**Juan Camilo Restrepo Velez 000373886**

Tabla de contenido

[1) EU Profesores Prototipo 2](#_Toc40630967)

[1.1) MainActivity 2](#_Toc40630968)

[1.2) SesionActivity 4](#_Toc40630969)

[1.3) NewSesionActivity 7](#_Toc40630970)

[1.4) ActividadActivity 12](#_Toc40630971)

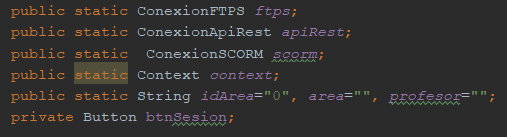
[Referencias 14](#_Toc40630972)

# 1) EU Profesores Prototipo

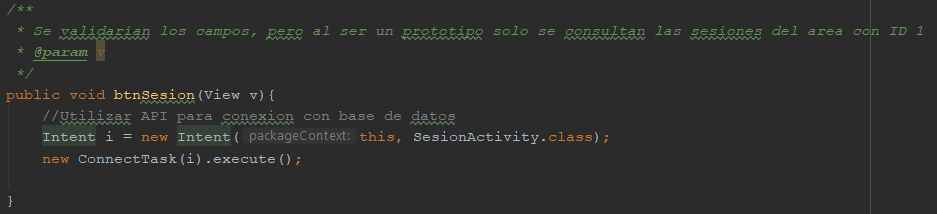
Esta aplicación es un prototipo de la app Profesores en la cual se centrará en crear sesiones sobre una misma área y siempre se crearán tres actividades, cada una con tres recursos (una imagen, un documento, un apk) en código estático (“quemado”).

## 1.1) MainActivity

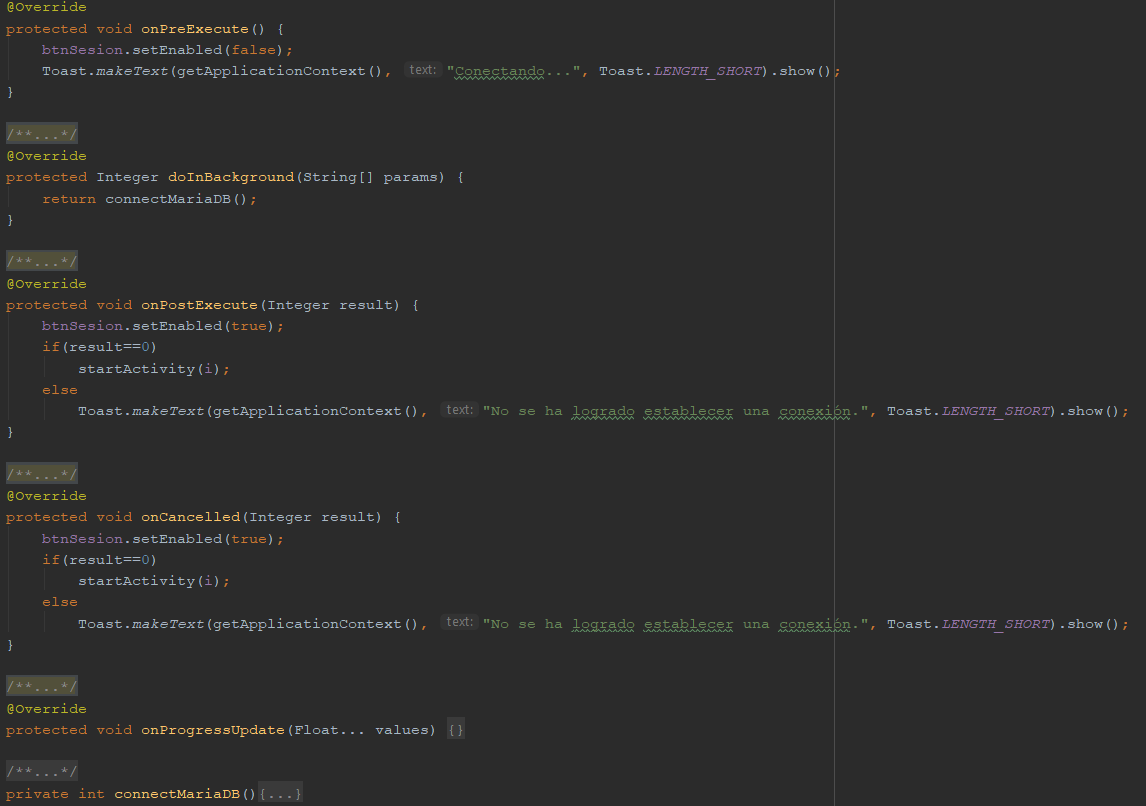
Es la ventana principal donde se debería validar el LogIn del profesor, pero al ser un prototipo simplemente se entrará con el filtro de que el ID del Area es igual a 1, por lo que en esta clase se definen los atributos que se requerirán en las demás clases:



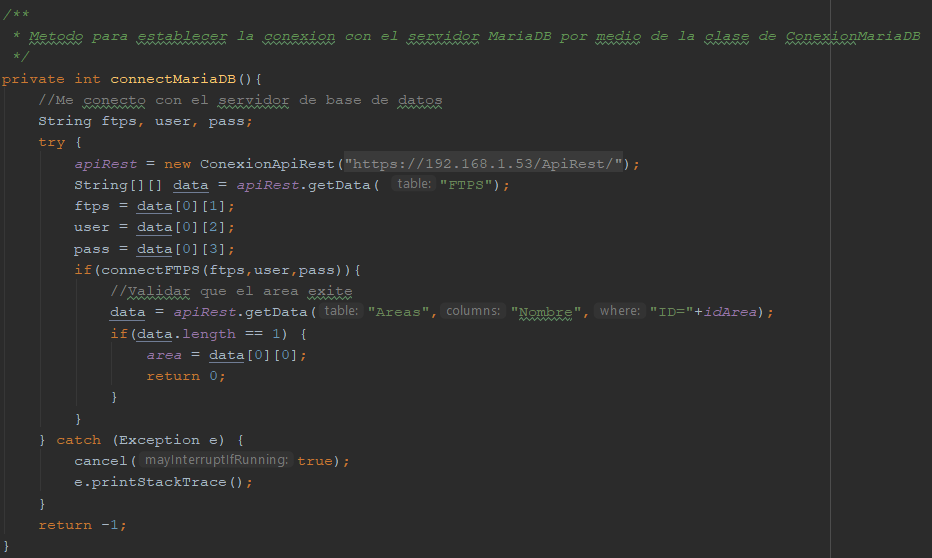
En el onCreate se inicializar el IdArea en 1, el profesor en oscar y el context. Luego está el botón para iniciar sesión:



