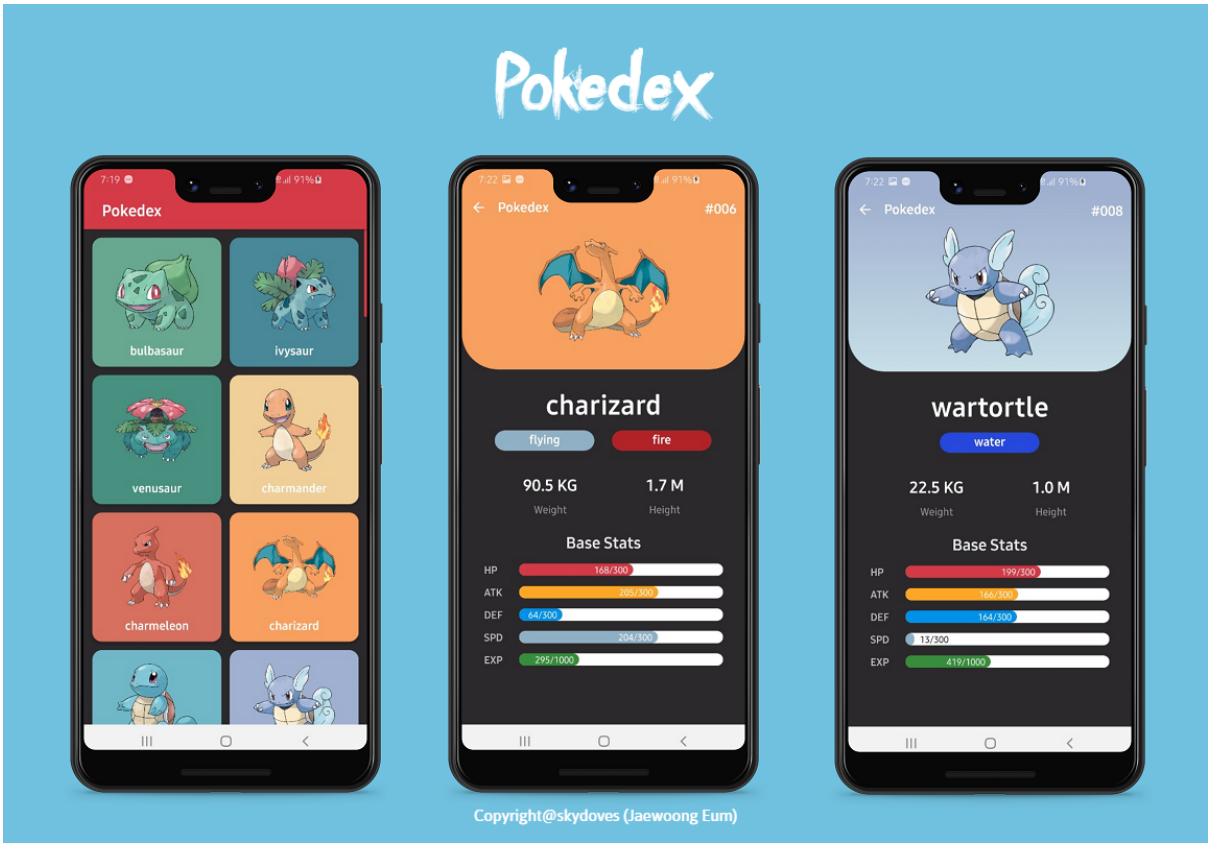
Proyecto Pokédex

El proyecto consistirá en la realización de una aplicación móvil en Android que sea similar a una Pokédex. Contendrá información de Pokémon, bayas, movimientos, entre otros.



Requisitos

- La aplicación debe visualizarse correctamente en diferentes dispositivos móviles (tablets y teléfonos con diferentes resoluciones, ratios y orientaciones de pantalla).
- La aplicación debe contener un listado con todos los Pokémon.
- La aplicación debe contener una forma de acceder al detalle o información adicional de cada Pokémon.
- La aplicación usará la API PokeAPI. Debe llamar como mínimo a 3 endpoints de la API.
- La aplicación debe contener al menos un menú.
- La aplicación debe contener al menos 4 actividades / fragments diferentes.
- La aplicación debe usar el patrón MVVM.

