ANEXO I - MANUAL DEL PROGRAMADOR

*Aplicación Android para mejorar el rendimiento deportivo en piragüismo*

Trabajo de Fin de Grado

INGENIERÍA INFORMÁTICA

**Septiembre 2018**

**2017/2018**

**Autor:**

*Pablo Rubio Machacón*

**Tutores:**

*Guillermo González Talaván*

*Alberto Rodríguez Valle*

* Estructura del proyecto

El proyecto está diferenciado en dos grandes carpetas:

* A. Clases Java. Formada por estos paquetes.
  1. dataSesion
  2. etc
  3. main
  4. planningSession
  5. training
* B. Recursos layout XML

1. **Clases Java**
2. **dataSesion**

Conjunto de clases encargadas de la funcionalidad de analizar los entrenamientos de la aplicación.

* 1. **DataFolder**

Clase que presenta por pantalla una lista personalizada de todas las carpetas con todos los entrenamientos que hayamos realizado con su fecha como nombre.

Contiene las siguientes funciones:

-onCreate(): Inicializa y presenta la lista de carpetas.

-onRestart(): Actualiza esta lista cuando se borra alguna carpeta.

* 1. **DataTraining**

Clase que presenta por pantalla una lista personalizada de todos los entrenamientos existentes en una carpeta específica, es decir, que se hayan realizado el mismo día.

Contiene las siguientes funciones:

-onCreate(): Inicializa y representa la lista de entrenamientos.

-onCreateOptionsMenu(Menu menú): Inicializa el menú toolbar.

-onOptionsItemsSelected(MenuItem ítem): Gestiona los diferentes elementos del menú toolbar.

-wifiDirect(): Crea la actividad WifiDirect cuando se selecciona la opción de compartir entrenamiento del menú toolbar.

-showChangeNameDialog(File,String,String): Muestra un cuadro de diálogo personalizado cuando se mantiene pulsado un entrenamiento para cambiar el nombre de este.

-alertDelete(): Muestra un diálogo de confirmación para borrar los entrenamientos del directorio cuando se selecciona esta opción en el menú toolbar.

-delete(): Borra todos los entrenamientos del directorio.

-onRestart(): Actualiza la lista de entrenamientos cuando se borran o renombran.

* 1. **DataStudy**

Clase que presenta por pantalla el nombre y fecha del entrenamiento a analizar, así como los diferentes gráficos que se pueden realizar con sus datos.

Contiene las siguientes funciones:

-onCreate(): Inicializa y presenta por pantalla los datos del entrenamiento a analizar, así como las diferentes opciones de gráficos que hay.

* 1. **DataBlock**

Clase que presenta por pantalla los diferentes bloques que tenga un entrenamiento para su posterior análisis por separado.

Contiene las siguientes funciones:

-onCreate(): Inicializa y presenta la lista de bloques del entrenamiento a analizar.

-isFree(): Función que calcular si un entrenamiento es libre o personalizado, y por lo tanto, tiene bloques. Return String.

-onCreateOptionsMenu(Menu menú): Inicializa el menú toolbar.

-onOptionsItemsSelected(MenuItem ítem): Gestiona los diferentes elementos del menú toolbar.

-alertMessageP2P(): Muestra un diálogo de alerte al usuario para confirmar si se quiere enviar el entrenamiento de forma inalámbrica.

-alertMessage(): Muestra un diálogo de confirmación para borrar el entrenamiento cuando se selecciona esta opción en el menú toolbar.

-deleteSession(): Borra el entrenamiento.

-onRestart(): Actualiza la lista de entrenamientos cuando se borran.

* 1. **DataGraphic**

Clase que se encarga de crear y mostrar los gráficos de los datos del entrenamiento a analizar.

Contiene las siguientes funciones:

-onCreate(): Calcula todos los puntos de la gráfica a mostar.

-drawGraphics(): Dibuja todos estos puntos calculados anteriormente.

-onCreateOptionsMenu(Menu menú): Inicializa el menú toolbar.

