

Soccer Report

Arquitecturas Multimedia y de Propósito Específico

4º Grado en Ingeniería Informática

Alumnos:

Ginés Ros García

Carlos Amat Fernández

Profesor:

Gregorio Bernabé García

Convocatoria de Febrero de 2020

Índice

Introducción.....	3
Objetivos.....	3
Interfaz gráfica.....	4
Jugada de área o penalti.....	9
Guardar lo que hay en la memoria.....	10
Seleccionar eventos desde un desplegable.....	12
Exportar a excel.....	13
Funcionalidad adicional.....	14
Botón deshacer.....	14
Persistencia de objetos.....	14
Cronómetro.....	15

Introducción

Este trabajo ha consistido en la mejora de la aplicación Soccer Report abarcando la interfaz gráfica y la funcionalidad.

Para el desarrollo se ha empleado *android studio*, se ha subido la versión mínima de la aplicación a la 24 (Android Nougat) ya que era necesario para el uso de acceso a la room de Android para guardar los partidos, uso de streams y componentes de los layouts para mejorar la interfaz gráfica. En los emuladores utilizados se han definido las siguientes características:

Versión de Android	Tamaño de pantalla	Resolución	Tipo de dispositivo
7.0	5"	1080x1920	Móvil
7.0	10.1"	800x1280	Tablet
9.0	5.5"	1080x1920	Móvil

Objetivos

Se han establecido los siguientes objetivos:

- Re-diseñar la interfaz gráfica y mejorarla. En especial, el teclado no debe de tapar la pantalla cuando se introducen los datos.
- Introducir para cada equipo la posibilidad de Jugada de área o penalti.
- Solventar el problema de guardar lo que hay en la memoria en almacenamiento secundario.
- Poder elegir una serie de eventos para introducir de una lista desplegable y poder introducir una descripción.
- Crear una ventana dónde aparezcan todos los eventos que se van introduciendo y que se puedan seleccionar aquellos que se quieren volcar a una excel. Poder volcar los eventos seleccionados (tiempo, evento y descripción) a una excel.

Cabe destacar que además de los objetivos fijados se han añadido una serie de mejoras como puede ser la adaptación de la pantalla dependiendo de la resolución del dispositivo para que se pueda usar tanto en tablets como en móviles, empleo de Vectores en lugar de imágenes para obtener imágenes nítidas en la pantalla, añadida la opción de poder deshacer una acción o el uso de la room de Android para permitir almacenar los datos en una base de datos permitiendo la opción de multipartido y guardar las incidencias que introduce el usuario.

Para realizar todas estas mejoras se han añadido nuevas clases, layouts, vectores, etc que se explicarán a continuación.

Interfaz gráfica

Se han incorporado nuevos layouts:

1. new_principal.xml
2. new_incidencia.xml
3. new_incidencia.xml (sw 760dp)
4. new_estadoactual.xml

La primera pantalla que se le presenta al usuario es new_principal.xml, donde el usuario introducirá los nombres de los equipos locales y visitante, además de la fecha y hora del partido.

Desde esta pantalla el usuario puede cerrar la aplicación, cargar partidos anteriores o pasar a la siguiente pantalla, esto último sólo va a ocurrir si ha introducido los nombres de ambos equipos.

Para facilitar la introducción de los datos, tanto la fecha como la hora se introducirán automáticamente, no obstante el usuario podrá modificarlas posteriormente.

La pantalla tiene el siguiente aspecto:



Figura 1. Vista de la pantalla principal desde un dispositivo móvil




Figura 2. Vista de la pantalla principal desde una tablet

Si el usuario pulsa el botón *cargar* le aparecerá un listado con todos los partidos que ha realizado en el último estado que modificó.



Figura 3. Pantalla de carga de partidos anteriores

Una vez el usuario pulsa el botón siguiente pasa a la siguiente pantalla, `new_incidencia.xml`. En este caso dependiendo de la pantalla del dispositivo que se está usando se cargará un layout determinado.

Concretamente se ha establecido que para a partir de un ancho mínimo de pantalla de 760dp, tamaño medio de una pantalla de tablet.

