Desafío - MonsterCreator App

- Para realizar este desafío debes haber estudiado previamente todo el material disponibilizado correspondiente a la unidad.
- Una vez terminado el desafío, comprime la carpeta y sube el `.zip`

Descripción

A continuación vamos a poner a prueba tus conocimientos en la construcción de una aplicación utilizando un patrón de arquitectura MVVM, Room, Livedata y también Databinding.

La aplicación MonsterCreator está a medio terminar, el desarrollador anterior abandonó el proyecto y no dejo documentación donde indique en que estado esta.

La idea principal de MonsterCreator es ser una aplicación que permite obtener un listado de monstruos creados. Mostrando una imagen, nombre y monsterPoints. En la MainActivity se mostrará a través de un RecyclerView esta información (ver imagen de referencia)

Este Desafío solo te pedirá que muestres los datos provenientes de Room en la UI indicada. Utilizando para esta Tarea LiveData y usando un ViewModel, deberás crear datos programáticamente para probarlo.

Instrucciones

Para la realización de este desafío, debes descargar el .zip que se encuentra en plataforma, llamado Apoyo Desafío - MonsterCreator App donde encontrarás, el material base para poder desarrollar los siguientes puntos:

1. Tu primera tarea es revisar el código facilitado, Revisar las clases, métodos, layouts etc. Verificar si el proyecto puedes ejecutarlo en tu máquina, resolver los posibles problemas de compatibilidad y descargar las versiones más nuevas de las librerías utilizadas.
2. Crear el adaptador para el RecyclerView que mostrará todos los monstruos.

   Utiliza las vistas ya existentes para los items, Recuerda que también debes implementarlo en MainActivity para recibir los datos en el recyclerView que los mostrara.

   TIP: ya están creados los layouts para los monstruos (monster_item.xml), Pero no está añadido el RecyclerView en “content_main.xml”
3. Debes construir un repositorio que se encargue de comunicarse con las distintas fuentes de datos, , este nos ayudará a tener una sola fuente de verdad para los datos. Ya está creada la clase “MonsterRepository” completa el código que haga falta, no olvides revisar si tu DAO está correctamente configurado.

   TIP: Como posiblemente tendremos más de un ViewModel en el proyecto, es conveniente crear una interface para manejar los métodos que utilizaremos, de esta forma luego solo implementaremos esta interface en la clase repositorio y nos servirá como puente hacia este desde el viewModel.

```
interface MonsterRepositoryInterface {
   fun saveMonster(monster: Monster)
   fun getAllMonsters(): LiveData<List<Monster>>
   fun clearAllMonsters()
}
```

> No olvides utilizar algún método asíncrono para las tareas de escritura y borrado de la base de datos.

4. Debes crear el ViewModel que se encargará de manejar los datos provenientes de el Repositorio en la vista, para ello modifica el código en la clase **AllMonsterViewModel**

   TIP: Te recomiendo utilizar la interface sugerida, debería debería verse así:

```
class AllMonsterViewModel(

   private val monsterRepository: MonsterRepositoryInterface =
MonsterRepository() ) : ViewModel() {

   private val allMonsterLiveData = monsterRepository.getAllMonsters()

   fun getAllMonsterLiveData(): LiveData<List<Monster>> = allMonsterLiveData

   fun clearAllCreatures() = monsterRepository.clearAllMonsters()
}
```

5. Ahora debemos añadir el viewModel desde el cual nuestra MainActivity consumirá los datos. Realizar las configuraciones y escribir el código necesario para observar los datos utilizando LiveData, No olvides pasar los datos observados al adaptador.
6. Utiliza el boton del menu superior, para eliminar todos los datos, tambien cambiale el nombre que viene por defecto(“settings”) a (“borrar todos”), esto lo debes hacer refiriendo al método del ViewModel que realiza esta función .
7. Este paso es opcional y servirá para probar si funciona lo que hemos realizado , consiste en añadir las siguientes líneas de código en MainActity.

```
private val monsterRepository: MonsterRepositoryInterface =
MonsterRepository()
```

a. Luego vamos añadir lo siguiente:

```
val newMonster = MonsterGenerator()
monsterRepository.saveMonster(newMonster.generateMonster(MonsterAttributes(10,
10,10 ),"hola" , 0))
```

Si recargamos la aplicación tendremos al menos 1 elemento en nuestro listado, aun sin imagen, pero sabremos si funciona.

Recuerda eliminar este código antes de enviar el Desafío.

> Recuerda que el objetivo es mostrar un listado de Monstruos, como aun no crearemos el código necesario para que el usuario los cree, debes añadir algunos programáticamente para revisar el funcionamiento de LiveData.

Imagen de referencia: