Universidad Nacional Autónoma de México

Facultad de Contaduría y Administración

Sistema Universidad Abierta y Educación a Distancia

Alumno: López Acevedo Víctor Rafael

Grupo: 9696

Materia: INFORMATICA VI

Unidad: 4

Actividad: M4-01

Fecha: 18 de abril 2025

Actividad: Ciclo de vida de una activity

Objetivo del aprendizaje: Comprender el ciclo de vida de una actividad.

- Ciclo de vida de la actividad

Cuando un usuario navega por tu app, sale de ella y vuelve a entrar, las instancias de Activity de tu app pasan por diferentes estados de su ciclo de vida. La clase Activity proporciona una serie de devoluciones de llamada. que informan a la actividad cuando cambia un estado o que el sistema crea, detiene o reanuda una actividad, o destruye el proceso en el que reside la actividad.

Dentro de los métodos de devolución de llamada de ciclo de vida, puedes declarar el comportamiento que tendrá tu actividad cuando el usuario la abandone y la reanude. Por ejemplo, si estás crear un reproductor de video en streaming, puedes pausar el video y cancelar la conexión de red cuando el usuario cambia a otra aplicación. Cuando el usuario regresa, puedes volver a conectarte a la red y permitir que el usuario reanude el video desde mismo lugar.

Cada devolución de llamada te permite realizar un trabajo específico apropiado para un cambio de estado determinado. Hacer el trabajo preciso en el momento adecuado y administrar las transiciones correctamente hace que tu app sea más sólida y eficiente. Por ejemplo, una buena implementación de las devoluciones de llamada de ciclo de vida puede ayudar a que tu app evita lo siguiente:

- No falle si el usuario recibe una llamada telefónica o cambia a otra app mientras usa la tuya.
- No consuma recursos valiosos del sistema cuando el usuario no la use de forma activa.
- No pierda el progreso del usuario si este abandona tu app y regresa a ella posteriormente.
- No falle ni pierda el progreso del usuario cuando se gire la pantalla entre la orientación horizontal y la vertical.

- Conceptos de los ciclos de vida de las actividades

Para navegar por las transiciones entre las etapas del ciclo de vida de la actividad, la La clase Activity proporciona un conjunto básico de seis devoluciones de llamada: onCreate(), onStart(), onResume(), onPause(), onStop() y onDestroy() El sistema invoca cada una de estas devoluciones de llamada cuando la actividad entra en un nuevo estado.

Cuando el usuario comienza a abandonar la actividad, el sistema llama a métodos para desmantelarla. En algunos casos, la actividad solo se y aún reside en la memoria, como cuando el usuario cambia a otra app. En estos casos, la actividad aún puede volver al primer plano.

Si el usuario vuelve a la actividad, se reanuda desde donde el usuario la dejó.

