Autor: María José Villafuerte 22129	**Docente:** Cecilia Castillo
Sección: 21	**Fecha:** 09/10

Laboratorio 8

Ventajas de usar Coroutines:

1. Código Sincrónico: Esto hace que el código sea más legible y fácil de entender, ya que no tienes que lidiar con callbacks anidados.
2. Menos Anidación: Las Coroutines reducen la anidación de código en comparación con los callbacks anidados, lo que hace que el código sea más limpio y menos propenso a errores.
3. Manejo de Errores Mejorado: El manejo de errores con Coroutines es más parecido al manejo de excepciones en código síncrono, lo que facilita la captura y el manejo de errores.
4. Sintaxis más Limpia: La sintaxis para lanzar y esperar el resultado de una Coroutine es más limpia y concisa en comparación con los callbacks.
5. Menos Consumo de Memoria: Las Coroutines pueden ser más eficientes en cuanto al uso de memoria en comparación con callbacks, especialmente cuando se trata de muchas llamadas asíncronas.
6. Soporte Nativo para Cancelación: Coroutines proporciona soporte nativo para cancelar tareas asíncronas. Esto es útil para liberar recursos y evitar fugas de memoria.
7. Compatibilidad con Librerías Modernas: Muchas bibliotecas y API modernas están diseñadas para trabajar bien con Coroutines, lo que facilita la integración de estas tecnologías en tu aplicación.
8. Coroutines Scope: Puedes definir ámbitos (scopes) de Coroutines para manejar automáticamente la cancelación de tareas cuando una actividad o fragmento se destruye, lo que ayuda a evitar problemas de fugas de memoria.
9. Facilita la Programación Concurrente: Las Coroutines facilitan la programación concurrente, lo que permite realizar múltiples tareas de manera eficiente sin bloquear el hilo principal.

Ventajas de usar Callbacks:

1. Compatibilidad con Código Existente: Si ya tienes una base de código que utiliza callbacks, puede ser más fácil mantener la compatibilidad en lugar de migrar completamente a Coroutines.
2. Control Fino: Los callbacks pueden proporcionar un mayor control sobre el flujo de datos y eventos en situaciones específicas.
3. Bajo Overhead: En algunas situaciones, los callbacks pueden tener un menor overhead en comparación con Coroutines, especialmente para tareas muy simples.

Video:

Link: https://youtu.be/WLKsfnCgk80

Git:

Link: https://github.com/Maria-Villafuerte/Lab_8_Movil