PEC 1

Presentación

En esta primera PEC se creará la estructura de la aplicación con la que trabajaremos durante el curso. Se trabajarán los 3 primeros módulos de los apuntes de la asignatura.

Competencias

En esta PEC se trabajarán las siguientes competencias del Máster universitario en Desarrollo de aplicaciones para dispositivos móviles:

- Utilizar de manera efectiva los lenguajes de programación de las plataformas móviles más representativas del mercado.
- Usar las herramientas y entornos de desarrollo disponibles para las plataformas móviles más representativas del mercado.

Objetivos

En el desarrollo de la PEC se persiguen conseguir los siguientes objetivos:

- Aprender a desarrollar una aplicación para dispositivos móviles y tabletas con la estructura master-detail.
- Aprender a utilizar actividades y fragmentos en una aplicación.
- Realizar listas complejas con RecyclerView.
- Mostrar tarjetas con la vista CardView.
- Aprender a utilizar el ConstraintLayout.

Descripción de la PEC/práctica a realizar

PARTE PRÁCTICA

Ejercicio 1: Estructura de la aplicación

El primer ejercicio consiste en crear una aplicación Android con la estructura master-detail. Esta estructura se caracteriza por estar compuesta de dos pantallas: una master con una lista de items y una detalle con la información específica de cada item. La característica especial de este tipo de estructura es que en pantallas pequeñas se verá primero la lista de items y al pulsar sobre uno de ellos, su detalle; en cambio, en pantallas grandes se verá directamente la lista de items y el detalle del item seleccionado en la misma pantalla.

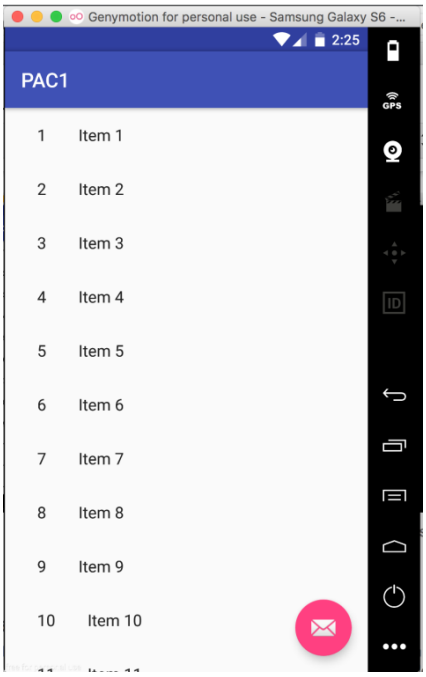
Abre el Android Studio, crea una nueva aplicación con el nombre y dominio que quieras y selecciona una aplicación para móvil y tableta con la versión mínima de Android 4.0.

En la siguiente pantalla se selecciona el tipo de actividad principal del proyecto, en este caso se utilizará "Add no activity". Esto hará que Android Studio implemente nuestro proyecto vacío.

Primero, crea las dos actividades principales del proyecto: **BookListActivity** y **BookDetailActivity,** para la lista y el detalle respectivamente, y regístralas en el archivo Manifiest.

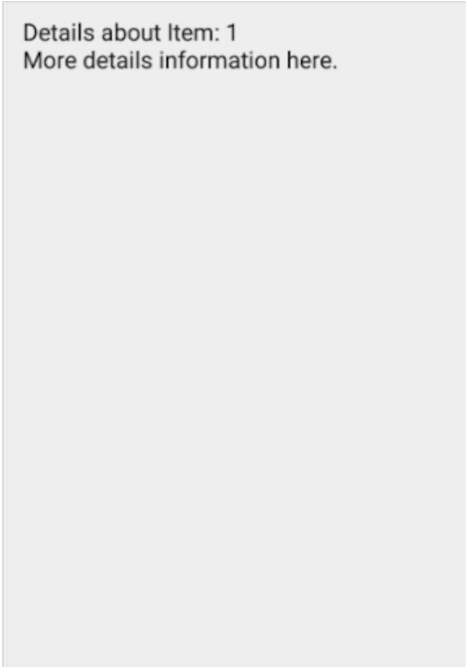
Para que se muestre un tipo de layout u otro según si estamos en móviles o tabletas, será necesario crear un layout XML diferente según el tipo de resolución de pantalla para la actividad **BookListActivity**.