Para conseguir esto, hay que crear una carpeta llamada *layout[-propiedades]*, en nuestro caso nos quedaría de la siguiente forma:

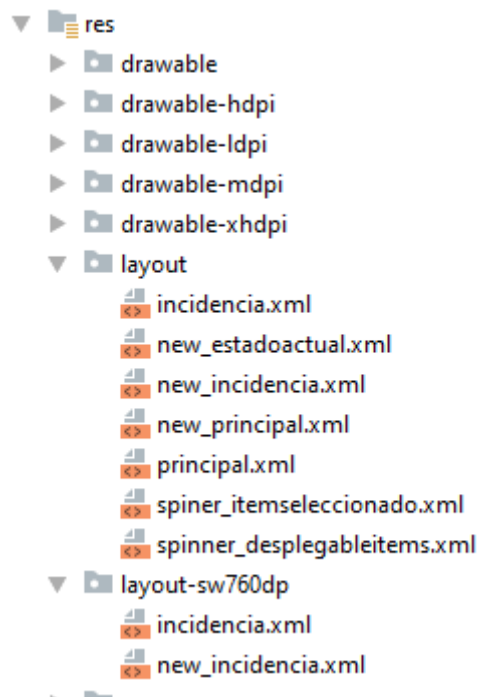


Figura 4. Estructura de layouts

De tal forma que si usamos un dispositivo móvil veremos la siguiente pantalla:



Figura 5. Vista de new_incidencia.xml desde un móvil

Sin embargo si usamos una tablet tendremos lo siguiente:

The screenshot shows a tablet interface for a soccer report application. The main area is a green field with a score of 3-2 (Murcia 3, Cartagena 2) at 00:35. The field is divided into two halves, each with buttons for 'Gol', 'Falta', 'Jugada de área', and 'Penalti'. A 'Dorsal' button is at the bottom left. The right sidebar contains a 'Fin 1ª parte' button, a 'Seleccione una opción' dropdown, and a list of events. At the bottom, there are buttons for 'Local', 'Visitante', 'Entra', 'Sale', and 'Añadir evento Gol'. The event log on the right lists goals, yellow cards, and red cards for both teams.

Figura 6. Vista de new_incidencia.xml desde una tablet

Se puede comprobar a primera vista que desde una tablet tenemos un panel lateral derecho que nos muestra el estado del partido, todos los eventos que introduzca el usuario hará que se actualice este panel de forma automática para poder ver el estado actual del partido.

Para ayudar al usuario a introducir incidencias, cada vez que añada una nueva, aparecerá un mensaje indicando el tipo de incidencia que ha introducido, tal como aparece en la figura 6.

Además se ha añadido un botón *Ver estado* que nos llevará a la última vista añadida, *new_estado.xml*, desde esta vista podemos ver este mismo panel lateral que aparece en la tablet, de esta forma también podemos ver este contenido desde la vista de un dispositivo móvil. Este layout se ha implementado con un *scrollView* por lo que si el contenido es más grande que el alto de la pantalla, podremos deslizarlo para ver el contenido completo.