-onOptionsItemsSelected(MenuItem ítem): Gestiona los diferentes elementos del menú toolbar.

-getData(): Función que calcula los puntos (x,y) de la gráfica a partir de los datos del entrenamiento.

-alertMessage(): Muestra un diálogo de confirmación para borrar el entrenamiento cuando se selecciona esta opción en el menú toolbar.

-deleteSession(): Borra el entrenamiento del sistema.

1. **Etc**

Conjunto de clases encargadas de distintas funcionalidades para el correcto funcionamiento de la aplicación. Está formada por las siguientes subcarpetas:

* 1. **Frequency**

En esta subcarpeta se encuentran todos los archivos necesarios para calcular la frecuencia de paladas durante un entrenamiento.

Esta formada por las siguientes clases que ya se explicó su funcionalidad en la memoria del proyecto.

* AverageReverseFilter
* ExtrapolatorTransformer
* RateMeasurerReverseFilter
* SamplerSinkAndSource
* SensorsSimulatorMultiSource
* StandardOutputConsumer
* StrokeDetectorReverseFilter
* ZReverseFilter
* DataConsumer
* DataMultiSource
* DataProducer
* DataSink
* DataSource
* DataTransformer
* FPieceOfData
* PieceOfData
* TFPieceOfData
* TimedPieceOfData
* TPieceOfData
* TV3PieceOfData
* V3PieceOfData
  1. **listItem**

Esta subcarpeta está formada por las siguientes clases que se encargarán de mostrar las listas personalizadas de los entrenamientos y sesiones de la carpeta dataSesion.

* + 1. **AdapterCategory**

Clase adaptador de la lista personalizada.

Tiene las siguientes funciones:

-getCount(): Devuelve el total de objetos de la lista. Return int.

-clear(): Limpia la lista de objetos.

-getItem(int): Devuelve el objeto de la lista seleccionado. Return Object.

-getItemID(int): Devuelve el id del objeto seleccionado. Return long.

-getView(): Devuelve la vista del adaptador de la lista. Return View.

* + 1. **Category**

Clase que contiene los atributos de la lista personalizada, así como sus getters y setters.

* 1. **music**

En esta subcarpeta se encuentra la clase SoundManager, que se encarga de gestionar la reproducción de sonidos de la aplicación.

Contiene las funciones load(int) y play(int) para cargar y reproducir respectivamente un sonido.

* 1. **wifiDirect**

En esta subcarpeta se encuentran las clases que gestionan la función de compartir entrenamiento entre el entrenador y sus palistas de forma inalámbrica.

Esta formado por las siguientes clases.

* + 1. **WifiDirect**

Clase principal de esta función de la aplicación.

Está formada por las siguientes funciones:

- onCreate(): Función que se encarga de buscar y mostrar por pantalla todos los dispositivos disponibles para enviar el entrenamiento correspondiente, además de registrarse al broadcast.

-onResume(): Cancela su registro al broadcast.

-onPause(): Vuelve a suscribirse al broadcast.

-onBackPressed(): Finaliza la actividad cuando se pulsa el botón back.

Además, está formada por las siguientes clases:

+ServerClass: Clase servidor de esta funcionalidad. Extend Thread.

+ClienteClass: Clase cliente de esta funcionalidad. Extend Thread.

* + 1. **WifiDirectBroadcastReceiver**

Clase Broadcast de esta funcionalidad.

Está formada por las siguientes clases:

-onReceive(): Se encargará de gestionar los mensajes que reciba el broadcast y dependiendo de la acción que se trate se ejecutarán diferentes opciones.

Estas acciones son:

-Cuando el estado del Wifi cambia

-Cuando un dispositivo cambia

-Cuando la conexión P2P cambia

* 1. **Xml**

Subcarpeta formada por la clase XML que se encargará de guardar los datos de las sesiones personalizadas, así de leerlos para su posterior entrenamiento.

Está formada por las siguientes funciones:

-getInfoXML(): Función que devuelve toda la información correspondiente de una sesión. Return String [ ].

-writeSessionXML(): Función que guarda los datos que ha introducido el usuario en un fichero xml para almacenarlo en el sistema. Return true.

-calcDate(): Función que devuelve la fecha actual. Return String.

1. **main**

En esta carpeta se encontrará la clase de inicio de la aplicación.

Está formada por la siguiente clase:

* 1. **Main**

Es la primera clase que se ejecuta al iniciar la aplicación. En ella se pedirán los permisos necesarios para realizar las diferentes funciones de la aplicación y se crearán las carpetas necesarias para guardar los entrenamientos y sesiones.

Contiene las siguientes funciones:

-onCreate(): Pide los permisos necesarios y comprueba si existen las carpetas necesarias para el correcto funcionamiento de la aplicación, si no es así, las crea.

-onRequestPermissionsResult():

-isReadStoragePermissionsGranted():

-isWriteStoragePermissionsGranted(): Son tres funciones que se encargarán de preguntar los permisos necesarios y recibir las respuestas de estas peticiones.

1. **planningSession**

Conjunto de clases encargadas de la funcionalidad de crear un entrenamiento personalizado para su posterior entrenamiento.

Esta formada por las siguientes clases:

* 1. **Planning**

Clase que presenta por pantalla una lista personalizada con todas las sesiones que hallamos creado.

Contiene las siguientes funciones:

-onCreate(): Inicializa y representa la lista de sesiones personalizdas.

-onCreateOptionsMenu(Menu menú): Inicializa el menú toolbar.

-onOptionsItemsSelected(MenuItem ítem): Gestiona los diferentes elementos del menú toolbar.

-onRestart(): Actualiza la lista de sesiones cuando se borran o renombran.

* 1. **NewSession**

Clase que presenta por pantalla una interfaz para que el usuario introduzca los datos de la aplicación.

Contiene las siguientes funciones:

-onCreate(): Prepara y presenta la interfaz para que el usuario pueda introducir los datos de la sesión.

-minus(EditText): Función relacionada con un botón que reducirá una unidad el número que se encuentre en el EditText correspondiente.

-plus(EditText): Función relacionada con un botón que aumentará una unidad el número que se encuentre en el EditText correspondiente.

-onCreateOptionsMenu(Menu menú): Inicializa el menú toolbar.

-onOptionsItemsSelected(MenuItem ítem): Gestiona los diferentes elementos del menú toolbar.

-alertMessage(): Función que creará un diálogo para alertar al usuario de si quiere guardar los datos de la sesión.

-saveSession():Función que guardará los datos introducidos por el usuario si este selecciona la opción de guardar sesión.

* 1. **Session**

Clase que presentará al usuario los datos de una sesión creada con anterioridad de forma clara y ordenada.

Contiene las siguientes funciones:

-onCreate(): Carga los datos y los presenta en una interfaz de forma clara y ordenada.

-onCreateOptionsMenu(Menu menú): Inicializa el menú toolbar.

-onOptionsItemsSelected(MenuItem ítem): Gestiona los diferentes elementos del menú toolbar.

-startMessage(): Función que creará un diálogo para alertar al usuario de que si quiere iniciar esa sesión para realizar el entrenamiento correspondiente.

-startSession(): Función que iniciará la actividad Training con los datos de esta sesión personalizada.

-alertMessage(): Muestra un diálogo de confirmación para borrar la sesión cuando se selecciona esta opción en el menú toolbar.

-deleteSession(): Borra la sesión del sistema.

1. **Training**

Conjunto de clases encargadas del cálculo, gestión y tratamiento de los datos en directo de un entrenamiento.

Esta formada por las siguientes clases:

* 1. **Training**

Clase que presenta por pantalla la interfaz principal del entrenamiento, los datos de la sesión personalizada si la hemos iniciado, crea los servicios y procesos que, en segundo plano, nos calcularán velocidades, distancias, frecuencias y tiempos.