Crea layout **book_list.xml** para la resolución **w900dp** con una lista simple de items y un frame layout para el detalle, y otro layout **book_list.xml** en la carpeta de la resolución por defecto con la lista de ítems solamente. Pon el mismo identificador para la lista de ítems, ya que esta lista se gestionará desde la misma actividad. En **BookListActivity,** rellena este layout con una lista por defecto (con valores Item 1, Item 2, Item 3 ...), tal y como se muestra en la siguiente imagen:



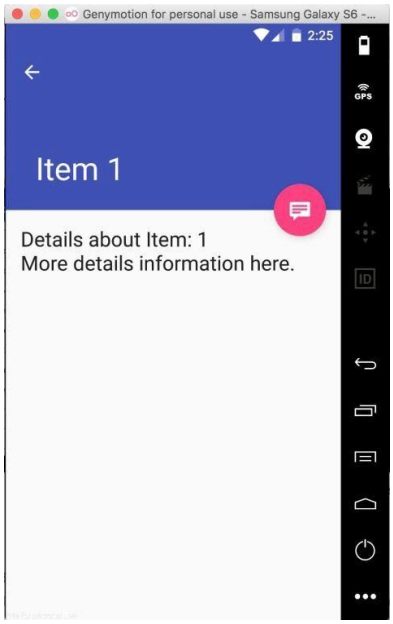
Desde la actividad **BookListActivity,** detecta si el frame layout está presente en la actividad. En caso de que esté presente, significa que estamos en una tableta, por lo tanto, se deberá mostrar el detalle de la lista dentro de este frame.

Para mostrar el detalle de la lista tanto en tableta como en teléfono móvil, será necesario utilizar un fragmento. Este fragmento se añadirá al frame de **BookListActivity** o directamente en **BookDetailActivity.** Por lo tanto, crea un fragmento **BookDetailFragment** donde se mostrarán los detalles del libro. De momento, crea unos detalles por defecto como se muestra en la siguiente imagen:



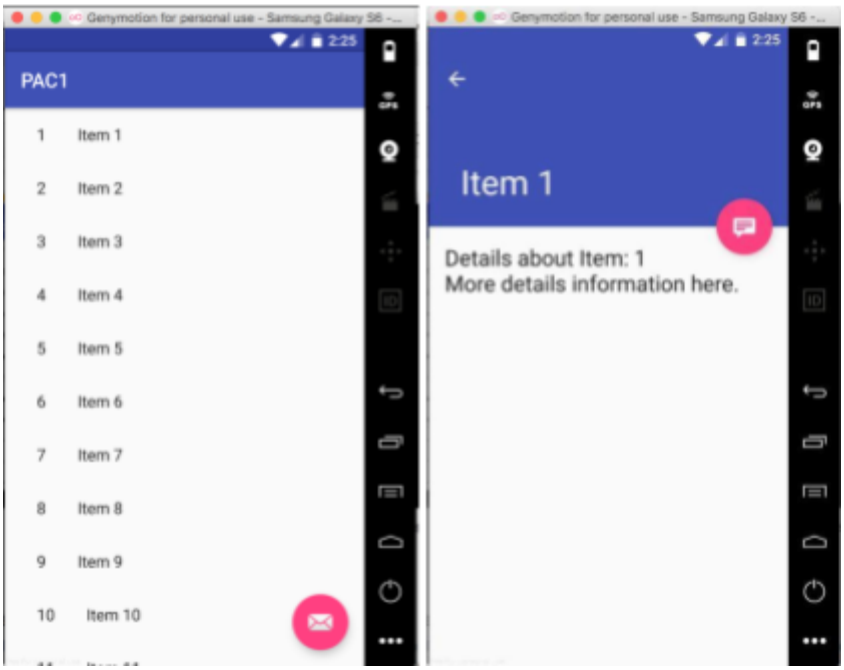
En **BookListActivity** añade este fragmento al frame layout creado anteriormente, en caso de que estemos en una tableta.

BookDetailActivity es la actividad donde se mostrará el detalle del libro. Esta actividad se utilizará solo en caso de que estemos en un teléfono móvil. Por lo tanto, modifica su layout para que tenga un frame donde se pueda añadir el fragmento anterior. Puedes customizar esta actividad tal y como se muestra en la siguiente imagen, para que tenga una barra superior y el detalle del fragmento esté abajo:



Una vez preparadas estas actividades y el fragmento, si ejecutamos el código en una tablet en horizontal, veremos que en la misma pantalla está la lista y el detalle:

En cambio, si ejecutamos el código en un teléfono tenemos dos pantallas, la de la lista y la del detalle:



Ejercicio 2: Implementar una lista con RecyclerView

En el segundo ejercicio, se creará una lista de tipo RecyclerView para mostrar la lista de libros en el primer fragmento de la aplicación y una vista de tipo CoordinatorLayout para mostrar los detalles de los libros.

Primero, haz una copia del proyecto creado en el ejercicio anterior, ya que en los siguientes ejercicios trabajaremos sobre la estructura creada en el ejercicio 1 y al final de la práctica se entregarán dos proyectos: un proyecto con el código del ejercicio 1 y otro proyecto con el resto de ejercicios.

Abre el nuevo proyecto copiado en Android Studio para continuar.

Ahora vamos a crear el modelo del libro: crea un nuevo paquete con el nombre "modelo" y la clase "BookModel" con una lista estática de items de tipo "BookItem". Nuestros libros tendrán los siguientes detalles:

- Identificador (int)
- Título (String)
- Autor (String)
- Fecha de publicación (Date)
- Descripción (String)
- URL imagen de portada (String)

Actualizar el modelo "BookItem" con estos elementos.

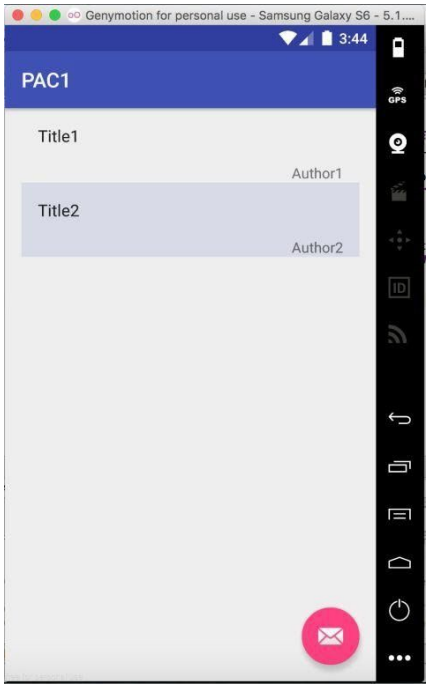
A continuación, crea un par de libros de prueba para poder ver los resultados:

```
/ **
* An array of sample book items.
* /
Public static final List <BookItem> ITEMS = new ArrayList <> ();

static {
BookItem book1 = new BookItem(0, "Title1", "Author1", new Date (), "Description", null);
BookItem book2 = new BookItem(1, "Title2", "Author2", new Date (), "Description2", null);
ITEMS.add (book1);
ITEMS.add (book2);
}
```

Una vez preparado el contenido de prueba, tendremos que crear un RecyclerView que mostrará la lista de libros.

En la actividad "BookListActivity" crearemos nuestro RecyclerView para mostrar nuestros libros.