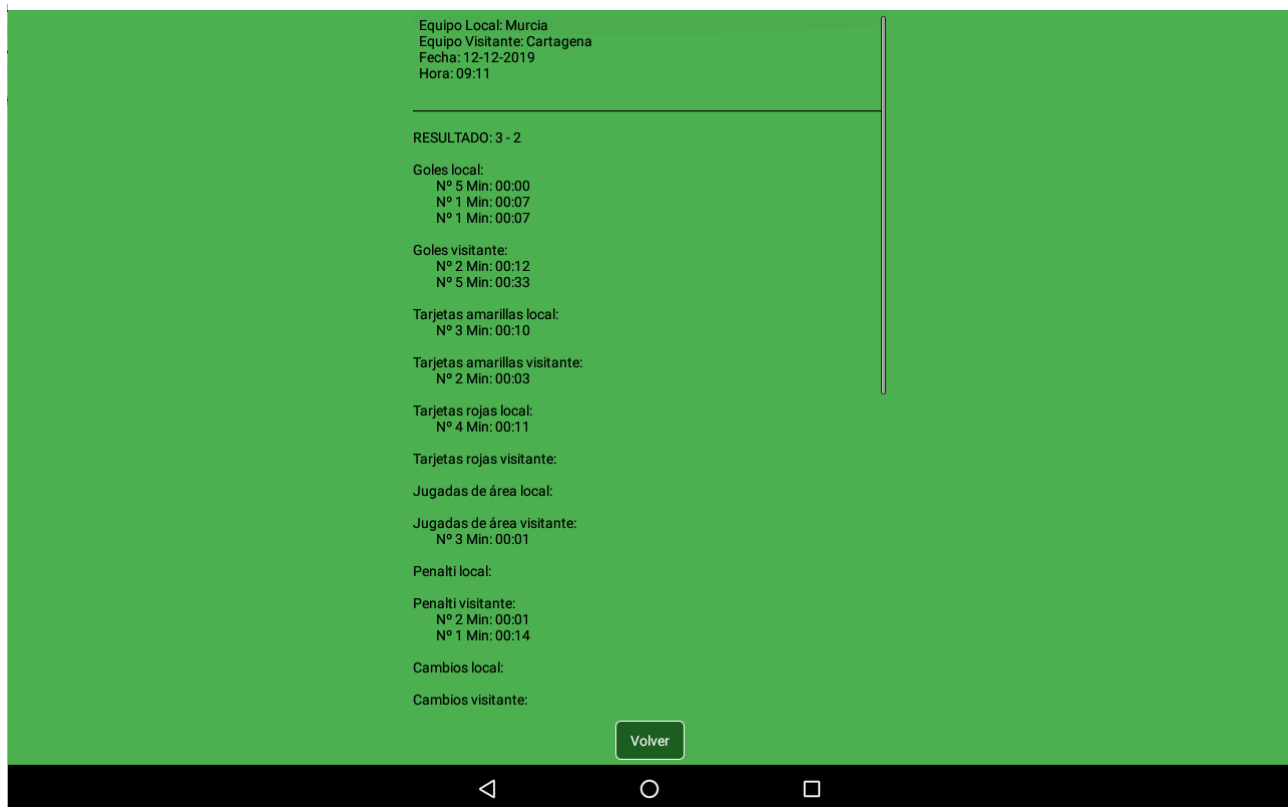


Figura 7. Vista de new_incidencia.xml

Para potenciar la adaptabilidad, se ha establecido que todos los layouts sólo se puedan de forma horizontal para aprovechar mejor la pantalla. Además se ha empleado de base un *constraint Layout* y *guides* que nos permite la posibilidad de adaptar los elementos en posiciones relativas de los mismos, como se muestra en la Figura 8.

En cuanto a la estética de las nuevas ventanas se ha establecido el uso de Vectores en lugar de imágenes, además para los eventos se ha seguido usando botones excepto para los casos de tarjeta roja, tarjeta amarilla o cambio, esto es así debido a que si por cada botón se sustituía por una Vector, la pantalla quedaría muy cargada, por tanto esta opción presenta un buen equilibrio en cuando al contenido y diseño.

Se han sustituido los textos en los que se indicaba qué era cada campo por el *hint* del mismo.

