

Informe Actividades y Su Ciclo de Vida

Santiago Quesada Marulanda

SENA

ADSI 2469283

Cristhian David Henao Hoyos

24 de Marzo de 2023

¿Qué es el logcat?

El logcat es una herramienta que permite llevar el registro de mensajes en Android. Esto permite imprimir mensajes dependiendo del estado de la aplicación, así como también los posibles errores que se presenten.

Niveles en logcat

- **I / Info:** Muestra mensajes de información
- **W / Warning:** Muestra mensajes con advertencias
- **E / Error:** Muestra errores en la aplicación
- **F / Fatal:** Muestra errores fatales
- **D / Debug:** Muestra mensajes que sirven para la depuración de aplicaciones
- **V / Verbose:** Muestra la menor prioridad, con más información de la necesaria

Actividades

- **onCreate() y onStart()**

```
package com.example.proectomensajesqm

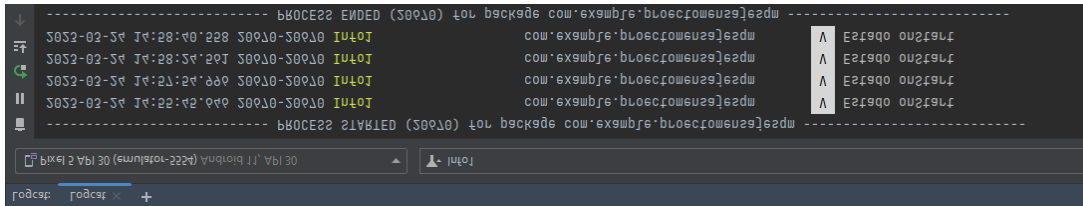
import ...

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }

    override fun onStart() {
        super.onStart()
        Log.v( tag: "Info1", msg: "Estado onStart")
    }
}
```

El método **onStart()** se vuelve visible pero no está en primer plano y listo para la interacción con el usuario. El método **super.onStart()** ejecuta el comportamiento del método **onStart()** en la clase principal.

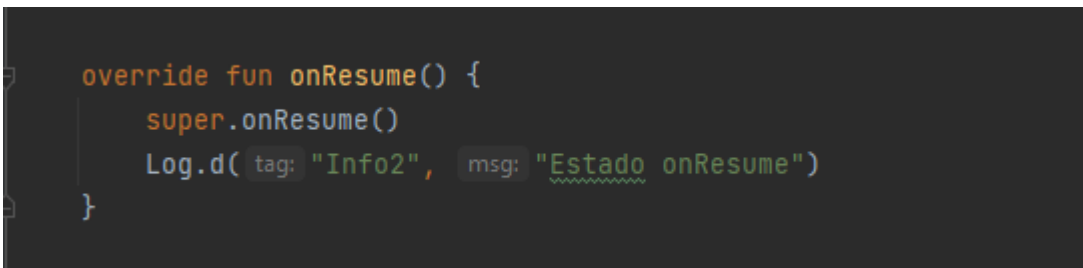
El método **Log.i()** utiliza una etiqueta **“Info1”** que ayuda a identificar el origen del mensaje registrado, así como también el mensaje que se va a registrar **“Estado onStart”**.



The screenshot shows the Logcat window in Android Studio. The filter is set to 'Info1'. The log messages are as follows:

Time	Process	Message
2023-03-24 14:58:45.128	com.example.proectomensajesqm	Info1 Estado onStart
2023-03-24 14:58:45.128	com.example.proectomensajesqm	Info1 Estado onStart
2023-03-24 14:58:45.128	com.example.proectomensajesqm	Info1 Estado onStart
2023-03-24 14:58:45.128	com.example.proectomensajesqm	Info1 Estado onStart