Crea dos layout XML para mostrar los elementos de la lista pares e impares mostrando el siguiente diseño:



Crea un adaptador para la lista que extienda de:

RecyclerView.Adapter <SimpleItemRecyclerViewAdapter.ViewHolder>

En el método "onCreateViewHolder", carga un layout para los elementos pares de la lista y otro para los elementos impares, según el "viewType":

```
@Override
public ViewHolder onCreateViewHolder (ViewGroup parent, int viewType) {

View view = null;
// ============ INICIO CÓDIGO COMPLETAR ===============


// ============ FIN CÓDIGO A COMPLETAR =================
return new ViewHolder (view);
}
```

Agrega el método "getItemViewType" y devuelve el tipo par o impar según su posición:

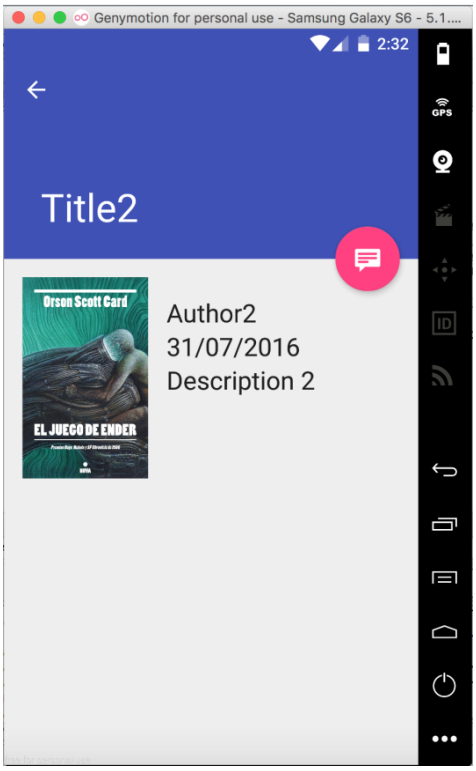
```
private final static int EVEN = 0;
private final static int ODD = 1;

@Override
public int getItemViewType(int position) {
int type;
// ============ INICIO CÓDIGO COMPLETAR ===============


// ============ FIN CÓDIGO A COMPLETAR =================
return type;
}
```

Una vez ya está la lista preparada, faltará modificar la pantalla del detalle. Como ahora no tenemos un identificador entre los parámetros de los libros, utilizaremos la posición en la lista para identificar el libro. En el método "onBindViewHolder", modifica el contenido para obtener la posición y pasarla a la clase del detalle:

```
@Override
public void onBindViewHolder(final ViewHolder holder, final int position) {
holder.mitema = mValues.get (position); holder.mTitleView.setText(mValues.get (position).title);
holder.mAuthorView.setText(mValues.get (position).author);

// ============ INICIO CÓDIGO COMPLETAR ===============
// Guarda la posición actual en la vista del holder
// ==== ======== FIN CÓDIGO COMPLETAR =================

holder.mView.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick (View v) {
int currentPos;
// ============ INICIO CÓDIGO COMPLETAR ======== =======
// Obtener la posición del libro donde se ha hecho click de la vista

// ============ FIN CÓDIGO COMPLETAR ========= ========
if (mTwoPane) {
//============ INICIO CÓDIGO COMPLETAR ===============
// Iniciar el fragmento correspondiente a tablet, enviando el argumento de la posición seleccionada
// ============ FIN CÓDIGO COMPLETAR ==================
} else {
// ============ INICIO CÓDIGO COMPLETAR ===============
// Iniciar la actividad correspondiente a móvil, enviando el argumento de la posición seleccionada
// ============ FIN CÓDIGO COMPLETAR ==================
}}});
}
```

Modifica los lugares donde se obtiene este argumento para obtener un "int" en vez de un "string".

Por último, modifica la vista del detalle para mostrar todos los detalles del libro: el autor, la fecha de publicación, la descripción y la imagen:

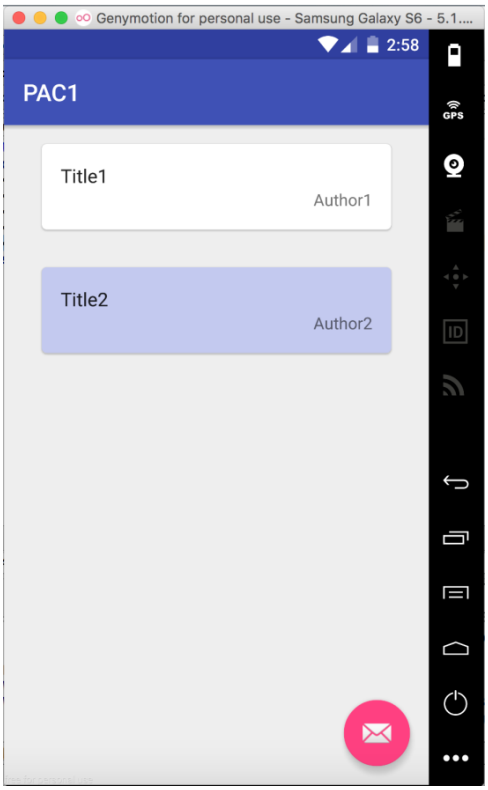


Como imagen del libro, de momento, utiliza una imagen estática.

Ejercicio 3: Implementar tarjetas en una lista

Modifica los elementos de la lista RecyclerView para que sean del tipo CardView.
Primero, incluye la librería de apoyo de Android que incluye las clases de CardView, para poder utilizarla en el proyecto.

Después modifica los dos archivos creados al ejercicio anterior (para los libros en posición par y los impares), para que utilicen una vista de tipo "CardView":

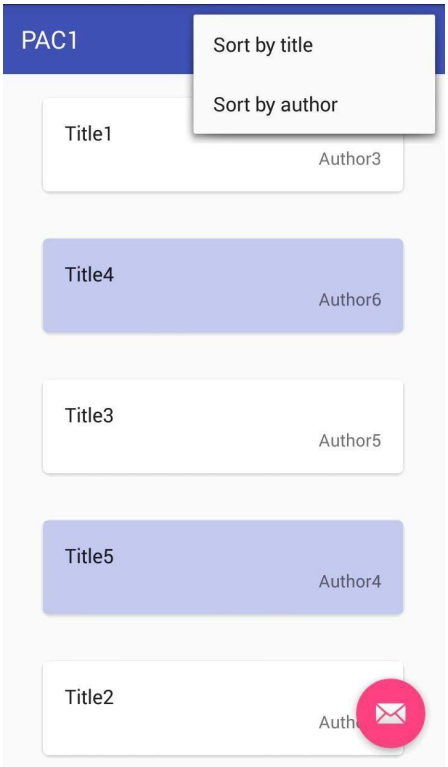


Cambia la elevación de la tarjeta a 2dp, un radio de la esquina de la tarjeta de 4dp y color de fondo blanco para las posiciones pares y "#C3C9EF" para las posiciones impares.

Ejercicio 4: Implementar métodos de ordenación

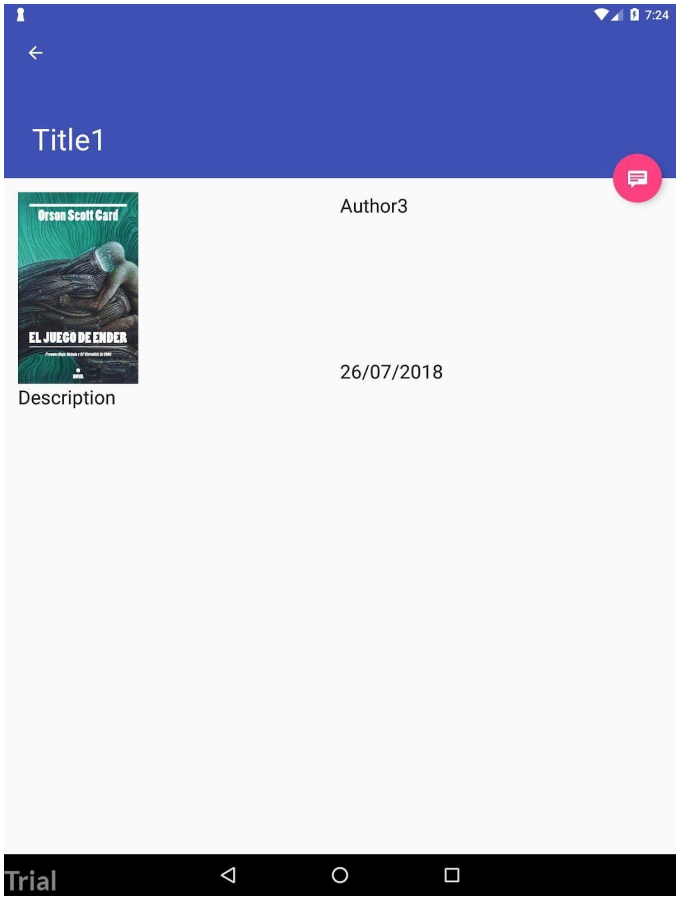
Implementa la ordenación de la lista y crea un menú en la vista con el método onCreateOptionsMenu. Agregar dos opciones en el menú que permitan ordenar la lista por título o autor.

```
@Override
public boolean onCreateOptionsMenu (Menu menu) {

MenuInflater inflater = getMenuInflater ();

inflater.inflate (R.menu.menu_list,menu);
return true;
}
```