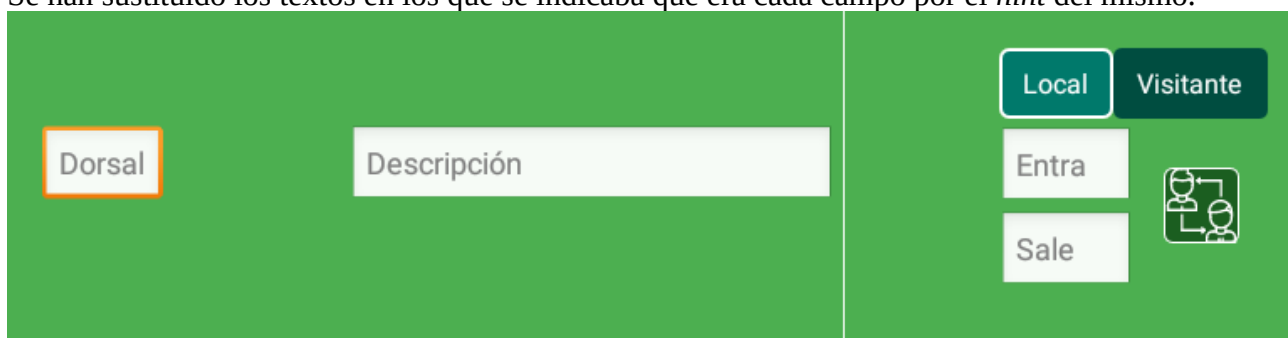


Figura 9. Uso de atributo hint de los Edit Text

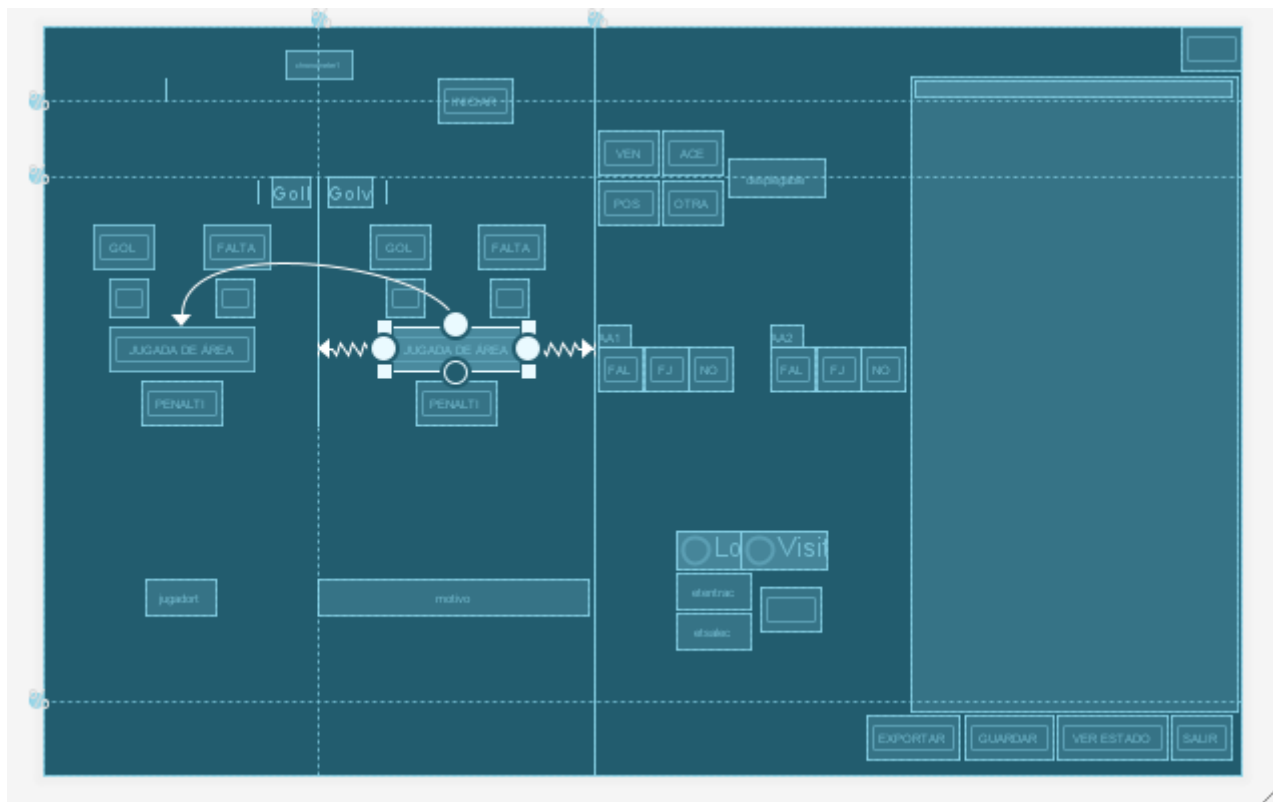


Figura 9. Blueprint de new_incidencia.xml

Jugada de área o penalti

Como se puede ver tanto la Figura 5 y Figura 6, se ha añadido estas dos opciones, como ocurre en la versión anterior aparecen duplicadas, una para cada equipo.

Inicialmente se pretendía quitar esta duplicidad mediante un switch el cual nos permitiera seleccionar entre Equipo Local o Equipo Visitante y posteriormente añadir la incidencia, como ocurre en el caso de los cambios, sin embargo esto hacía que la introducción de incidencias fuera lenta, por lo que finalmente se ha mantenido esta opción.