Contiene las siguientes funciones:

-onCreate(): Función de inicio que presenta la interfaz principal del entrenamiento y los datos de la sesión personalizada si la hemos iniciado. A parte de los servicios y procesos en segundo plano que calcularán y procesarán los datos del entrenamiento.

-startTimer(): Crea una cuenta atrás con sonidos y vibración antes de iniciar el entrenamiento .

-calcDate(): Función que calcula y devuelve la fecha actual. Return String.

-locationStart(): Crea la clase y se registra al servicio del GPS del dispositivo móvil.

-startBackgroundService(): Crea y se registra al servicio que calculará la frecuencia de paladas del palista.

-finishBackground(): Finaliza el servicio creado en la función anterior.

-alertMessage(): Función que creará un diálogo para alertar al usuario de si quiere guardar los datos del entrenamiento.

-updateCrono(): Función que se encargará de actualizar el tiempo restante de cada serie y descanso de la aplicación.

-updateStrokes(): Función que se encargará de actualizar la frecuencia de paladas calculada por el servicio y la información de la series o descansos correspondientes.

-updateGPS(): Función que recibe las actualizaciones del GPS y calcula la velocidad y distancia a partir de estos datos.

-onBackPressed(): Función que no permite que el usuario se salga de la interfaz al pulsar el botón back a no ser que pulse el botón Stop.

-feedBack(): Función que realimenta con sonidos y vibraciones el inicio y final de series y descansos.

-onDestroy(): Función que se ejecuta cuando se destruye la actividad y termina con los servicios en segundo plano que sigan activos.

-onStop(): Función que se ejecuta cuando se para la actividad y termina con los servicios en segundo plano y reestablece a 0 los valores de las medidas.

* 1. **Background**

Clase que se crea al iniciar el entrenamiento y se encarga de gestionar los tiempos del cronómetro y actualizarlos en la interfaz principal en segundo plano.

Contiene las siguientes funciones:

-start(): Se ejecuta al crear la clase y toma el tiempo inicial del entrenamiento.

-stop(): Se ejecuta cuando se para el entrenamiento y hace que se deje de contar tiempos.

-run(): Función que se ejecuta los mientras el entrenamiento esté activo y que calcula y da el formato correcto al cronómetro de la aplicación.

-checkForPaused(): Función que comprueba si el entrenamiento está pausado para pausar el hilo en el que se ejecuta la clase.

-pauseThread(): Función que se ejecuta cuando se pulse el botón pause de la interfaz de entrenamiento, que pausará el hilo en el que se ejecuta la clase.

-resumeThread(): Función que se ejecuta cuando se pulsa el botón resume de la interfaz principal de entrenamiento, que reanudará la ejecución del hilo en el que se ejecuta la clase.

* 1. **MyLocationListener**

Clase encargada de subscribirse al servicio del GPS y recibir los datos de este cuando se actualice.

Contiene las siguientes funciones:

-onLocationChanged(): Se ejecuta cuando cambian las coordenadas del GSP.

-onStatusChanged(): Se ejeceuta cuando cambia el estado del GPS.

-onProviderEnabled(): Se ejecuta cuando el GPS está disponible.

-onProviderDisabled(): Se ejecuta cuando el GPS deja de estar disponible.

* 1. **MyService**

Servicio que se encargará de calcular la frecuencia de paladas haciendo uso del servicio del acelerómetro y de las clases de la carpeta etc/frequency.

Contiene las siguientes funciones:

-onStartCommand(): Se ejecuta al iniciarse el servicio y crea todas las clases necesarias para calcular la frecuencia que quedan a la espera de datos del acelerómetro.

-onDestroy(): Se ejecuta cuando se destruye el servicio y deja de estar registrado al acelerómetro, así como eliminar las clases creadas anteriormente.

-onBind(): Devuelve un canal de comunicación entre un servicio y una clase asociada.

-onUnbind(): Se ejecuta cuando deja de estar conectado con una clase.

-addConsumer(): Añade tantos consumidores (Sensores del teléfono) como sean necesarios, en este caso solo añade el acelerómetro.

-getFrecuency(): Se ejecuta cuando llegan nuevos datos del acelerómetro y envía estos a la cadena de clases creada en la función onStartCommand().

-onNewFrek(): Función a la que llega una frecuencia cada vez que se calcula una nuevay prepara una cadena con los datos a mostrar en la interfaz.

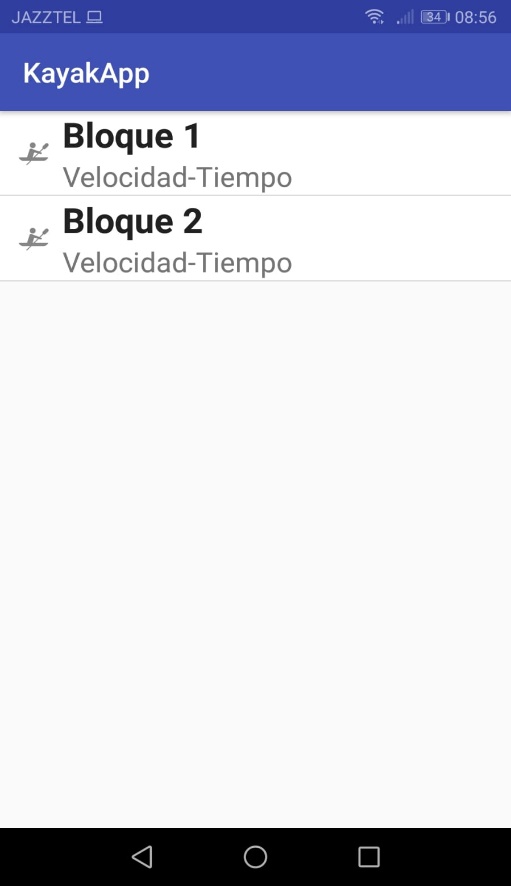
-getData(): Función que es llamada desde la interfaz del entrenamiento y que devuelve la última frecuencia calculada.

Además, tiene una clase SensorEventListener al que le llegarán los datos del acelerómetro cada vez que este se actualice, que se los mandará a la función getFrecuency().

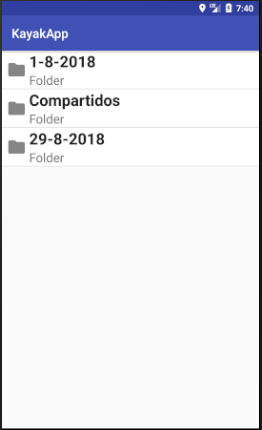
1. **Recursos layout XML**

En la carpeta res/layout del proyecto se encuentran todos los ficheros xml que implementarán las interfaces de la aplicación. Son las siguientes:

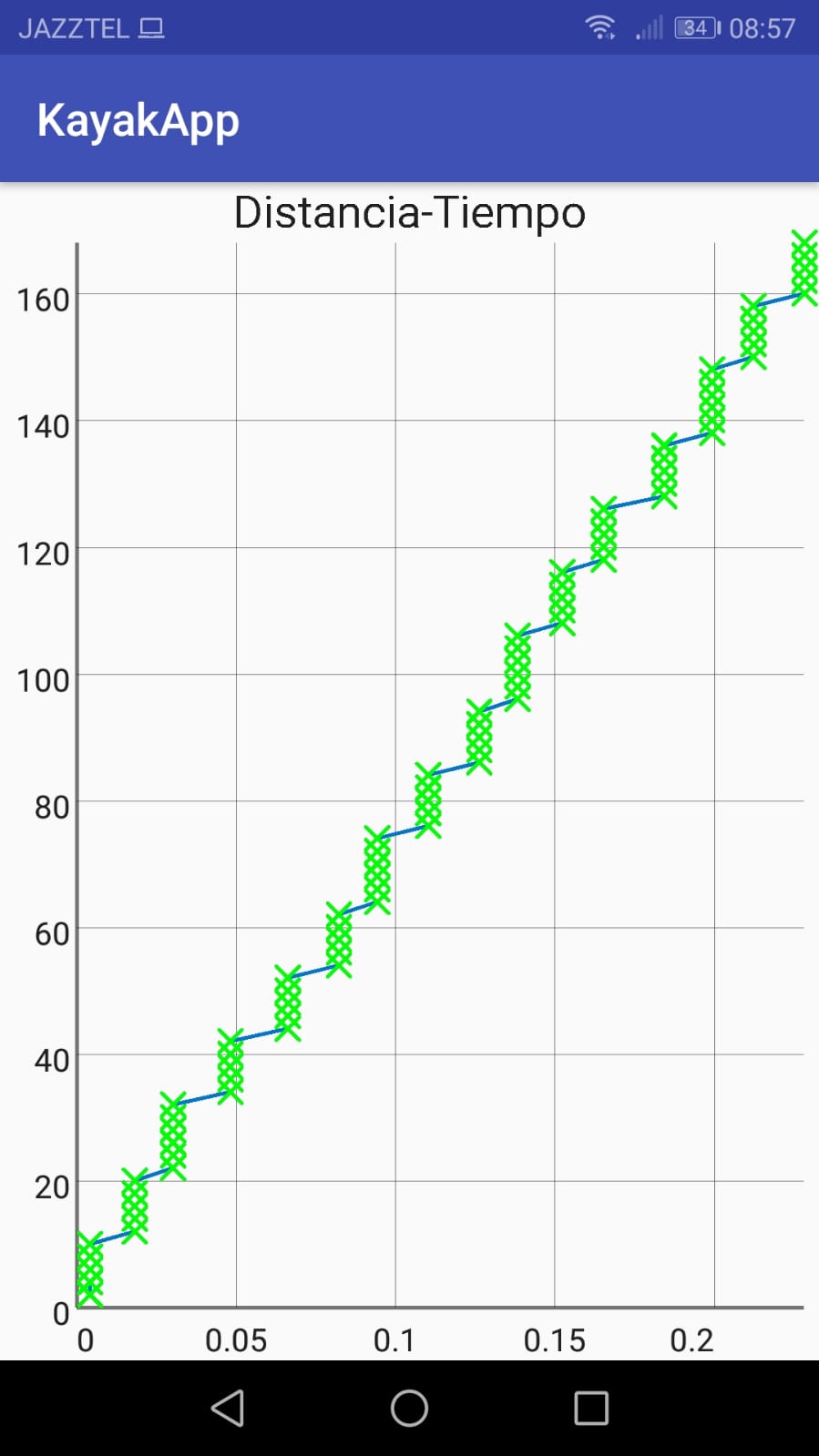
* activity\_data\_blocl.xml: Corresponde a la clase DataBlock.



* activity\_data\_folder.xml: Corresponde a la clase DataFolder.



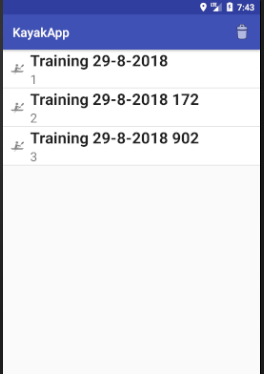
* activity\_data\_graphic.xml: Corresponde a la clase DataGraphic.



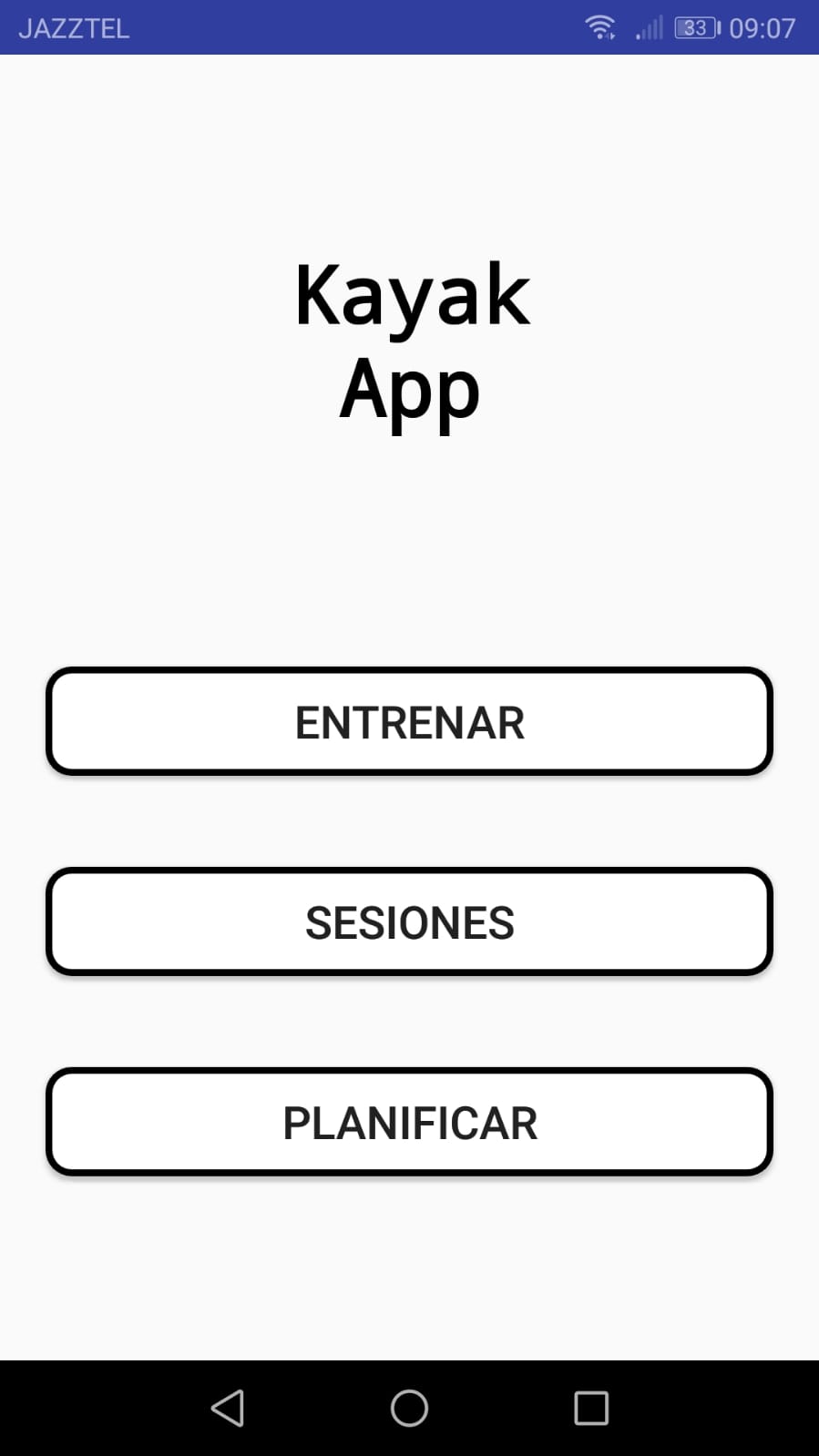
* activity\_data\_study: Corresponde a la clase DataStudy.



* activity\_data\_training: Corresponde a la clase DataTraining.



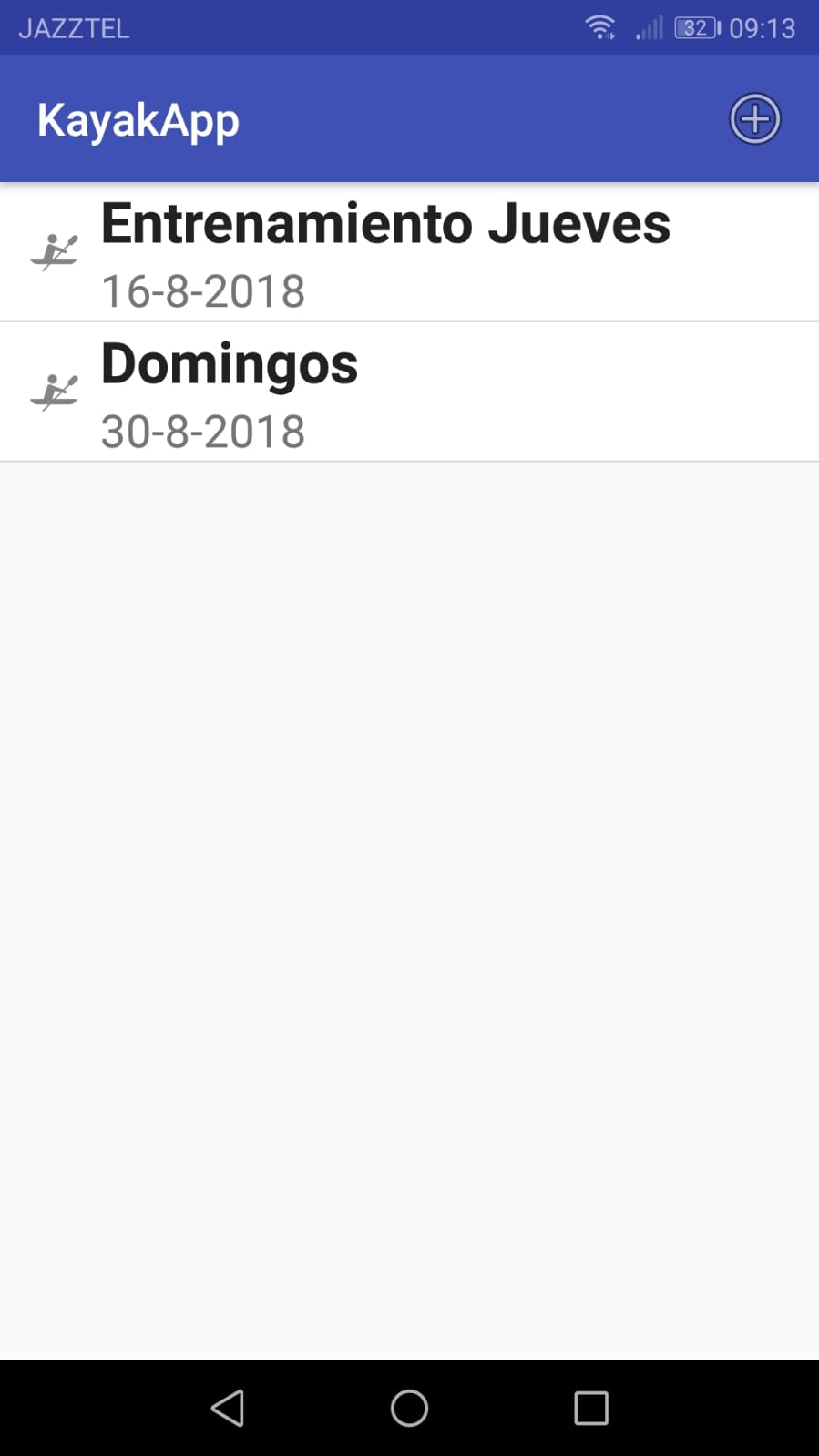
* activity\_main: Corresponde a la clase MainActivity.



* activity\_new\_session.xml: Corresponde a la clase NewSession.



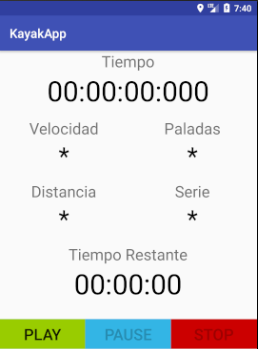
* activity\_planning.xml: Corresponde a la clase Planning.



* activity\_session.xml: Corresponde a la clase Session.



* activity\_training.xml: Corresponde a la clase Training.



* activity\_wifi\_direct.xml: Corresponde a la clase WifiDirect.



* custom\_dialog.xml: Corresponde con la interfaz del diálogo que se usa para renombrar entrenamientos.

