UNIVERSIDAD DON BOSCO

Desarrollo de Software para Android

Patron MVVM – DSA441

Kenya Elizabeth Parada Palma PP220664

William Ernesto Ramos Valladarres RV220068

Javier Eliseo Gutiérrez Flores GF220089

Axel Giovanni Ramirez Alfaro - RA160395

DOCENTE:

Alexander Alberto Sigüenza Campos

INDICE

Introducción----------3

- Qué es el patrón MVVM----------4
- Cuáles son sus componentes principales

  y cómo se relacionan entre sí----------4-5

- Cómo se aplica el patrón MVVM
  en Android con Kotlin----------5

- Cuáles son las ventajas y
  desventajas de utilizar el patrón MVVM
  en el desarrollo de aplicaciones móviles----------5-6

Bobliografia----------7

INTRODUCCION

El patron MVVM (Model View ModelVIew) tiene el objetivo para llevar a cabo la separación del apartado de la interfaz de usuario (*View*) de la parte lógica (*Model*). Esto lo hace con el objetivo de que el aspecto visual sea completamente independiente.

- ¿Qué es el patrón MVVM?

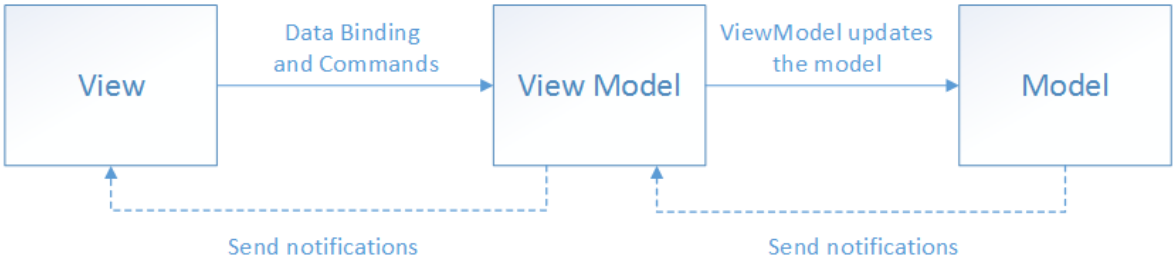
Los patrones están diseñados para proveer marcos que agilizar tanto el entendimiento así como la codificación, todos tienen la misma función y propósito cambian entre ellos los componentes a diseñar pero la lógica sigue siendo igual,

En sí el patrón Model-View-ViewModel (MVVM) ayuda a separar limpiamente la lógica de presentación y negocios de una aplicación de su interfaz de usuario (UI). Esto ayuda a abordar diversas problemáticas y facilita la prueba, mantenimiento y su evolución, así como puede ayudar a la reutilización de código para otras funciones.

Con el patrón MVVM la interfaz y la presentación subyacen y la lógica de negocia es separada en 3 clases independiente: la vista, la interfaz, el modelo de vista, y el modelo.

- ¿Cuáles son sus componentes principales y cómo se relacionan entre sí?

Hay tres componentes principales en el patrón MVVM: el modelo, la vista y el modelo de vista.



La vista conoce el modelo de la vista y el modelo de la vista conoce el modelo, pero el modelo no es consciente del modelo de vista y el modelo de vista no es consciente de la vista, por lo tanto el modelo de vista aísla la vista del modelo y permite que el modelo evolucione independientemente de la vista.

Osea que existe una relación de independencia entre View y el Viewmodel, pero el View Model funciona como una especie de filtro aislando a Viewmodel. Esto nos indica que el Viewmodel puede cambiar y mejorar sin que la View esté presente siempre.

- ¿Cómo se aplica el patrón MVVM en Android con Kotlin?

Para implementar el patrón MVVM en Android se deben seguir los siguientes pasos:

1. Crear un nuevo proyecto
2. Crear la clase Modelo
3. Establecer todos lo que se va a utilizar en la app en la activity_main.xml
4. Crear la clase ModelView
5. Definir el view en el MainActivity
6. Ejecutar la app

- ¿Cuáles son las ventajas y desventajas de utilizar el patrón MVVM en el desarrollo de aplicaciones móviles?

  Ventajas
  - Simplifica las pruebas unitarias
  - Los cambios aislados son menos arriesgados y es más fácil experimentar con ellos.
  - Mejora el mantenimiento a largo plazo.
  - Admite el trabajo en equipo, unir códigos se vuelve más fácil.

Desventajas

- Para poder programar en este patrón se debe tener un conocimiento previo.
- Se deben utilizar un mayor número de ficheros.
- Se debe utilizar una estructura predefinida.

Bibliografía

Model-View-ViewModel (MVVM)
https://learn.microsoft.com/es-es/dotnet/architecture/maui/mvvm
https://keepcoding.io/blog/pasos-para-la-implementacion-de-mvvm-en-android/#Pasos_para_la_implementacion_de_MVVM_en_Android