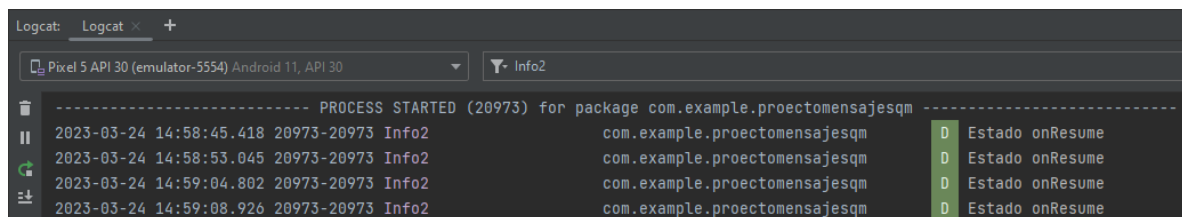
- **onResume()**



```
override fun onResume() {  
    super.onResume()  
    Log.d( tag: "Info2", msg: "Estado onResume")  
}
```

El método **onResume()** llama al sistema Android cuando la actividad se reanuda y pasa de estar en segundo a primer plano. Se utiliza para inicialización después de que la actividad se haya detenido. El método **super.onResume()** garantiza que se realice la inicialización correspondiente.

El método **Log.i()** utiliza una etiqueta **“Info2”** que ayuda a identificar el origen del mensaje registrado, así como también el mensaje que se va a registrar **“Estado onResume”**.



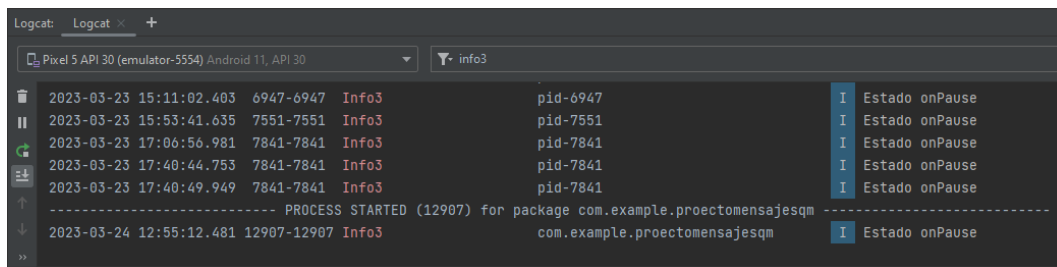
The screenshot shows the Logcat window in Android Studio. The filter is set to 'Info2'. The log messages are as follows:

Time	Process	Message
2023-03-24 14:58:45.128	com.example.proectomensajesqm	Info2 Estado onResume
2023-03-24 14:58:53.045	com.example.proectomensajesqm	Info2 Estado onResume
2023-03-24 14:59:04.802	com.example.proectomensajesqm	Info2 Estado onResume
2023-03-24 14:59:08.926	com.example.proectomensajesqm	Info2 Estado onResume

- onPause()

```
override fun onPause() {  
    super.onPause()  
    Log.i( tag: "Info3", msg: "Estado onPause")  
}
```

El método **onPause()** se llama cuando la se abandona dicha Actividad o cuando se inicia otra Actividad y se aleja de la actual. El método **super.onPause()** ejecuta la función de la clase principal, luego se registra el mensaje usando la función **Log.i()**. El log escribe un mensaje en el logcat, que en este caso es **"Estado onPause"** con la etiqueta **"Info3"**.



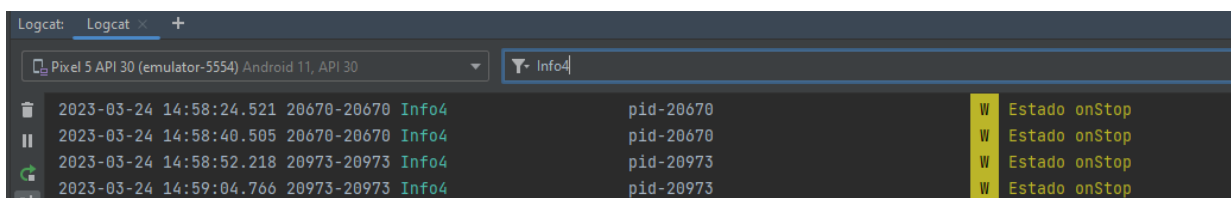
The screenshot shows the Logcat window in Android Studio. The filter is set to 'info3'. The log entries are as follows:

Time	Process	Tag	Message
2023-03-23 15:11:02.403	6947-6947	Info3	pid-6947 Estado onPause
2023-03-23 15:53:41.635	7551-7551	Info3	pid-7551 Estado onPause
2023-03-23 17:06:56.981	7841-7841	Info3	pid-7841 Estado onPause
2023-03-23 17:40:44.753	7841-7841	Info3	pid-7841 Estado onPause
2023-03-23 17:40:49.949	7841-7841	Info3	pid-7841 Estado onPause
----- PROCESS STARTED (12907) for package com.example.proectomensajesqm -----			
2023-03-24 12:55:12.481	12907-12907	Info3	com.example.proectomensajesqm Estado onPause

- onStop()

```
override fun onStop() {  
    super.onStop()  
    Log.w( tag: "Info4", msg: "Estado onStop")  
}
```

El método **onStop()** es un método de ciclo de vida que se llama cuando la actividad ya no es visible para el usuario. El método **super.onStop()** para garantizar que se realice la limpieza necesaria. Luego, registra un mensaje informativo en la consola utilizando la clase de registro de Android con una etiqueta de **"Info4"** y un mensaje de **"Estado onStop"**.



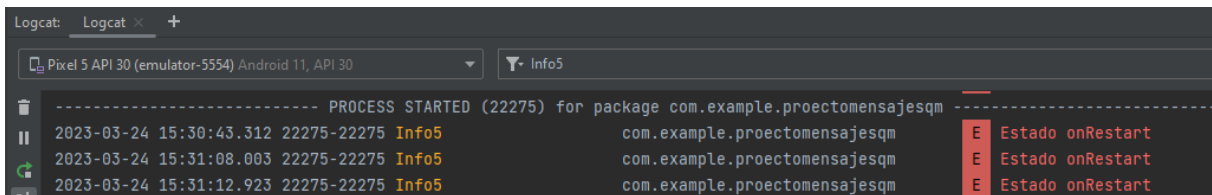
The screenshot shows the Logcat window in Android Studio. The filter is set to 'Info4'. The log entries are as follows:

Time	Process	Tag	Message
2023-03-24 14:58:24.521	20670-20670	Info4	pid-20670 Estado onStop
2023-03-24 14:58:40.505	20670-20670	Info4	pid-20670 Estado onStop
2023-03-24 14:58:52.218	20973-20973	Info4	pid-20973 Estado onStop
2023-03-24 14:59:04.766	20973-20973	Info4	pid-20973 Estado onStop

- **onRestart()**

```
override fun onRestart() {  
    super.onRestart()  
    Log.e( tag: "Info5", msg: "Estado onRestart")  
}
```

El método **onRestart()** se ejecuta cuando una Actividad se encuentra abandonada temporalmente y va a iniciarse nuevamente, con esto el sistema retoma la Actividad y su estado será **onStart()**. Esta llama al método **super.onRestart()** para realizar cualquier inicialización de superclase necesaria y luego registra un mensaje en la consola usando el método **Log.i()** con una etiqueta **"Info5"** y un mensaje **"Estado onRestart"**.



- **onDestroy()**

```
override fun onDestroy() {  
    super.onDestroy()  
    Log.i( tag: "Info6", msg: "Estado onDestroy")  
}
```

El método **onDestroy()** se llama cuando la actividad está a punto de destruirse y se puede usar para realizar tareas de limpieza o liberar los recursos utilizados por la actividad. Este llamando al método principal **super.onDestroy()** para garantizar que el sistema realice todas las tareas de limpieza necesarias, y luego registra un mensaje usando el método **Log.i()** para indicar que la actividad está en el estado "onDestroy".

Version Control ▶ Run Profiles ⚙️ Local 📦 App Configuration ⚙️ Build ⚙️ TODO 🐛 Problems 📧 Terminal ⚙️ Services 🛠️ App Inspection

----- PROCESS ENDED (J8814) for package com.example.bloectomen291ezdw -----
S052-02-54 J4:51:5J'245 J8814-J8814 Iu40q com.example.bloectomen291ezdw I Est9do on9ez1oY
----- PROCESS STARTED (J8814) for package com.example.bloectomen291ezdw -----