Ideas

- Puedes hacer que vayan apareciendo Pokémon (periódicamente, cuando el usuario pulse un botón, como quieras) y se unan a tu equipo. Serán de una especie, pero tendrán pequeñas variantes en los atributos. Estos Pokémon deberán guardarse en local (SQLITE, ficheros JSON, etc.). Por ejemplo tendrás la especie magikarp y puedes tener varios magikarp en tu equipo.
- Puedes tratar de simular un combate Pokémon e integrarlo en la aplicación (con Pokémon base o con los Pokémon creados).
- Puedes tratar de modificar la interfaz según el tipo del Pokémon.

Recursos

API
https://pokeapi.co/
Retrofit
https://square.github.io/retrofit/
GSON
https://github.com/google/gson
Glide (Librería para cargar imágenes)
https://github.com/bumptech/glide

Ejemplo de aplicación Pokédex en la PlayStore:
https://play.google.com/store/apps/details?id=com.talzz.datadex&hl=en&gl=US&pli=1

Ejemplo de aplicación Pokédex en Github:
https://github.com/skydoves/Pokedex

Implementación de Pokédex con Retrofit, GSON y Glide:
https://github.com/alvareztech/Pokedex

Implementación de Pokédex avanzada y tutorial:
https://morioh.com/p/4c3a1ed79519
https://github.com/philipplackner/JetpackComposePokedex/tree/Part2-RetrofitSetup?ref=morioh.com&utm_source=morioh.com

RÚBRICA DE EVALUACIÓN DEL PROYECTO:

Criterio	Excelente	Bueno	Regular	Deficiente
Diseño de la interfaz.	La interfaz se visualiza correctamente en móviles y tablets con diferentes resoluciones y orientaciones de pantalla. Y además es original y visualmente atractiva. Hay muchos elementos multimedia.	La interfaz se visualiza correctamente en móviles y tablets con diferentes resoluciones y orientaciones de pantalla. Y además es visualmente agradable / atractiva. Hay algún elemento multimedia.	La interfaz no se visualiza correctamente en móvil o tablet con diferentes resoluciones de pantalla.	La interfaz no se visualiza correctamente en móvil y / o tablet.
Estructura de la interfaz.	Hay varias actividades y dentro de ellas diferentes fragments. Hay diferentes tipos de menús que funcionan correctamente. Se usan RecyclerViews correctamente.	Hay varias actividades y dentro de ellas diferentes fragments. Hay algún menú que funciona correctamente. Se usan RecyclerViews correctamente.	Hay varias actividades y fragments. Y hay algún menú que funciona correctamente.	Hay una única actividad sin fragments. Hay algún menú pero con errores.
Arquitectura de la aplicación.	Hay separación de código en modelos, vistas y viewmodels y la información se mantiene con LiveData.	El código está correctamente dividido: hay modelos, vistas y viewmodels.	El código está separado en modelos y vistas.	Casi todo el código está en las activities.
APIs y persistencia	Se usan APIs para recuperar información y se recuperan datos avanzados (datos relacionados, imágenes, etc.) y BB.DD o ficheros para guardar información local o configuraciones.	Se usan APIs para recuperar información y se recuperan datos avanzados (datos relacionados, imágenes, etc.).	Se usan APIs o BB.DD. para almacenar o recuperar información.	No se usan APIs o BB.DD. para almacenar o recuperar información.
Innovación y originalidad.	Desarrolla varias funcionalidades adicionales. Utiliza varias herramientas y tecnologías no vistas en clase (notificaciones, ficheros, etc.).	Desarrolla varias funcionalidades adicionales. Y utiliza alguna herramienta no vista en clase.	Desarrolla alguna funcionalidad adicional.	La aplicación cumple los mínimos.
Calidad de la aplicación.	Usa nombres relevantes, no se repite código, mantiene la convención de código para Java. Hace algún tipo de testing.	Usa nombres relevantes, no se repite código, mantiene la convención de código para Java.	Mantiene la convención de código para Java.	No mantiene la convención de código para Java.