Ejercicio 5: Modificar el detalle utilizando ConstraintLayout

Modifica el detalle del libro, utilizando una ConstraintLayout, para conseguir una estructura parecida a esta:



La descripción debe estar bajo la imagen. La parte de imagen y descripción deben ocupar un 50% de la pantalla. A la derecha, deben aparecer autor (en la parte superior), y fecha (en la parte inferior).

PARTE AVANZADA

Ejercicio 6: Implementar una lista con elementos de diferente tamaño

Modifica los elementos de la lista RecyclerView para mostrar una lista como la que se muestra a continuación:

El tamaño de los elementos debe ser dependiendo del tamaño de la imagen del libro y la última fila de libros debe tener la parte inferior descolorida para indicar al usuario que la lista es mayor. Aplica un degradado a las imágenes para que sea más fácil leer el texto.
Ver este tipo de lista solo en teléfono y añade más libros en la lista para poder ver el efecto.

** Agregar varios libros de prueba con imágenes de diferentes tamaños para comprobarlo.*

PARTE TEÓRICA

Ejercicio 7: Java o Kotlin

Explica por qué elegir Java o Kotlin para empezar un proyecto nuevo a día de hoy. Ventajas y posibles inconvenientes de cada lenguaje.

Recursos

Básicos

- Módulo 1: Estructura de una aplicación de Android.
- Módulo 2: Controladores: actividades, fragmentos y servicios.
- Módulo 3: Interfaz gráfica: layouts.

Complementarios

- Documentación oficial de Android.

Criterios de valoración

- La PEC debe resolverse de forma individual.
- El código fuente debe comentarse y el estilo de programación debe ser limpio y pulido.
- La entrega es única, no se han de entregar los ejercicios por separado.
- Cada ejercicio tiene el siguiente peso:

PARTE PRÁCTICA

- Ejercicio 1: 25%
- Ejercicio 2: 25%
- Ejercicio 3: 10%
- Ejercicio 4: 10%
- Ejercicio 5: 10%

PARTE AVANZADA

- Ejercicio 6: 10%

PARTE TEÓRICA

- Ejercicio 7: 10%

Formato y fecha de entrega

Se debe entregar un archivo ZIP con nombre PAC1_Apellido1Apellido2Nombre.zip con los siguientes archivos:

- Aplicación en formato APK del ejercicio 1.
- Aplicación en formato APK del proyecto realizado en los ejercicios 2 al 6.

- Archivo .txt con nombre PAC1_Apellido1Apellido2Nombre.txt que contenga **únicamente** la **URL** de tu proyecto en GitHub.
    - Si el repositorio es privado añadir previamente aqueudot@uoc.edu con permiso de lectura.
    - Utiliza una nueva rama "ejercicio-1" para el 1er ejercicio.
    - Usa la rama "master" para los ejercicios 2 al 6.
- Documento con la respuesta a las preguntas teóricas.
- Código fuente de todo el proyecto del ejercicio 1.
- Código fuente de todo el proyecto de los ejercicios 2 al 6.

Estos archivos deben entregarse en el espacio Entrega y registro de actividades del aula **antes de las 23:59 del día marcado en el aula**. No se aceptarán entregas fuera de plazo.