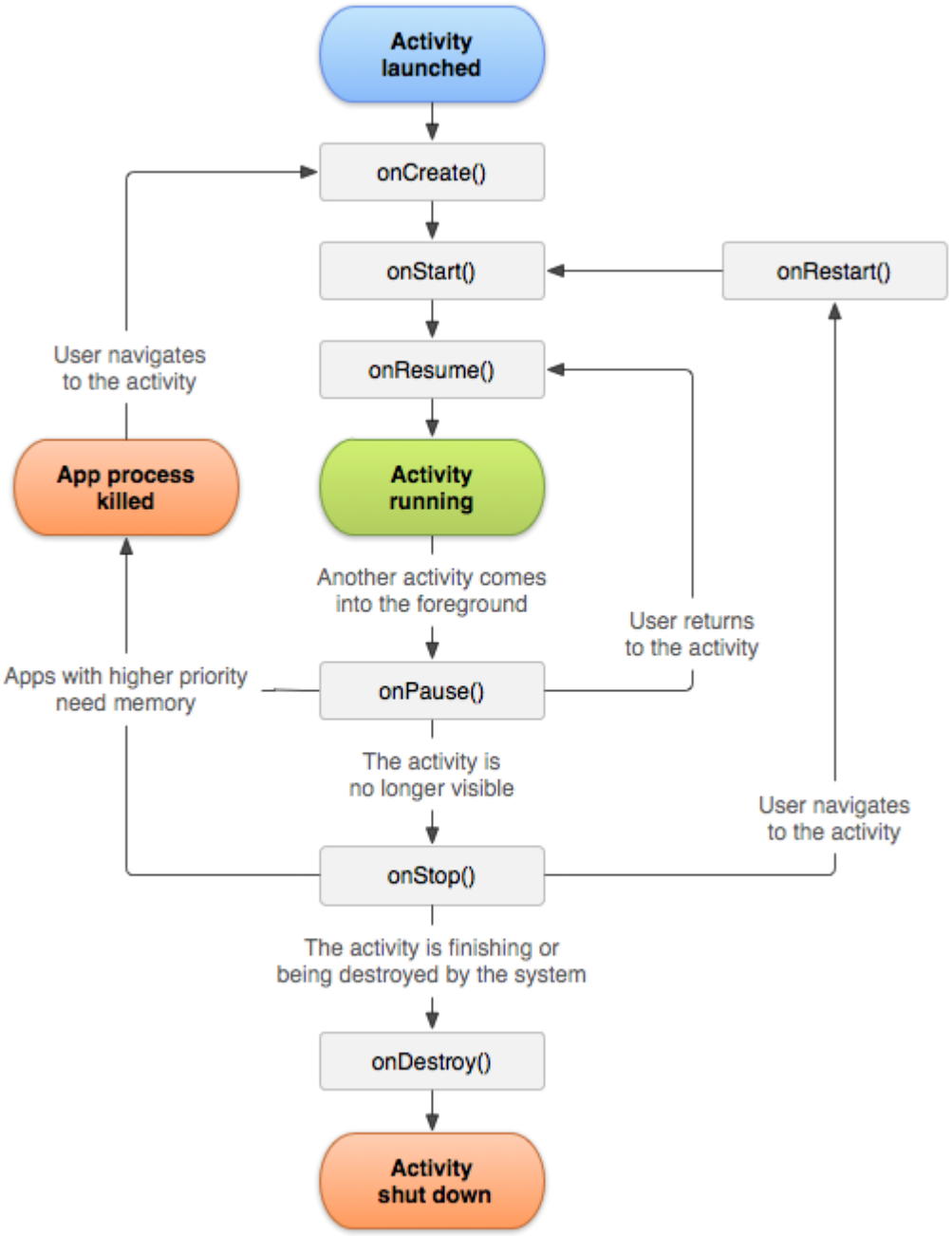


Ilustración simplificada del ciclo de vida de una actividad

- Devoluciones de llamada del ciclo de vida

onCreate()

El método onCreate() se ejecuta cuando la actividad es creada por primera vez, y es el punto de partida para inicializar todos los elementos esenciales. Aquí se

configuran componentes clave como vistas, listas, datos iniciales, variables y la asociación con un ViewModel. También se recibe el parámetro savedInstanceState, que permite recuperar el estado anterior de la actividad si es que existía, aunque si se trata de una nueva instancia, este será nulo. Además, si existen componentes que observan el ciclo de vida de la actividad, también son notificados en este momento a través del evento ON_CREATE, lo que permite realizar tareas específicas para esta fase inicial.

onStart()

Luego sigue onStart(), que es invocado cuando la actividad se está volviendo visible al usuario. Aunque aún no está en primer plano, es en esta etapa donde se prepara la interfaz de usuario, como inicializar vistas o elementos visuales que aparecerán en pantalla. Es un paso breve y transitorio, ya que inmediatamente después, el sistema continúa con la siguiente fase del ciclo.

onResume()

El método onResume() es el que marca la entrada de la actividad al primer plano y su estado completamente interactivo. Aquí es cuando la aplicación ya puede recibir y manejar interacciones del usuario. La actividad se mantiene en este estado hasta que algo ocurre que la interrumpe, como una llamada telefónica, la apertura de otra aplicación o el apagado de la pantalla. Si eso pasa, se invoca onPause(). Siempre que la actividad retoma la atención del usuario después de una pausa, se vuelve a ejecutar onResume(), por lo que este método también sirve para reinicializar todo lo que fue detenido anteriormente, especialmente elementos que deben estar activos mientras la app esté visible.

onPause()

Por su parte, onPause() se ejecuta cuando la actividad deja de estar en primer plano, aunque aún puede estar parcialmente visible, como ocurre en el modo multiventana o cuando aparece un diálogo encima. Este método permite pausar tareas que no deben ejecutarse si la actividad no está activa, como detener sensores, liberar recursos del sistema o pausar vistas previas. Aunque no siempre significa que la actividad va a cerrarse, sí es una advertencia de que el usuario se está alejando, por lo que se deben hacer ajustes temporales. Sin embargo, si se requiere una liberación de recursos más definitiva, es más adecuado hacerlo en onStop().

onStop()

Cuando la actividad ya no es visible, se llama al método onStop(). Esto sucede si otra actividad cubre por completo la pantalla o si la aplicación está siendo cerrada. En este momento se deben liberar todos los recursos que ya no son necesarios, detener animaciones, reducir el nivel de actualizaciones de ubicación y realizar tareas que requieran más procesamiento, como guardar datos en una base de

datos. Además, como en el modo multiventana la actividad puede seguir siendo visible mientras está pausada, onStop() es más adecuado que onPause() para liberar recursos relacionados con la interfaz.

onDestroy()

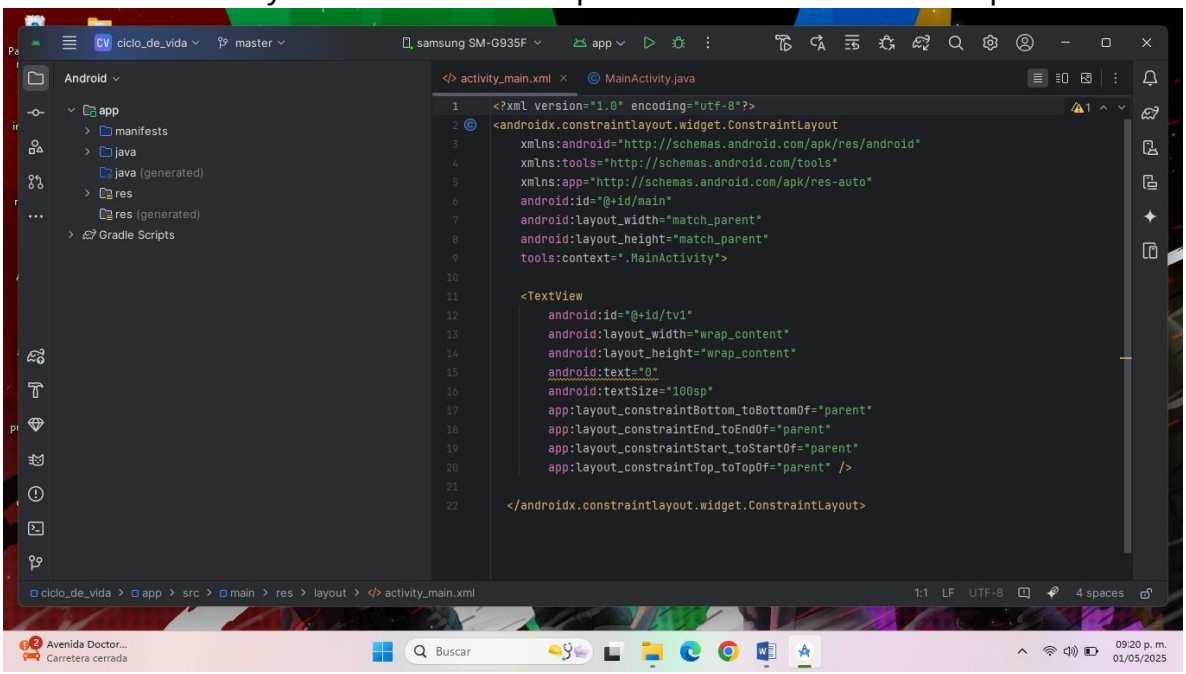
Finalmente, el método onDestroy() se ejecuta justo antes de que la actividad sea destruida, ya sea porque el usuario la ha cerrado por completo o porque el sistema necesita reiniciarla por un cambio de configuración, como una rotación de pantalla. Este método sirve para limpiar cualquier recurso que no haya sido liberado anteriormente. Si la destrucción es temporal, el sistema creará de inmediato una nueva instancia de la actividad y volverá a iniciar su ciclo con onCreate(). Para manejar estas situaciones sin pérdida de datos, se recomienda usar un ViewModel, el cual persiste entre recreaciones y puede limpiar sus propios datos si detecta que la actividad ya no se va a utilizar mediante el método onCleared() y comprobando si la actividad está finalizando con isFinishing().

Este flujo de métodos permite gestionar de forma eficiente el comportamiento de una actividad según su visibilidad y el uso que el usuario esté haciendo de la app en cada momento.

1. Crea una activity simple en Android Studio en lenguaje de programación Java.

Iniciamos Android studio y creamos una nueva aplicación

Dentro de la aplicación ya tendremos un cuadro de texto el cual modificamos para crear un contador y ver como actúa la aplicación en el ciclo de vida que este tiene

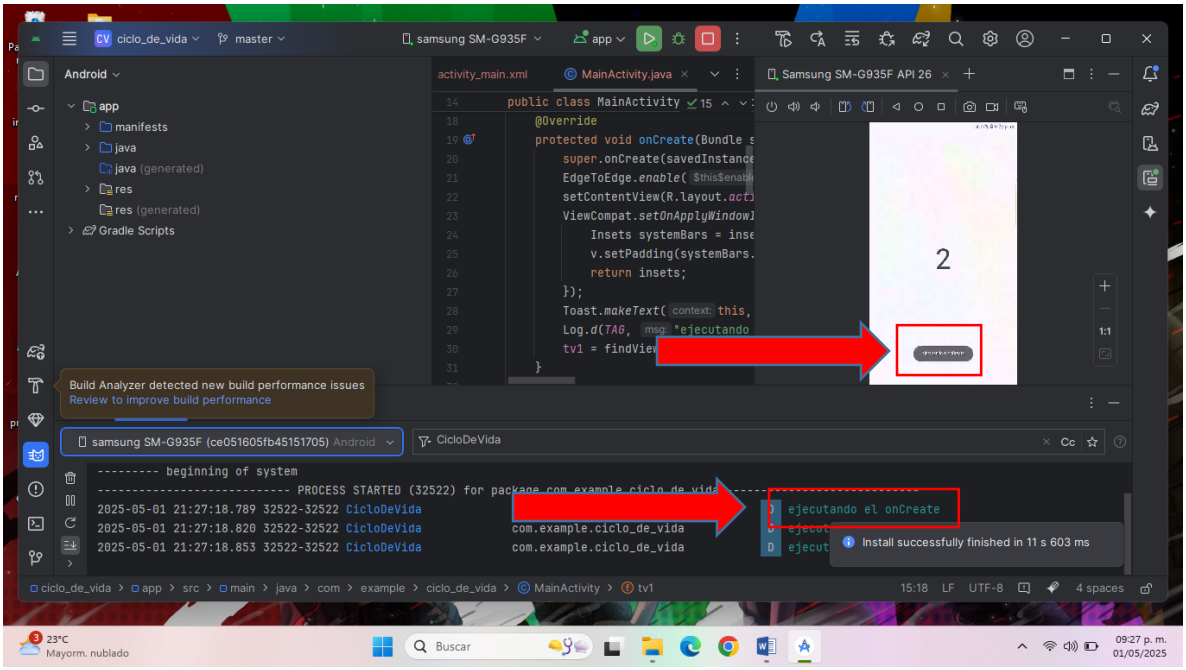


Este es el código que se usara en la vista de la aplicación

2. Imprimir un mensaje personalizado en cada uno de los métodos del ciclo de vida de una actividad.

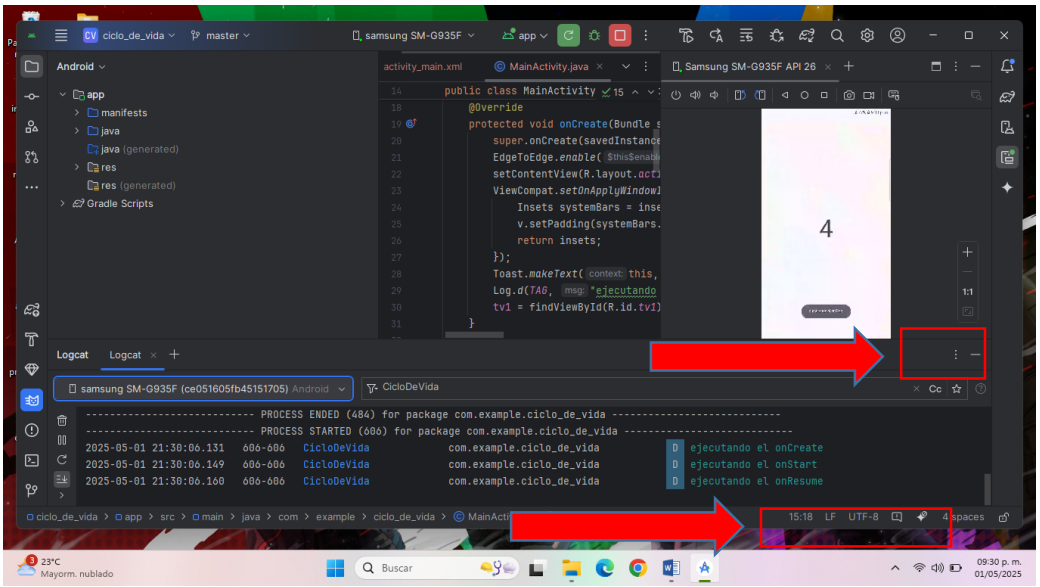
A continuación, se mostrarán las capturas de pantalla de cada uno de los mensajes que se piden y al final se pondrá el código que se usó para crear la aplicación

- **onCreate()**



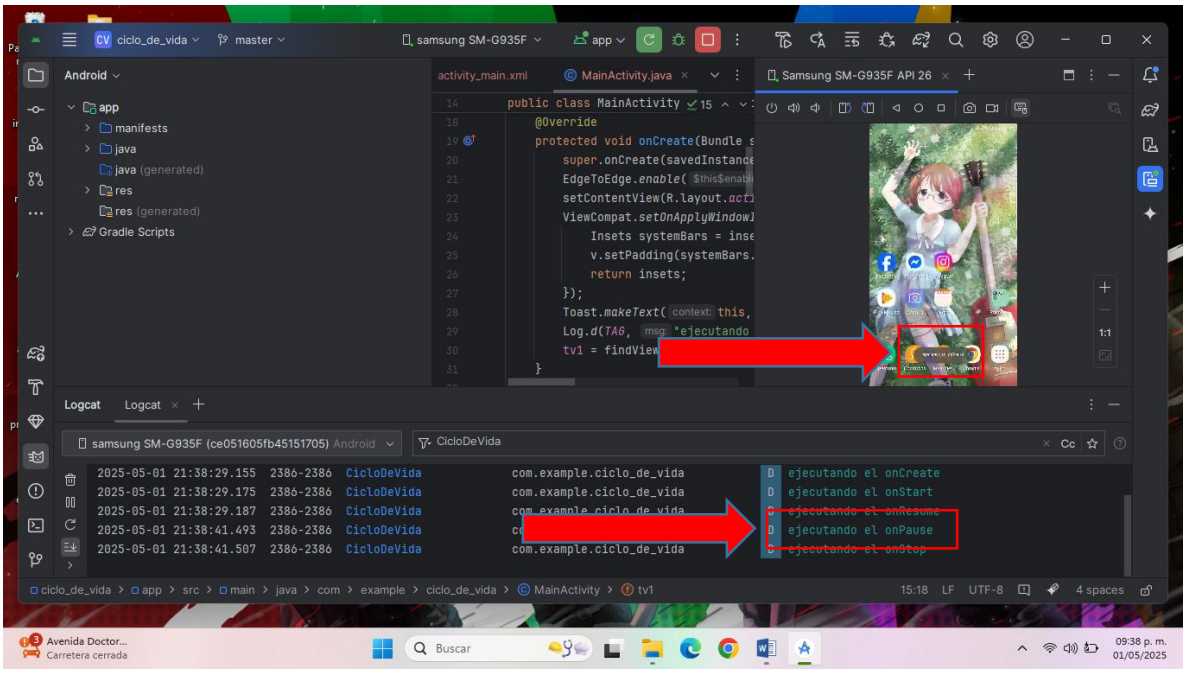
aquí se muestra el on créate, tanto en un toast como en el logcat

- **onStart()**



aquí vemos igual como pasa del estado oncreate al estado onstart, de igual modo se muestra con un toast y en el logcat el mensaje de que se pasó a ese estado

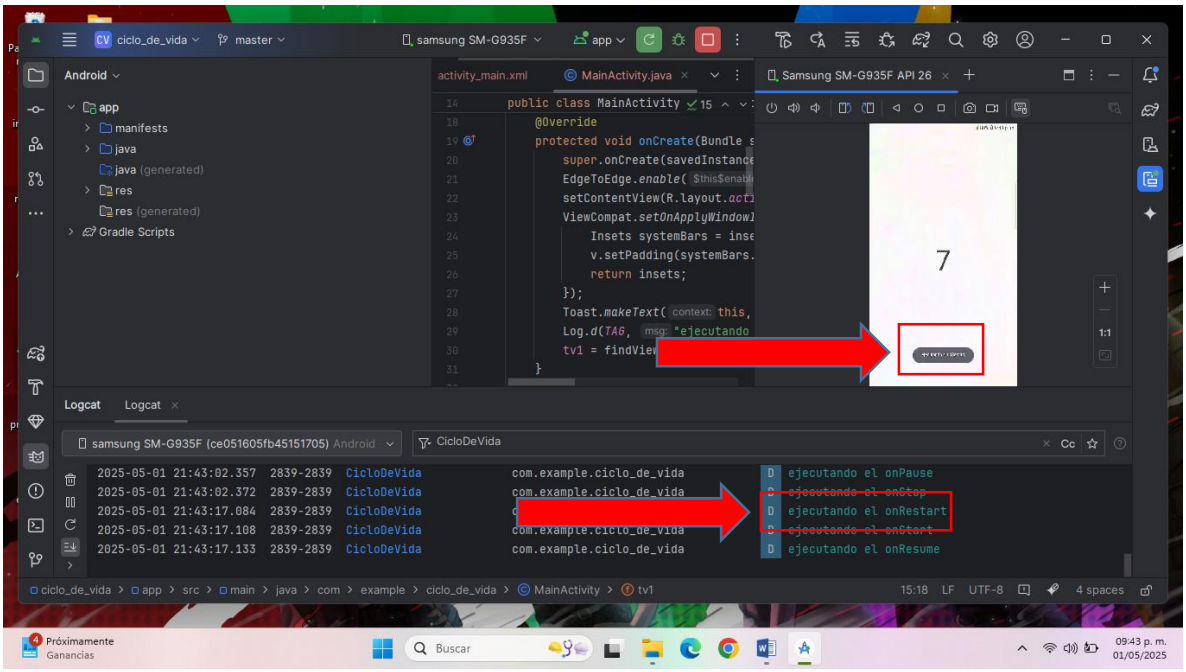
- **onPause()**



aquí vemos que si nos salimos de la alicacion sin cerrarla pasamos al estado onpause, en este estado la aplicación pasa a segundo plano y sigue activa

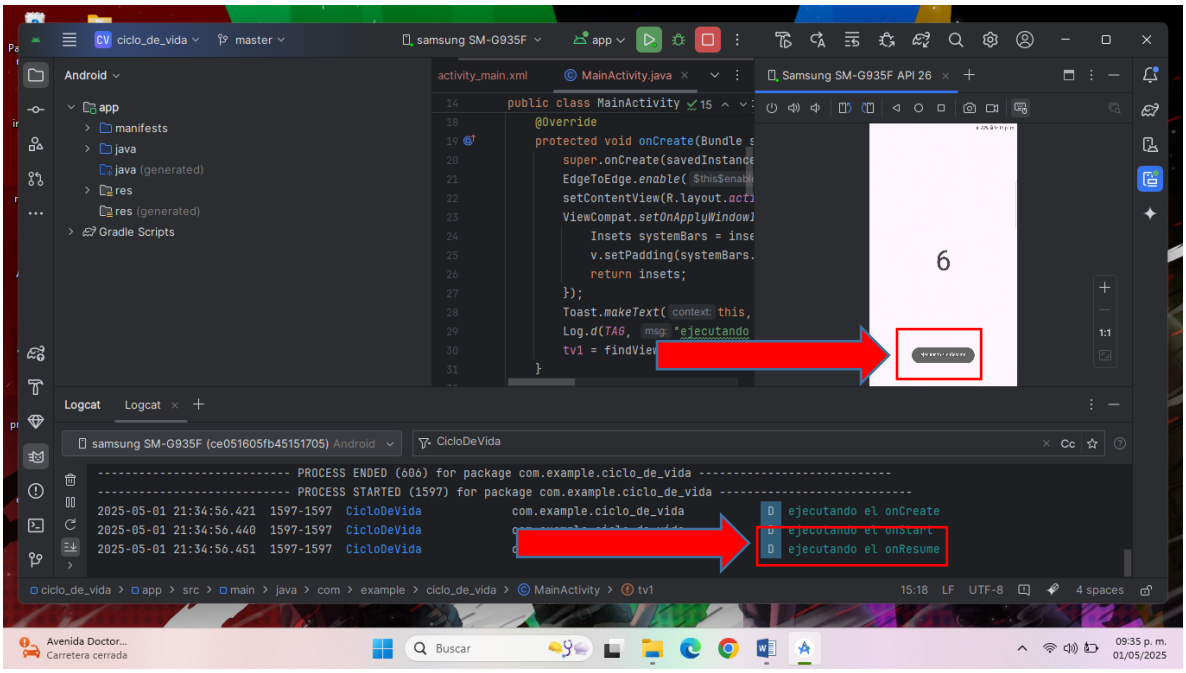
También podemos ver los mensajes que se generan mediante el toast y el logcat

- **onRestart()**



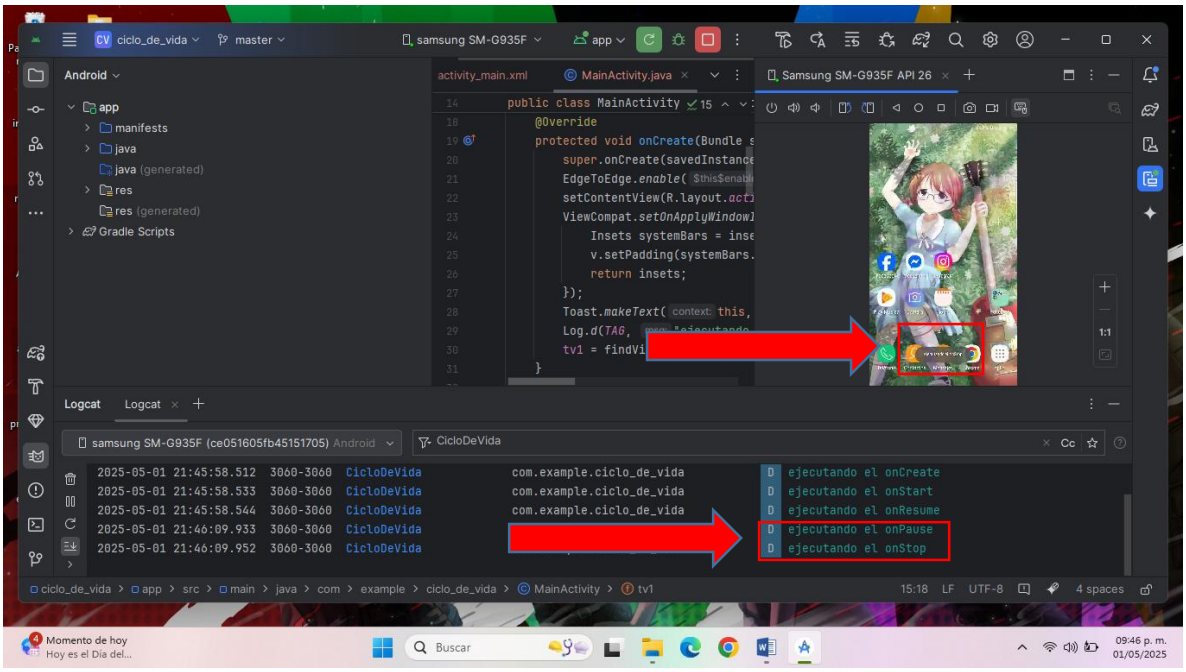
regresamos a la aplicación de nuevo y podemos ver que el contador reinicia su conteo desde donde se quedó, e igual podemos ver los mensajes en ambos modos

- **onResume()**



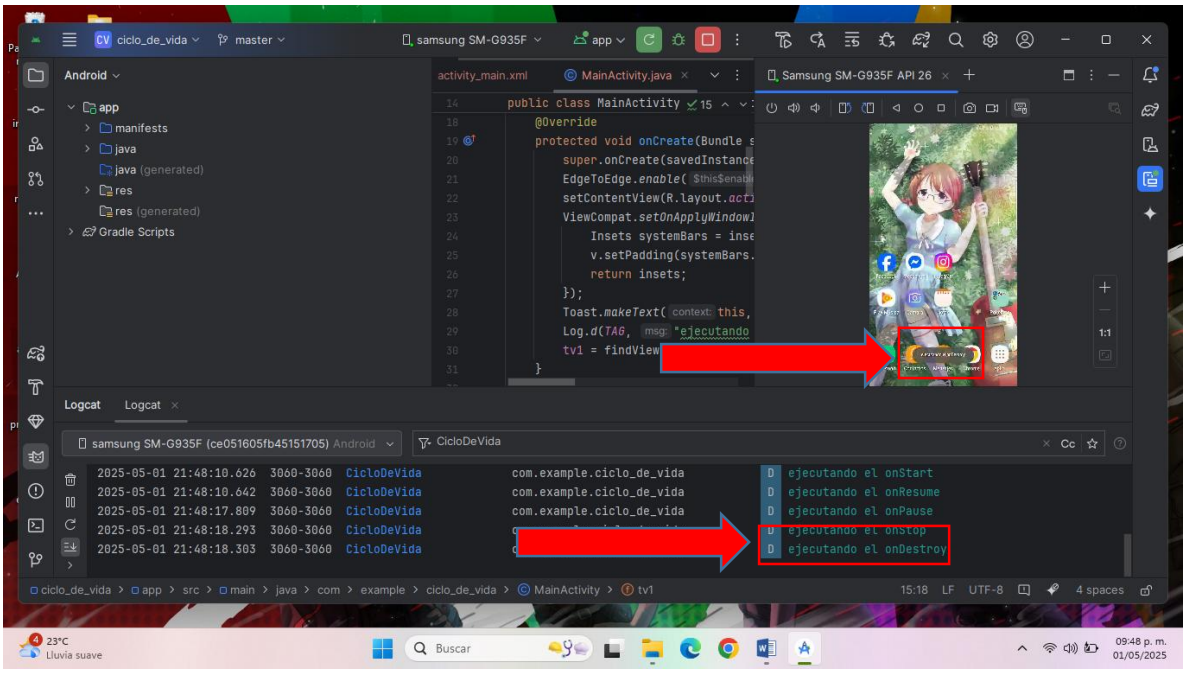
aquí pasamos al onresume

- **onStop()**



En esta parte de onstop es cuando la aplicación se detiene y deja de consumir los recursos

- **onDestroy()**



Una vez nos salimos de la aplicación y se cierra pasamos al estado ondestroy, que realiza el cierre definitivo y corta el uso de los recursos para que estén disponibles para otras actividades

Bibliografía

Android Developers, Última actualización: 2024-08-23 (UTC), Ciclo de vida de la actividad, consultado el 19 de abril de 2025, https://developer.android.com/guide/components/activities/activity-lifecycle?hl=es-419#lc