Como ocurre con todas las incidencias, se permite añadir una descripción, el cuál se asociará a dicha incidencia.

Guardar lo que hay en la memoria

Lo primero que se ha modificado es el lugar donde se almacena los informes generados con el botón *Guardar*, siendo el nuevo lugar el almacenamiento interno, de esta forma nos evitamos problemas en el caso de que se use en un dispositivo que no pueda tener una tarjeta SD externa y por tanto carezca de almacenamiento externo. Android nos proporciona el siguiente método para acceder al almacenamiento interno: `Environment.getExternalStorageDirectory()`

Para tener acceso a este path es necesario que el usuario de permiso a la aplicación para que pueda realizar operaciones de escritura y lectura desde la memoria principal, esto se hace mediante los siguientes pasos:

1. Modificar fichero `AndroidManifest.xml`
Añadimos el siguiente contenido

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
```

2. Mostrar una ventana en la que el usuario pueda aceptar estos permisos

```
ActivityCompat.requestPermissions(this, new String[]
{Manifest.permission.WRITE_EXTERNAL_STORAGE}, requestCode);
```

De esta forma al usuario se le mostrará el siguiente aviso:

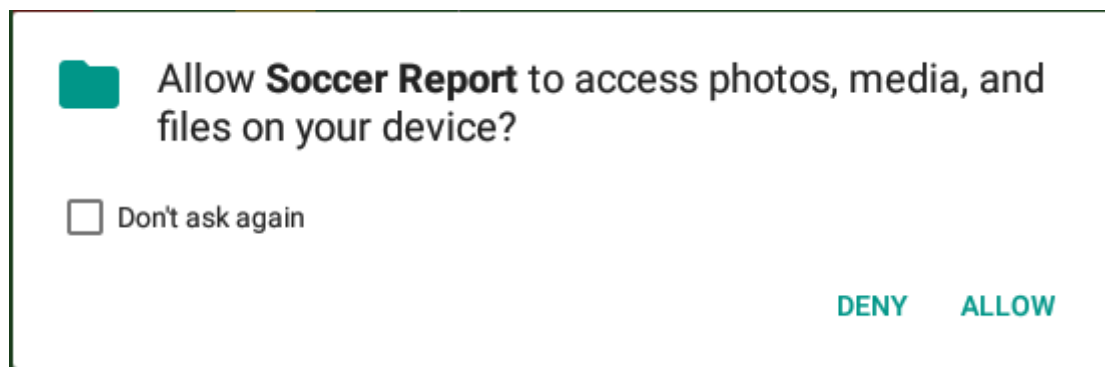


Figura 10. Solicitar acceso al almacenamiento interno

Una vez tenemos estos permisos del usuario, ya podemos crear nuestra carpeta de Soccer Report en el almacenamiento interno, además para facilitar la gestión de los informes se va a crear una carpeta por cada día y dentro de ellas estarán los informes que generemos mediante la aplicación.

Por tanto al pulsar sobre el botón guardar, primero solicitará al usuario los permisos en caso de que no los haya aceptado, a continuación creará la jerarquía de carpetas por días indicada anteriormente

y por último guardará los dos informes, uno similar al contenido de *new_estado.xml* y otro informe ordenado en el que tendremos los eventos del partido ordenados por tiempo.

En caso de volver a generar otros informes siempre se van a mantener los últimos, esta decisión se ha tomado para evitar una proliferación de ficheros pdf.

Además nos mostrará un mensaje si se han guardado correctamente con la ruta donde están almacenados o en caso contrario un mensaje indicando el error, como puede ser la no aceptación del acceso a los recursos.

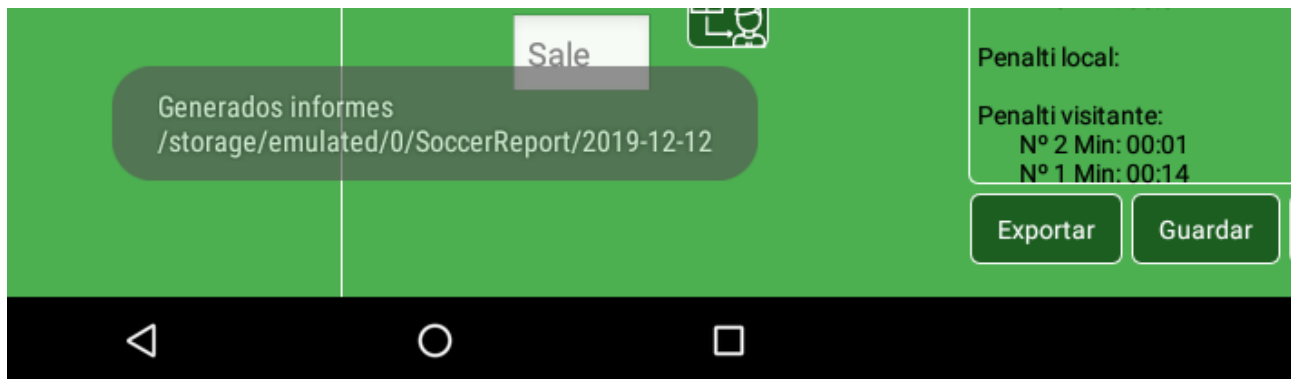


Figura 11. Mensaje para indicar que se han guardado correctamente

Para generar los informes se ha utilizado streams y predicados, por lo que en la clase *Partido* encontramos métodos como el siguiente:

```
public ArrayList<Incidencia> filtrarLocal(ArrayList<Incidencia> datos) {
    Predicate<Incidencia> predicadoLocal = new Predicate<Incidencia>() {
        @Override
        public boolean test(Incidencia incidencia) {
            return incidencia.getTipo() == TipoEquipo.Local;
        }
    };
    List<Incidencia> f =
datos.stream().filter(predicadoLocal).collect(Collectors.<Incidencia>toList());
    ArrayList<Incidencia> datosfiltrados = new ArrayList<Incidencia>(f);
    return datosfiltrados;
}
```

De esta forma se ha mejorado la funcionalidad de los informes y se han corregido errores previos.


Además de poder generar estos informes, el estado del partido se va a guardar internamente en una base de datos automáticamente, lo que va a permitir guardar el estado de un partido, permitiendo la opción de multipartido o recuperación de los datos en caso de cierre de la aplicación.

Esta nueva funcionalidad se explicará en un apartado posterior.

Seleccionar eventos desde un desplegable

Para esto se ha utilizado un *spinner* en el que nos mostrará un listado con todos los eventos proporcionados, dichos eventos están definidos en *contenidoDesplegable.xml*, situado en la carpeta *res/values*

Como se puede apreciar en la figura 5 y figura 6, en la parte superior derecha nos aparece una barra con el texto “*seleccionar opción*” al pulsar sobre ella nos aparecerán todos estos eventos, si pulsamos sobre uno de ellos automáticamente se creará una incidencia del tipo correspondiente.



The image shows a mobile application interface with a dropdown menu. The menu is titled "Seleccione una opción" and contains five visible options: "No Falta", "Imprudente", "Temerario", "Juego Brusco Grave", and "Conducta violenta". Each option is accompanied by a radio button. The "Seleccione una opción" header has a green dot in its radio button, indicating it is the selected option. The menu is styled with a light gray background and a dark border.

Opción	Seleccionado
Seleccione una opción	Yes
No Falta	No
Imprudente	No
Temerario	No
Juego Brusco Grave	No
Conducta violenta	No

Figura 12. Lista desplegable

Esto tipo de eventos aparece en los informes en el apartado “*Lista desplegable*”. Además como ocurre con el resto de incidencias, si añadimos una descripción antes de seleccionar la opción, esta se guardará junto con la incidencia.

Exportar a excel

Este apartado es similar al apartado donde almacenamos los datos del partido en el almacenamiento interno, es necesario que la aplicación tenga permisos para crear ficheros y se guardará en el mismo path dependiendo de la fecha del partido.

En este caso no se ha realizado una exportación a un fichero xlsx directamente, ya que dependiendo del entorno de ofimática podemos tener varias distribuciones, como puede ser el caso de libre Office que emplea ficheros ods, para facilitar tanto la programación y hacer que sea más flexible se ha optado por realizar estas exportaciones a ficheros csv, los cuales pueden ser posteriormente importados a cualquier hoja de cálculo, incluso en versiones actuales de Android, directamente se ejecutan sobre la aplicación “*hojas de cálculo*”.

