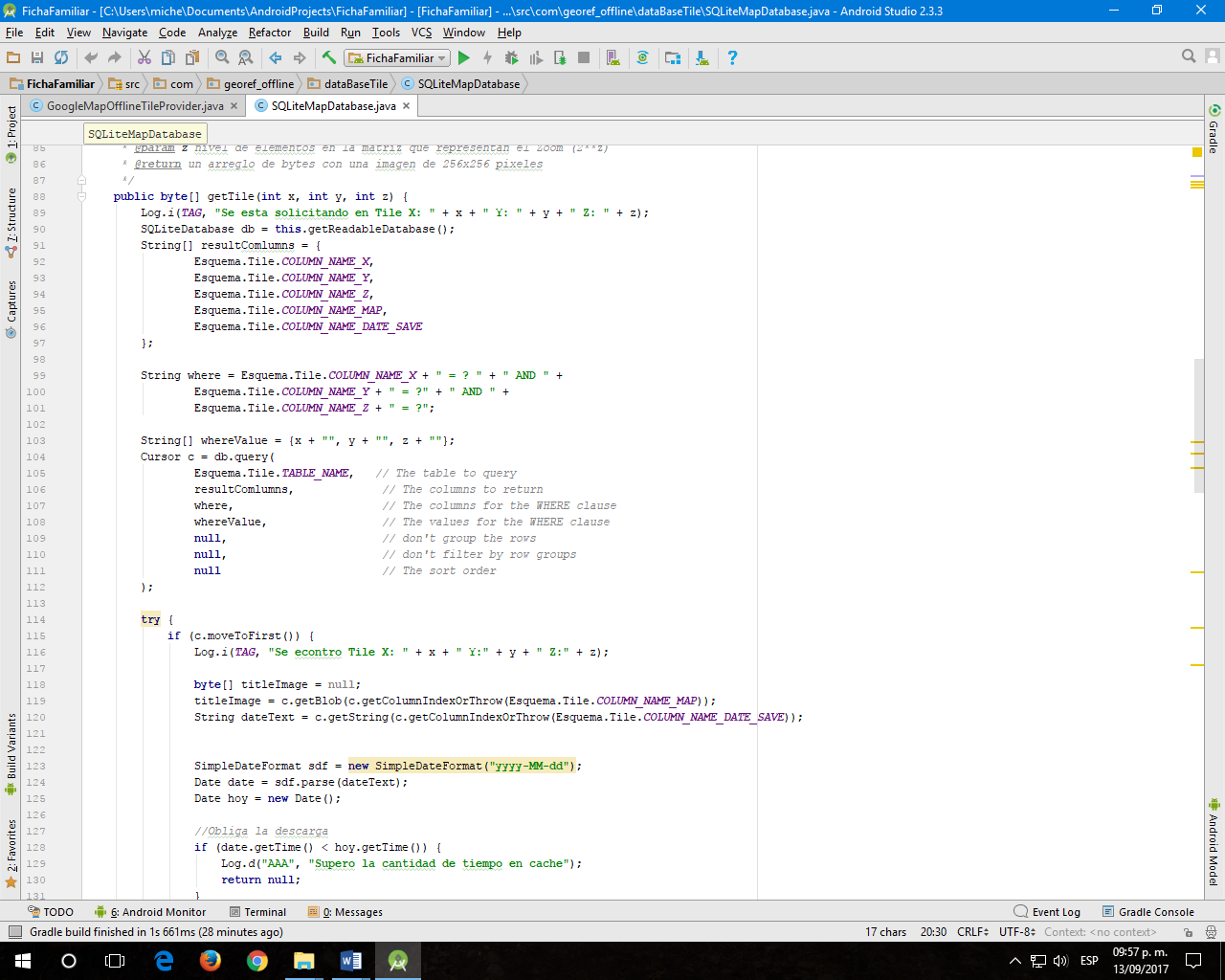


Fig. 6. Código para la creación de base de datos y métodos onCreate y onUpgrade de la clase SQLiteMapDatabase