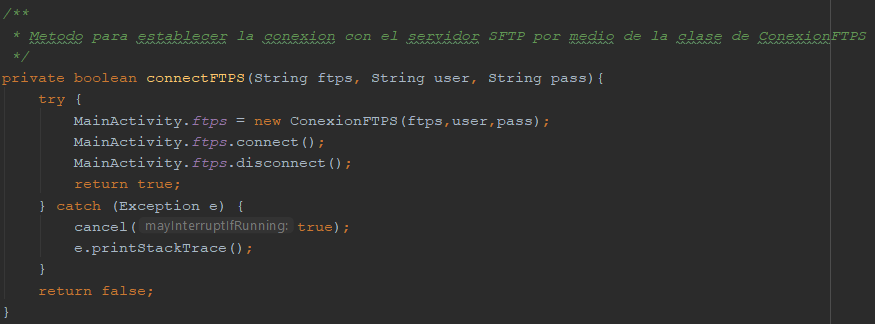
Se creará un hilo (tarea asincrónica) que probará la conexión con MariaDB por medio del API REST:



Si el resultado de la conexión fue 0 quiere decir exitoso, si es -1 quiere decir que no se pudo establecer la conexión con el API REST o el servidor FTP

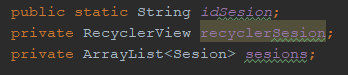


Y con los datos del servidor FTP obtenidos probará la conexión con el servidor FTP:

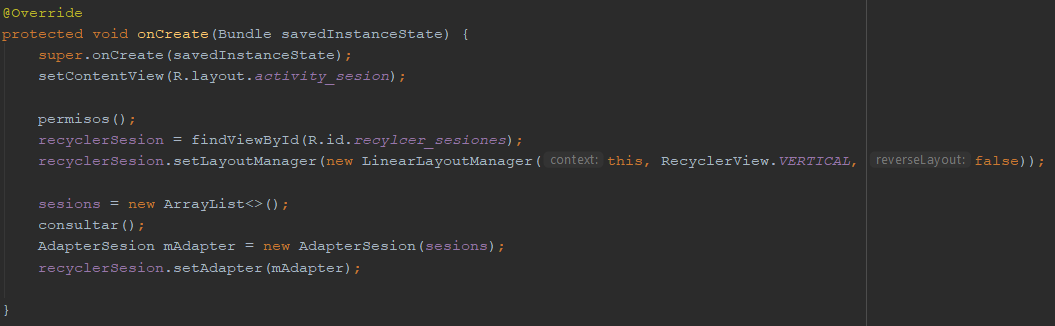


## 1.2) SesionActivity

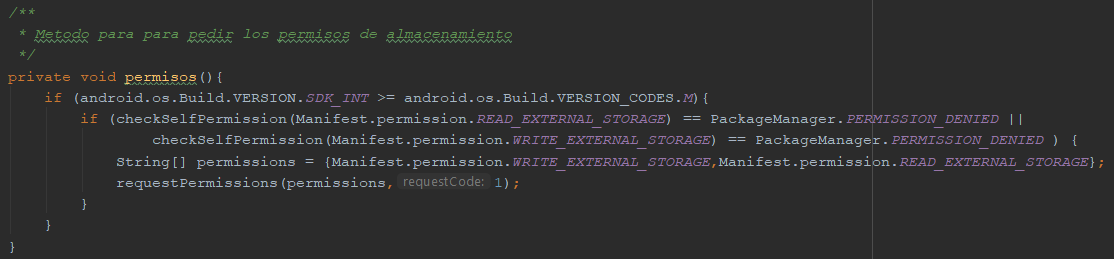
Esta clase estará encargada del manejo lógico para visualizar las sesiones que están creadas en el área cuyo ID es igual a 1, además de dar la opción de crear una sesión y de ver las actividades de una sesión especifica. Para ello primero se crean los atributos necesarios como es el RecyclerView para mostrar la lista de sesiones:



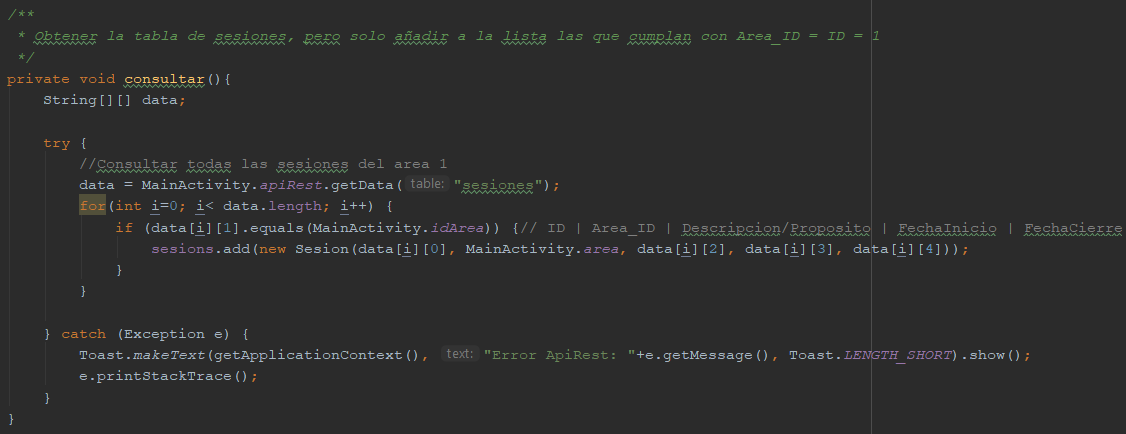
Luego en el onCreate se inicializan las variables y se piden los permisos de almacenamiento, además de consultar todas las sesiones que pertenecen al área con ID 1:



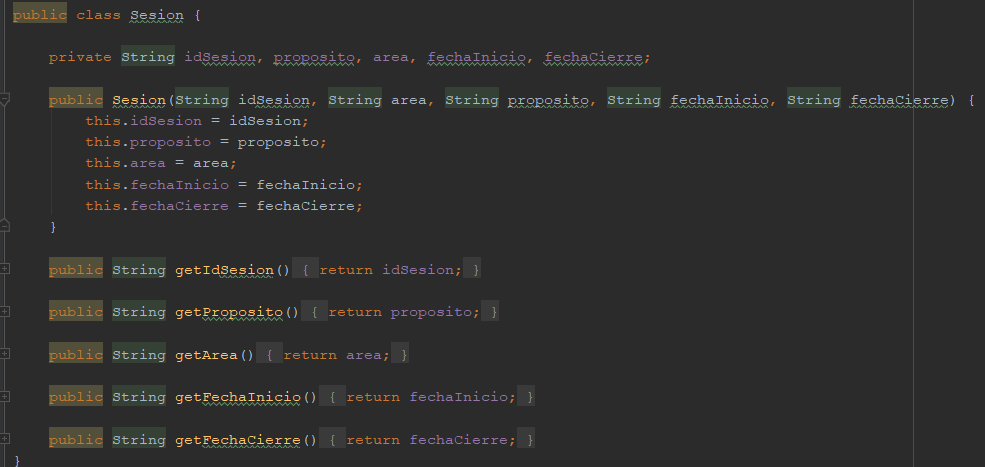
El código de los permisos es exactamente el mismo:



Y para consultar, se debe obtener toda la tabla de sesiones, pero solo añadir a la lista las que tengan la FK (la llave foránea) Area\_ID igual al ID del MainActivity, es decir, que sean igual a 1:



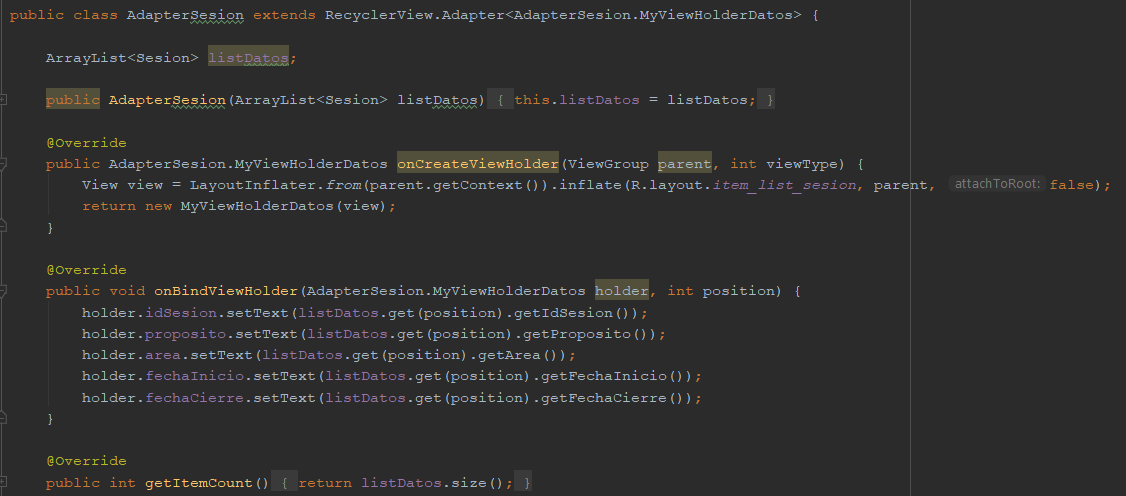
Cabe resaltar que, para esto, la lista donde se agregan los resultados es de tipo Sesion que tiene un atributo para cada columna:



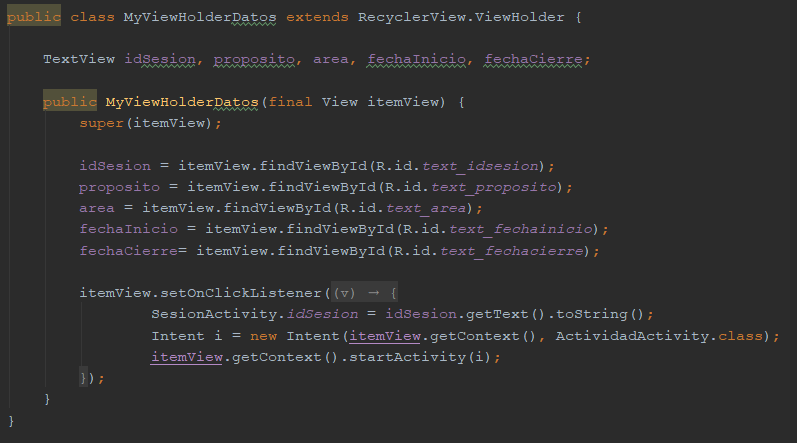
Que son los mismo que se mostrarán en el CardView ya que no se implementará ningún tipo de botón en la tarjeta:



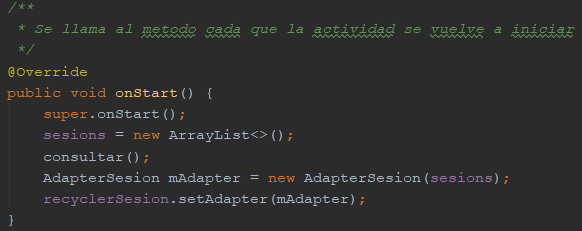
Que para poder mostrar tantas veces las tarjetas como sesiones existentes se crea la clase *AdapterSesion*que hará esto posible:



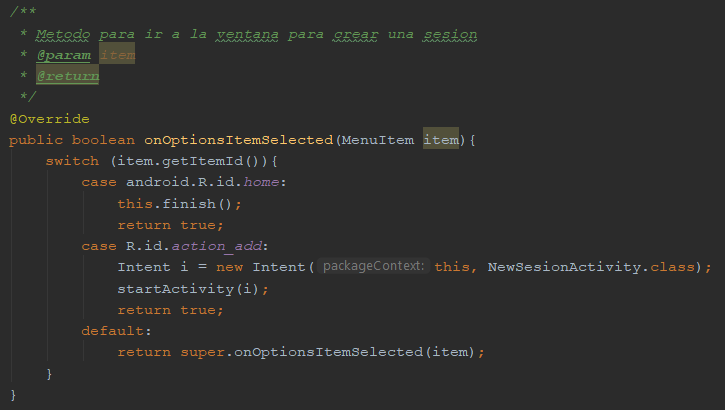
Y es en la clase *MyViewHolderDatos* donde se debe programar el código para cuando se dé clic sobre una tarjeta ir a la ventana con todas las actividades que tiene la sesión seleccionada:



Además, está el método onStart() que se llama cuando la actividad vuelve a iniciar por lo que vuelve a consultar las sesiones por si hubo algún cambio.

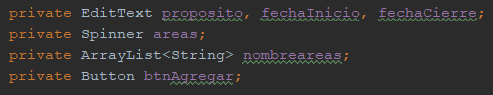


Por último, cuando se dé clic en el boton “+” de la venta de sesión se abrirá una ventana para poner los datos necesarios para crear una sesión y sus actividades con sus recursos por debajo:

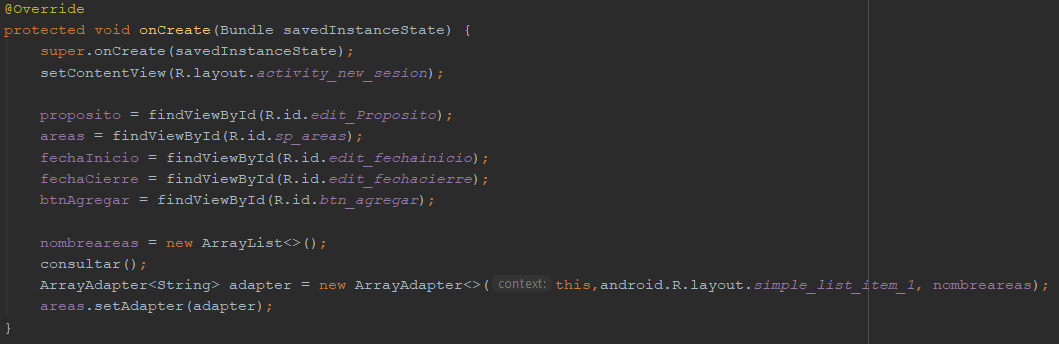


## 1.3) NewSesionActivity

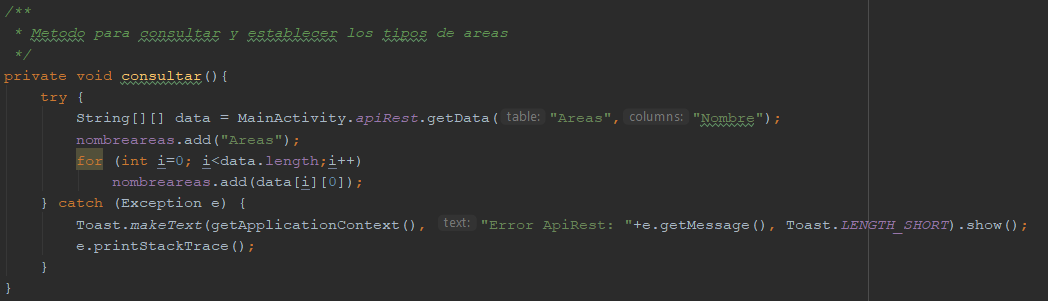
En esta clase se corresponde a crear una nueva sesión en el Area\_ID 1 y que esta contenga 3 actividades que su vez, cada una contiene 3 recursos, entonces estos son todas las variables globales a utilizar:



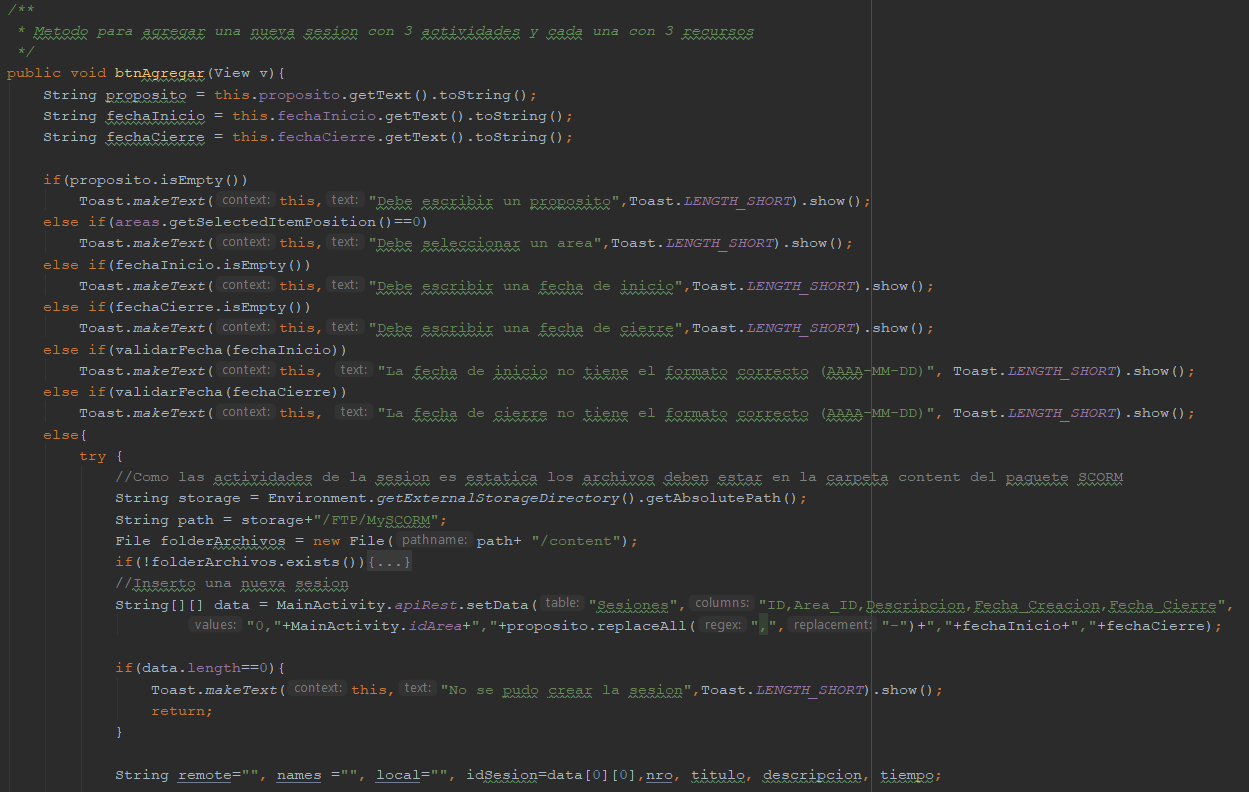
En el onCreate se inicializan las variables y se consultan las áreas para mostrar en el Spinner:



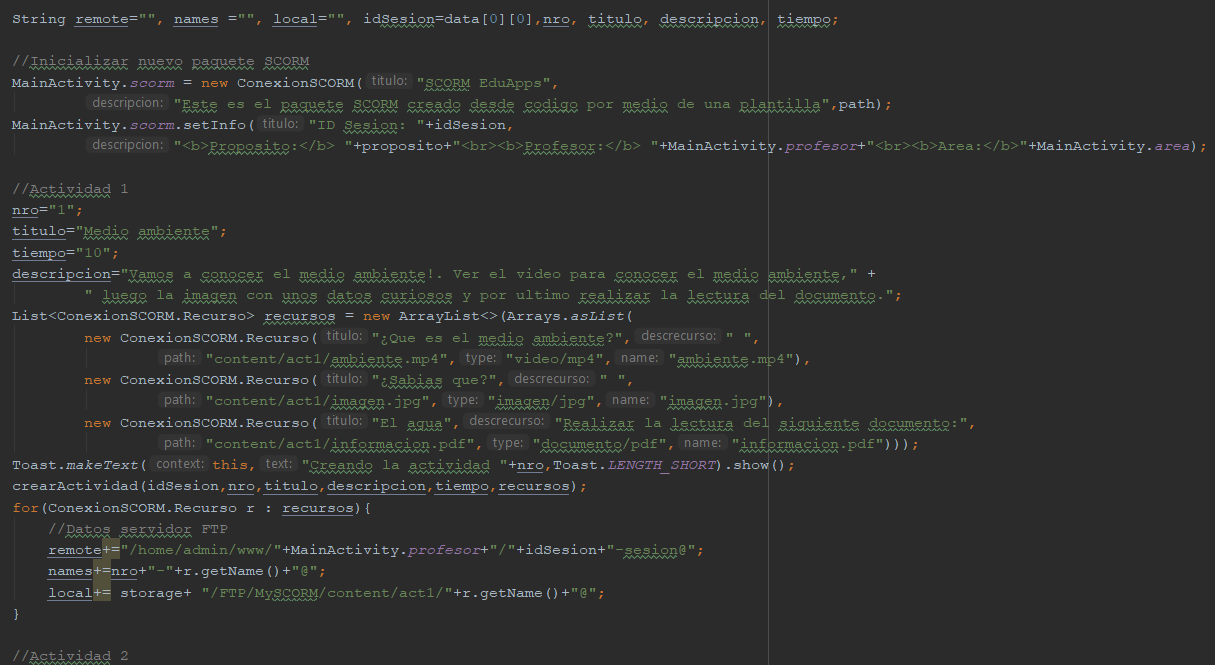
El método de consultar es el siguiente:

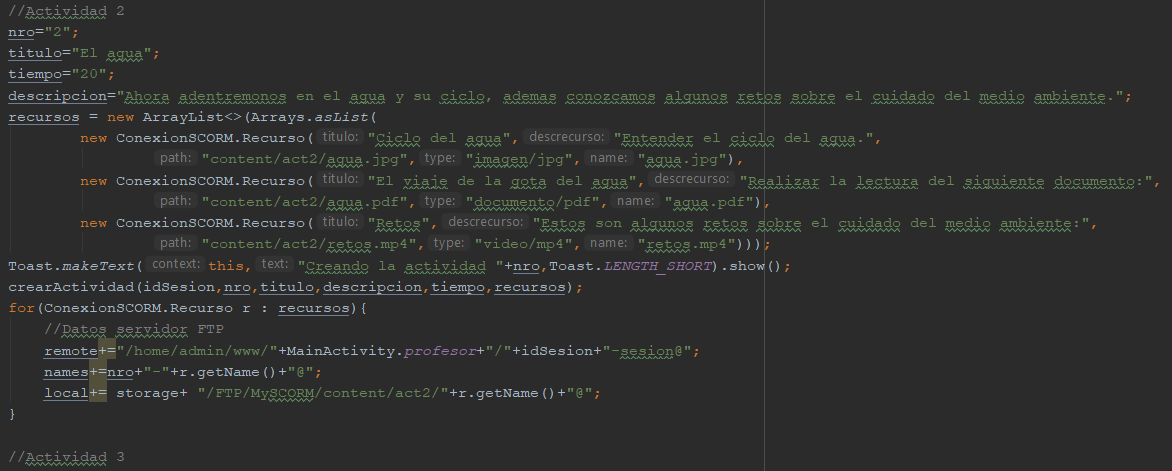


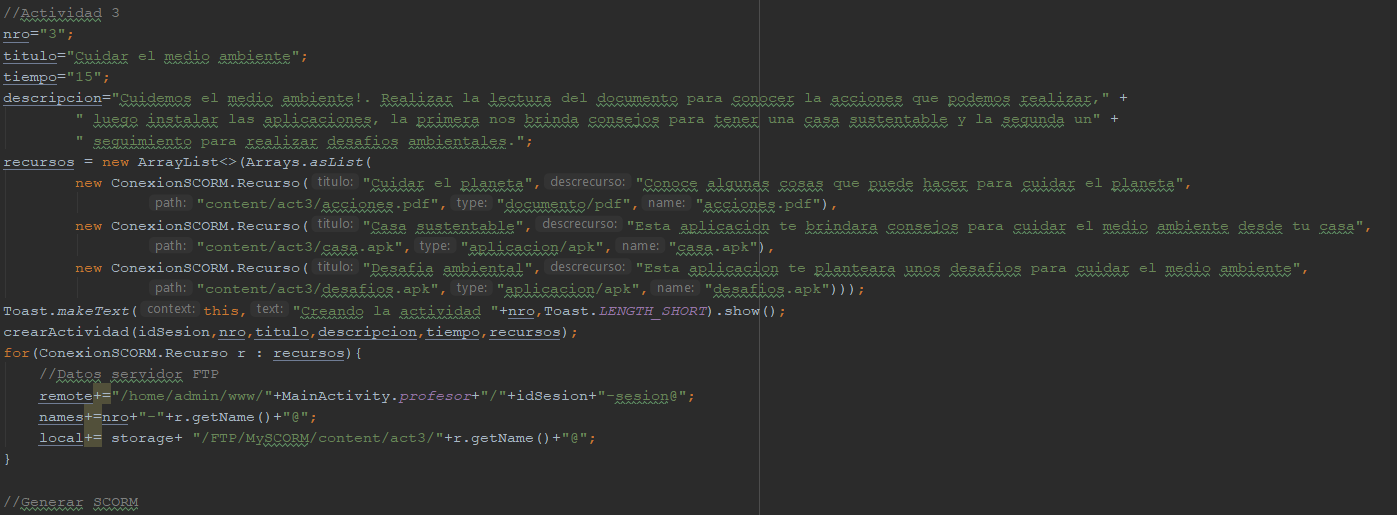
Y cuando se hace clic en el botón agregar se validan los campos y crea la sesión:



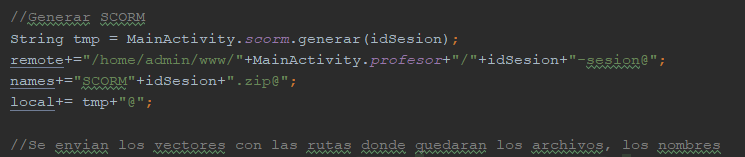
Luego de crear la sesión, se obtiene el ID con la cual quedo agregado y se establecen las variables necesarias para crear el registro de las actividades y para la creación del paquete SCORM, por lo que la creación de las actividades se hará por medio de una función enviando los datos de esta:



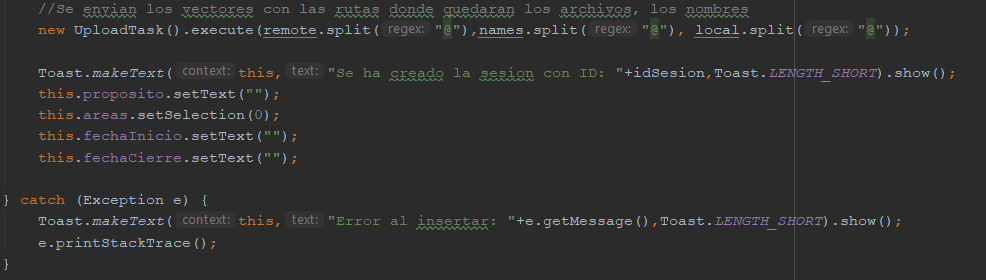




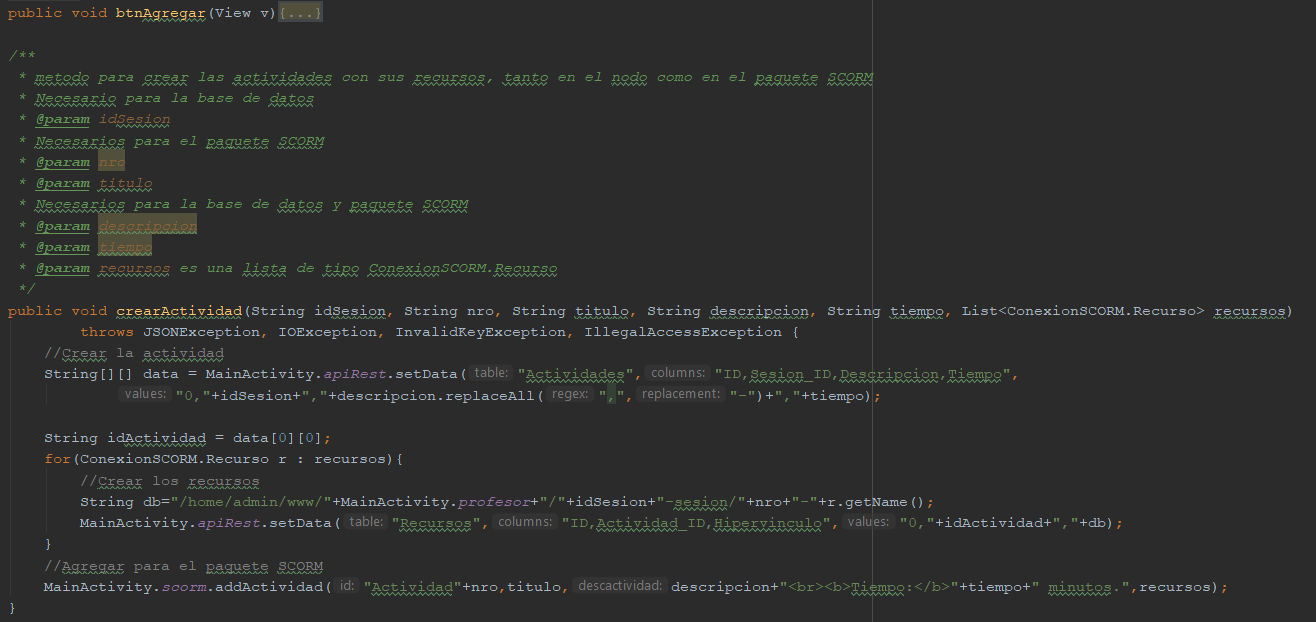
Luego de crear todas las actividades se procede a generar el paquete SCORM