En este caso al pulsar sobre el botón *exportar* se nos muestra los eventos que pueden aparecer en un partido, esta ventana permite múltiples opciones por lo que podemos seleccionar más de un tipo de incidencias.

Seleccione los eventos:	
Goles	<input checked="" type="checkbox"/>
Faltas	<input type="checkbox"/>
Tarjetas Amarillas	<input checked="" type="checkbox"/>
Tarjetas Rojas	<input type="checkbox"/>
Jugada de área	<input type="checkbox"/>
Penalti	<input checked="" type="checkbox"/>
Ventaja	<input type="checkbox"/>
Posición	<input type="checkbox"/>
Aceleración	<input checked="" type="checkbox"/>

OK

Cancelar

Figura 13. Selección de eventos a exportar a csv

Una vez seleccionados los eventos en la carpeta adecuada nos aparecerá el csv ordenado de forma cronológica, al igual como en el caso de guardar los datos en el almacenamiento interno, aparecerá un mensaje que nos indicará la ruta donde se han guardado o un error en otro caso.

No.	Minuto	Evento	Descripción
1	00:00	Gol	
2	00:01	Penalti	
3	00:03	Tarjeta Amarilla	
4	00:07	Gol	
5	00:07	Gol	
6	00:10	Tarjeta Amarilla	
7	00:12	Gol	
8	00:14	Penalti	
9	00:33	Gol	
10	38:19	Gol	Gol del empate!

Figura 14. contenido del CSV

Funcionalidad adicional

Además de todo lo mencionado anteriormente se ha añadido lo siguiente

Botón deshacer

En la parte superior derecha encontramos un nuevo botón, este botón eliminará la última incidencia que hayamos introducido.

De esta forma si en el caso de haber introducido una por error la podemos eliminar fácilmente.

Persistencia de objetos

Para realizar la persistencia se ha utilizado la room de android.

Esto consta de varias clases:

- AppDatabase, se ha implementado mediante un patrón singleton de esta forma podemos acceder desde cualquier parte de la aplicación, esta clase será la encargada de gestionar la base de datos.
- PartidoDao, esta clase es la que contiene los métodos donde se definen las Qwerys
- Partido, esta clase ya existía en la aplicación pero se ha modificado para que sea una Entity, esta clase va a ser los datos de la tabla que guardamos en la base de datos

- Conviértete, ya que nuestro objeto partido contiene una lista de objetos *Incidencia* y por tanto no es un tipo de objeto primitivo, es necesario indicar la forma en la que convertir esos datos a otro primitivo, en este caso se han convertido a un String mediante un Json, que en android se denomina Gson

Cronómetro

Ahora con el cronómetro presenta el comportamiento a un cronómetro real. Tiene cinco estados

1. Inactivo
2. Primera parte
3. Descanso
4. Segunda parte
5. Final del partidos

Al pulsar se realiza una transición de un estado a otro hasta llegar al final del partido, el cronómetro se inicia en la segunda etapa, se detiene en la tercera y última etapa.

En cualquiera de los casos al usuario le aparecerá el estado en el que se encuentra.

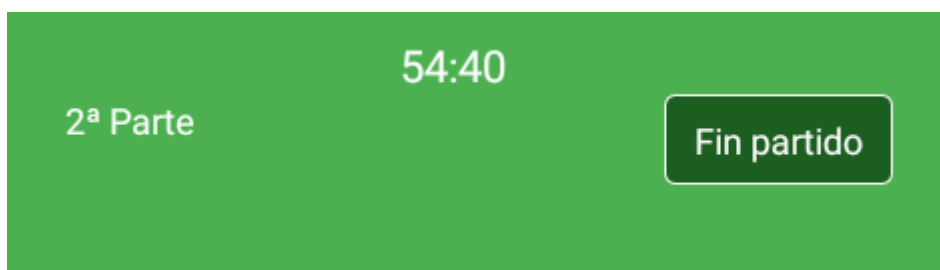


Figura 15. Cronómetro