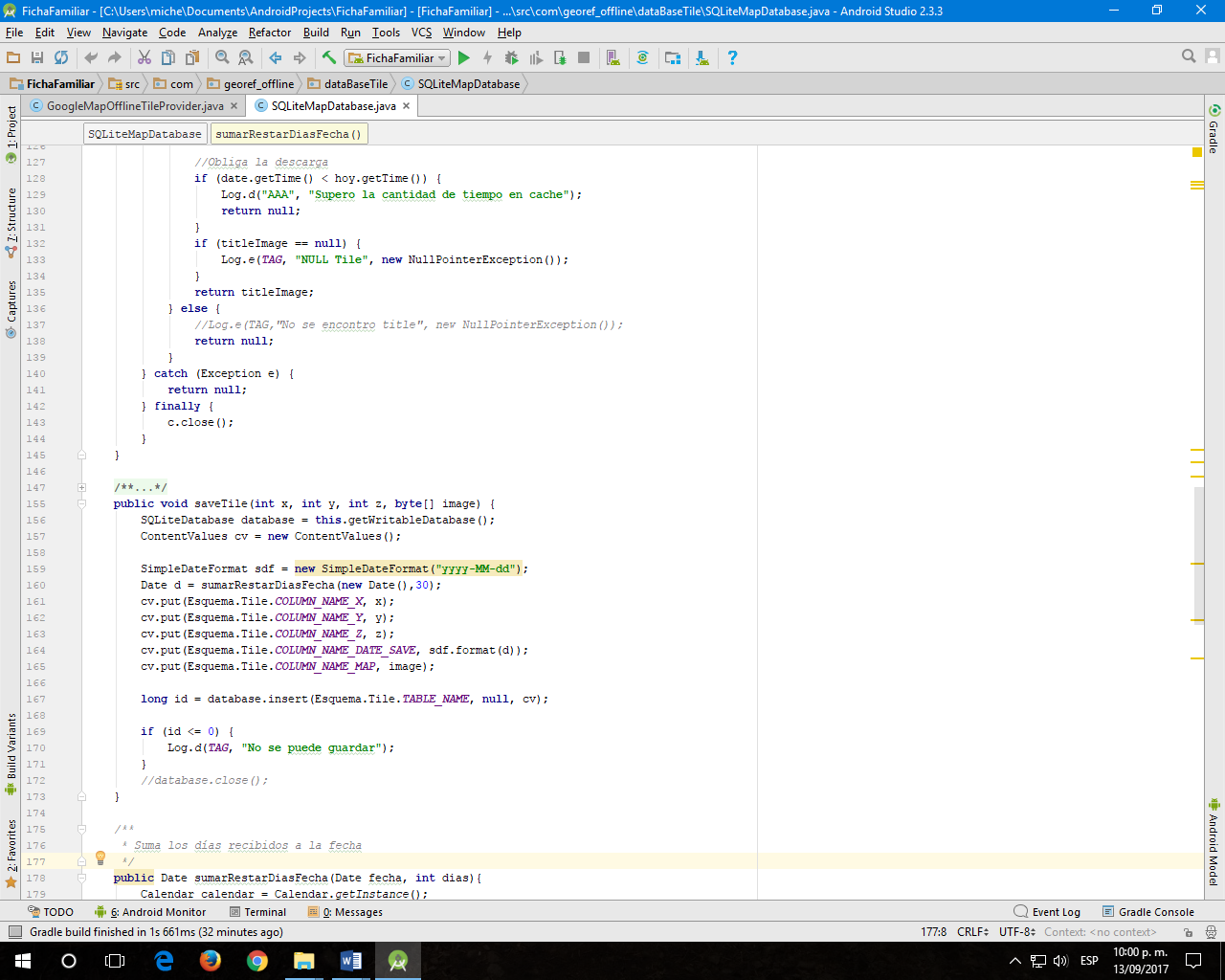


Fig. 7. Métodos getTile y saveTile de la clase SQLiteMapDatabase

Una vez implementada creada la clase GoogleMapOfflineTile Provider y su auxiliar SQLiteMapDatabase hacer uso de ella es realmente sencillo, en alguna implementación de Google Maps, hay que indicarle al objeto de tipo GoogleMap por el cual podemos cambiar el comportamiento visual del mapa, como añadir icono, cambiar la cámara y por supuesto sobreponer imágenes de mosaico (Tile). Se desactiva la obtención de imágenes de mapa original con la línea de código:

|  |
| --- |
| **mMap**.setMapType(GoogleMap.MAP\_TYPE\_NONE); |

Para poder utilizar solamente las imágenes que se obtiene con GoogleMapOffilineTileProvider, basta con agregar al objeto de tipo GoogleMap un TileProvider con la línea de código:

|  |
| --- |
| **mMap**.addTileOverlay(new TileOverlayOptions().tileProvider( new GoogleMapOfflineTileProvider(this)).zIndex(-100)).clearTile  Cache(); |

Una vez realizada la implementación se mostrará el mapa, utilizando los Tiles almacenados y de forma offline, ver Fig. 8.

Lograr esto es realmente sencillo gracias a que el diseño de Google Maps API no es engorroso, pero si no se tiene ningún conocimiento de este, es necesario leer la documentación oficial para entender un uso básico de sus servicios.

Esta idea es lo suficientemente versátil para poder aplicarse a diferentes aplicativos, especialmente en los que por su naturaleza pasaran periodos sin conexión a internet, como aplicación para recolectar información de personas en área rurales donde extrañamente se cuenta con algún tipo de conexión inalámbrica.

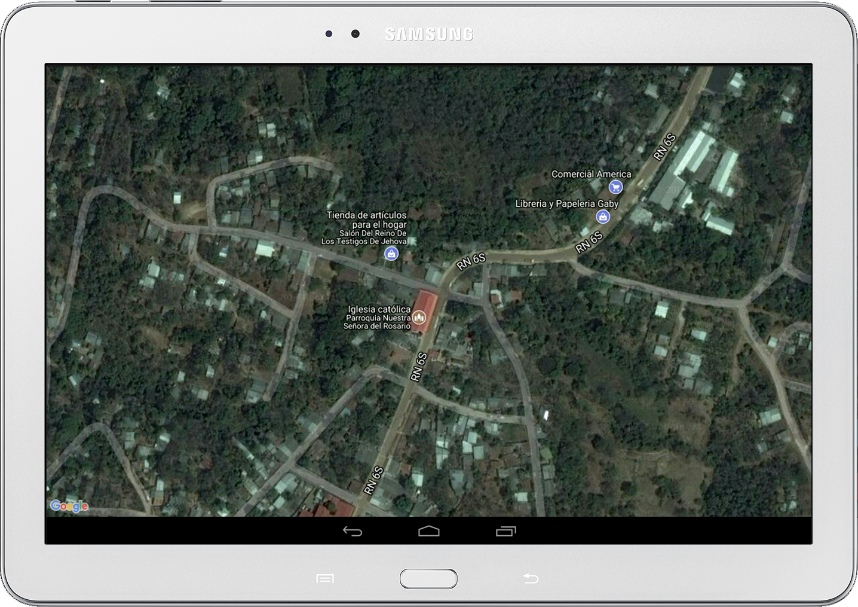


Fig. 8. Mapa de Google Maps, utilizando un TileProvider descrito

# CONCLUSIONES

* Google Maps provee un servicio de mapas muy versátil y potente, utilizar un TileProvider para permitir su uso durante periodos sin conexión brinda una experiencia de usuario superior.
* Poder usar mosaicos sobre los mapas existentes nos permite crear nuestros propios repositorios para poder aportar un mejor nivel de zoom, con elemento con más detalles y personalizados

# REFERENCIAS

1. (2017) Documentación API de Google Maps [online]. Available: https://developers.google.com/maps/documentation/android-api/map?hl=es-419
2. BOLFOR; ETSFOR. (1999). Cartografía y Uso de la Tecnología GPS. Santa Cruz, Bolivia: El País.
3. Buzai, G. D. (2013). Sistema de Información Geográfica (SIG): Teoría y Aplicación. Buenos Aires, Luján, Argentina: Universidad Nacional de Luján.
4. (2010) Información oficial del Gobierno de los Estados Unidos relativa al Sistema de Posicionamiento Global [online]. Available: http://www.gps.gov/spanish.php