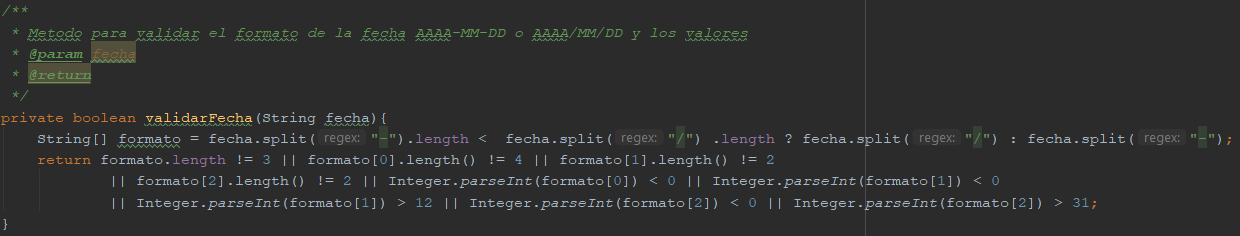
Por último, se suben todos los archivos y el paquete SCORM que un archivo .zip



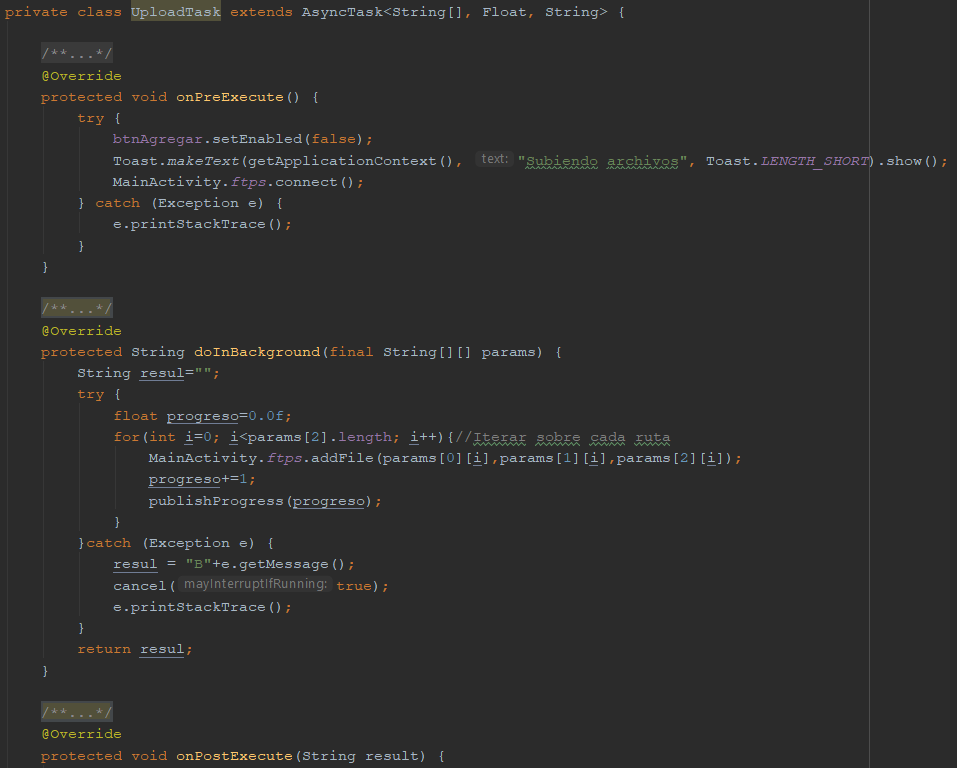
Por otro lado, la función para crear las actividades es la siguiente:

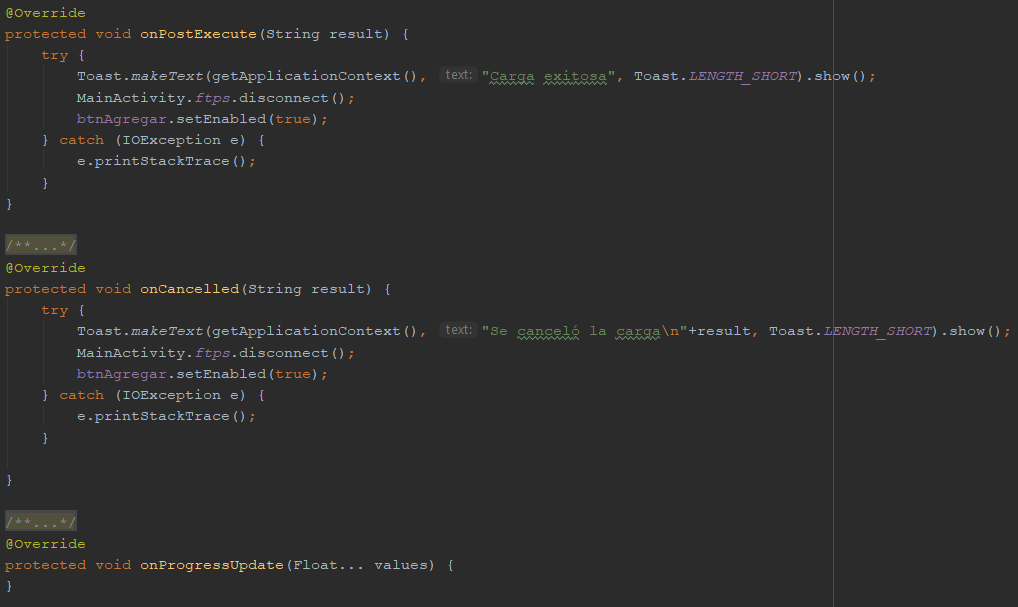


El método de validar fecha es el mismo utilizado en la app Estudiante:



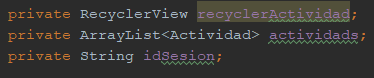
Y la clase encargada de subir todos los archivos de la sesión:



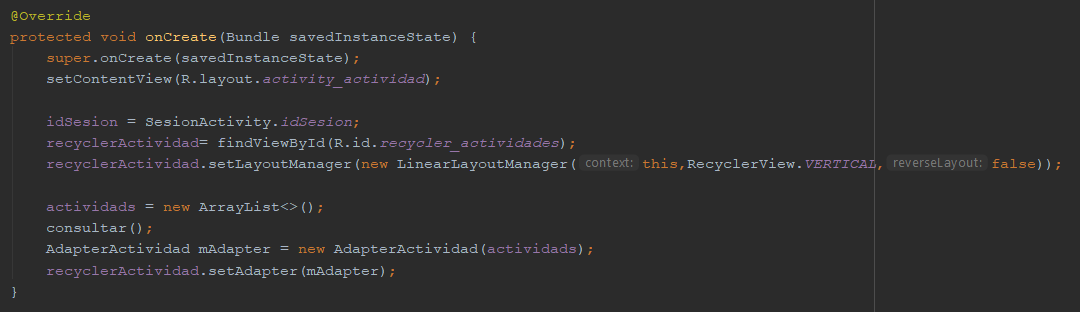


## 1.4) ActividadActivity

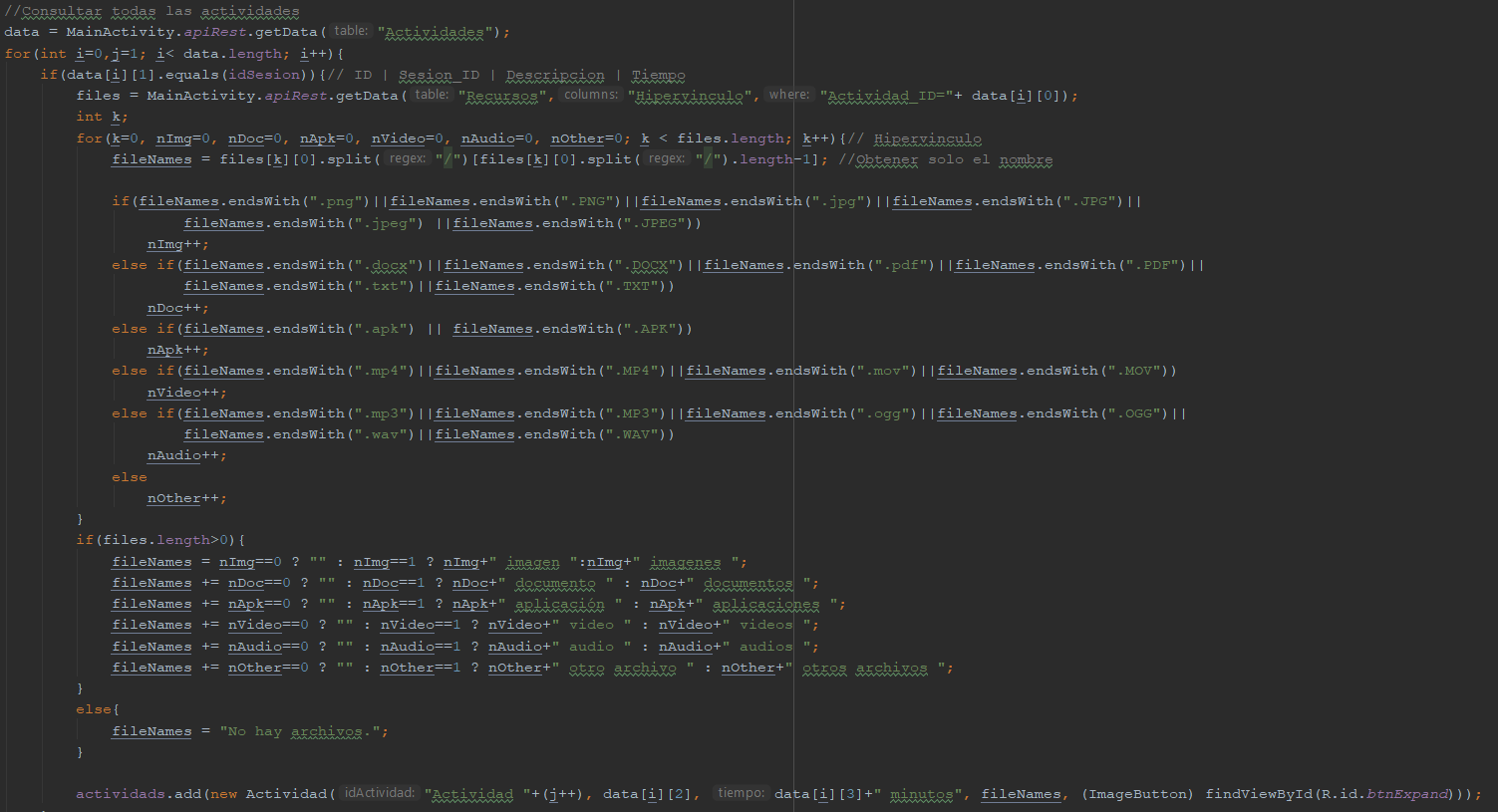
Se encarga de mostrar la lista de actividades que tiene una sesión en específico, solo se muestra la información sin botones, entonces en primer lugar se crean las variables necesarias como las listas de actividades y el RecyclerView y el idSesion que se utilizará:



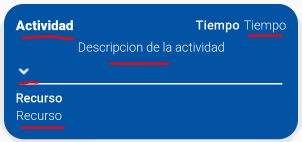
Luego en el onCreate() se hace las asignaciones e inicializaciones:



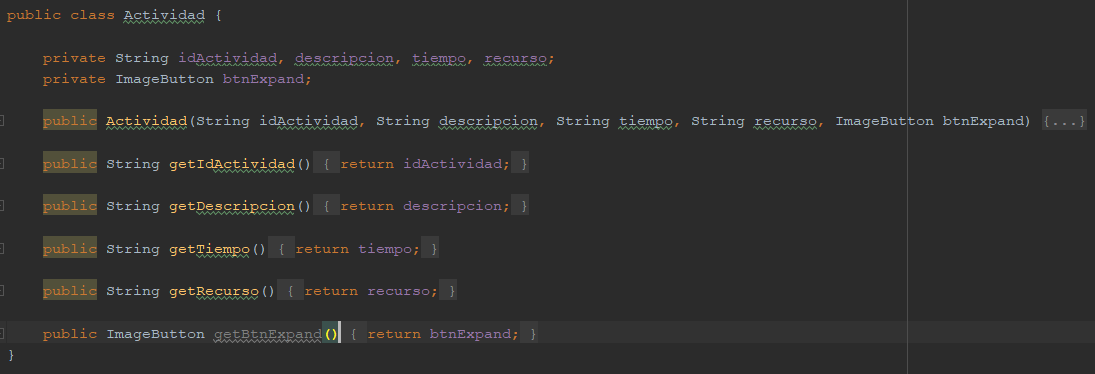
Como ya se ha hecho anteriormente, el método consultar se encargará de llenar la lista con los registros de la base de datos que cumplan con la condición de que el ID de sesión sea igual al idSesion:



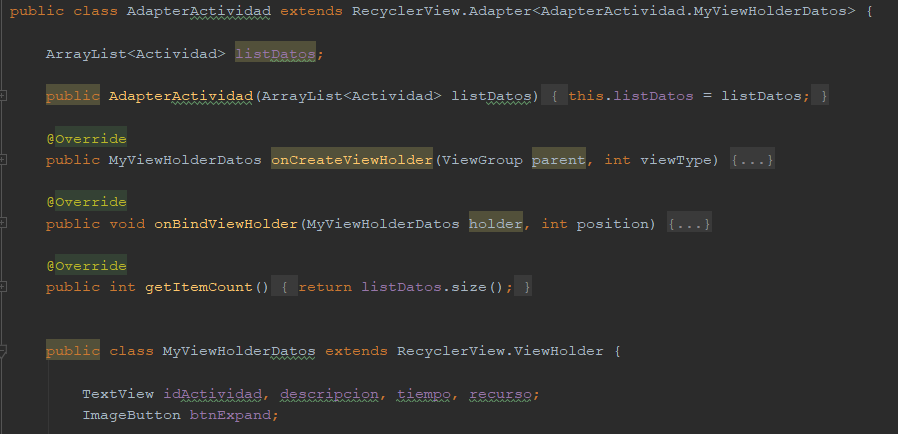
Solo se tiene un objeto demás en la lista que es el icono para expandir o colapsar la tarjeta para ver u ocultar, respectivamente, los recursos de la actividad, por lo que cumple que están todos los objetos de la CardView:



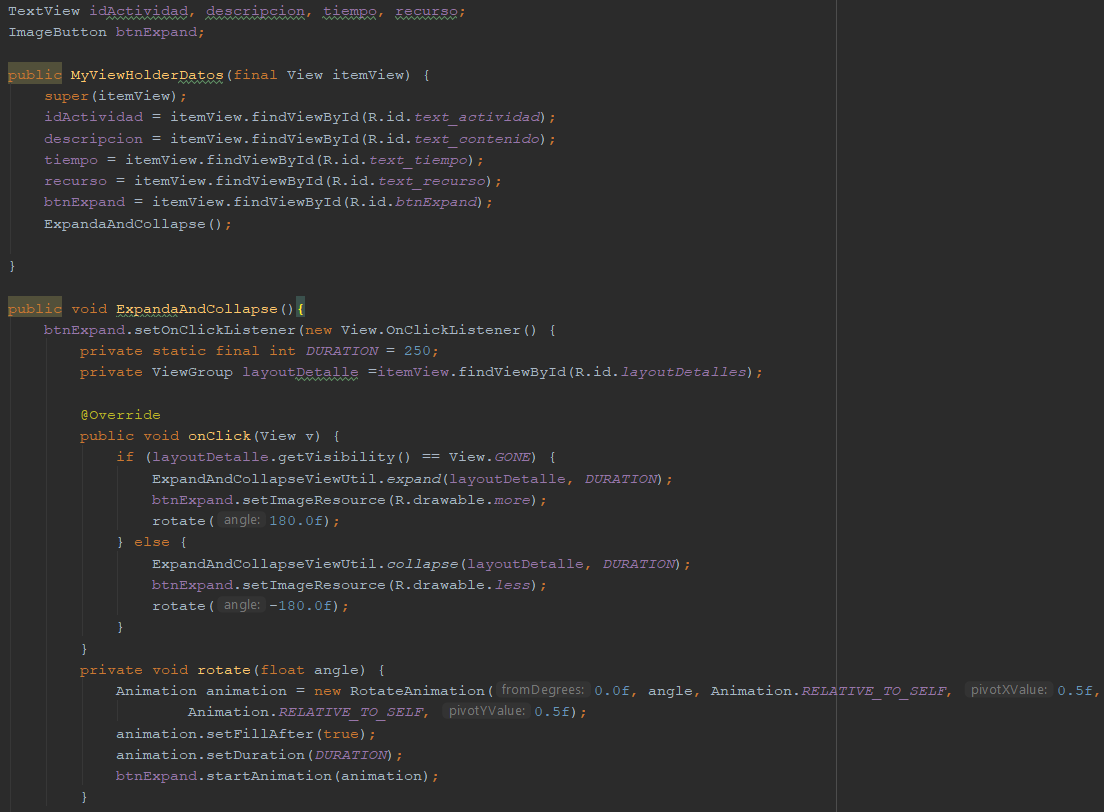
Y la clase que nos ayuda para crear una lista con todos estos objetos:



Por último, el *AdapterActividad* que nos replica la tarjeta en el RecyclerView:



Recordar que en la clase interna *MyViewHolderDatos* es donde se programa el código cuando se hace clic en la tarjeta o en algún objeto de esta:



## 1.5) NOTA

Ya que esta aplicación es un prototipo y se generarán las actividades y sus recursos de forma estática se deben cumplir las siguientes condiciones:

1. Se debe descarga la estructura SCORM que se encuentra en el proyecto de [GitHub](https://github.com/wilder3756/Ecosistema-EDUAPPS/tree/master/Paquete%20SCORM)
2. El contenido debe estar en una carpeta con el nombre *MySCORM* y debe estar ubicado en una carpeta llamada *FTP* en el almacenamiento del dispositivo
3. Como se crearán tres actividades con tres recursos, entonces deben estar en la carpeta *content* (dentro de la carpeta *MySCORM*) las tres subcarpetas que contienen los tres archivos de cada actividad

Para la reutilización de la plantilla y el código que genera el paquete SCORM se debe tener en cuenta:

* La estructura SCORM que se encuentra en el proyecto sirve para general un paquete SCORM 1.2
* Todos los archivos que se desean utilizar para que paquete deben estar ubicados en la carpeta *content* que está adentro de la estructura
* El código de *ConexionSCORM* crea los archivos: *index.html*, que es la página principal a mostrar y muestra el botón para abrir la página con todas las actividades, el *content.html*, archivo que muestra todas las actividades, y el archivo .zip, que comprime la carpeta que contiene todos los archivos SCORM por medio de la librería Zip4j versión 1.3.2 (para más información consultar su [sitio web](http://www.lingala.net/zip4j.html))
* El código de *prueba.java* es un ejemplo de cómo utilizar la clase *ConexionSCORM* para crear un paquete desde cero pasando solo la información que se muestra y la ruta de los archivos que de utilizarán.
* La plantilla que se usa para crear las actividades permite mostrarlas, hacerle su seguimiento y descarga los archivos. La plantilla fue tomada de [aquí](https://github.com/davidherney/tepuy), pero como la original eran tan densa (porque permite realizar muchos tipos de actividades) se adecuó a nuestra necesidad.

# Referencias

davidherney/tepuy. GitHub. (2020). Retrieved 15 May 2020, from <https://github.com/davidherney/tepuy>

JAVA, C. (2020). Comprimir una carpeta completa en Zip e ingresar el nombre de cual carpeta sera comprimida. JAVA. Stack Overflow en español. Retrieved 15 May 2020, from <https://es.stackoverflow.com/questions/75133/comprimir-una-carpeta-completa-en-zip-e-ingresar-el-nombre-de-cual-carpeta-sera>

Maven Repository: net.lingala.zip4j » zip4j » 1.3.2. Mvnrepository.com. (2020). Retrieved 15 May 2020, from <https://mvnrepository.com/artifact/net.lingala.zip4j/zip4j/1.3.2>