

# Dalvik Debug Monitor Server: (DDMS) - Logcat

## Sumario

- 1 Introducción
- 2 Xestión de ficheiros
- 3 Parámetros dun proceso
- 4 Parámetros do dispositivo
- 5 Enviar datos ao dispositivo
- 6 DDMS sen Eclipse
- 7 Logcat

## Introducción

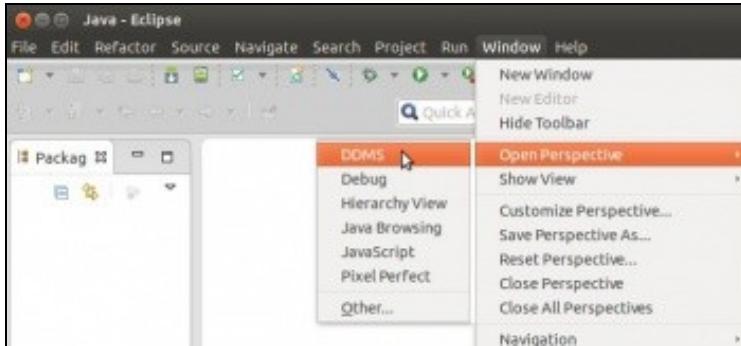
- **Dalvik Debug Monitor Server (DDMS)** é unha utilidade de debug que permite visualizar o consumo de CPU, de Memoria, de rede. Tamén permite, por exemplo, enviar unha chamada ou SMS a un AVD ou coordenadas GPS para que sexan recollidas nunha aplicación.
- U?ase tamén para facer debug das aplicac?ns como se verá na Unidade 2.
- Para o seu funcionamento precisa da utilidade ADB, e moitas das cousas realizadas con esta utilidade p?ndense realizar dende o DDMS.
- Co cal cando se inicia Eclipse, o DDMS inicia o servidor adb.
- No seguinte enlace p?dese obter m?is informaci?n: <http://developer.android.com/tools/debugging/ddms.html>
- **IMPORTANTE:** Se se est? a usar a versi?n de Android Studio a utilidade ch?mase "Android Device Monitor".
  - ◆ ?Es preciso ter instalado Java
    - ◊ No caso de Linux Java 8 ou superior.

## Xestión de ficheiros

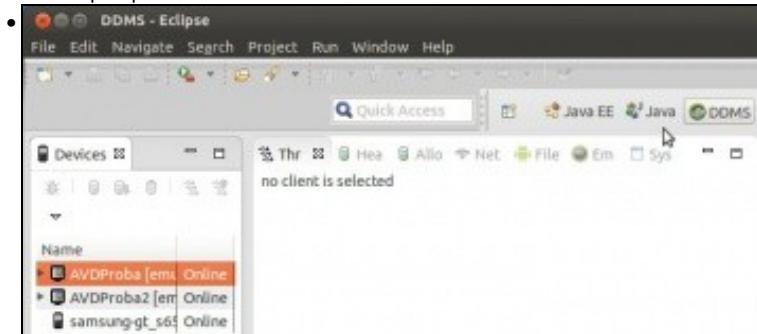
- Ao igual que co ADB vanse poder manipular ficheiros do dispositivo pero dun modo gr?fico.

- **Importante:** Se se est? a usar Android Studio:
  - ◆ Abrir un proxecto.
  - ◆ Ir ao men?u de: Tools-->Android-->Android Device Monitor

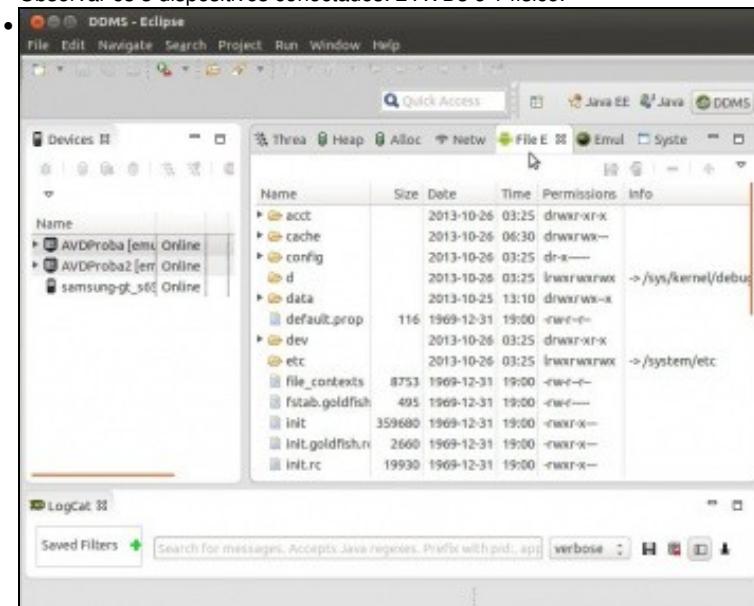
- Xestión de ficheiros



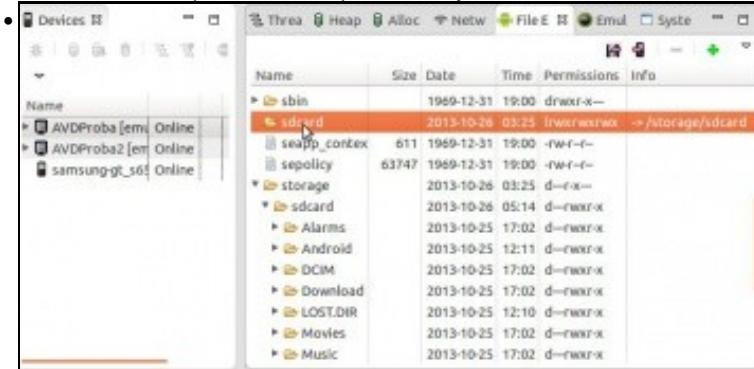
Abrir a perspectiva DDMS.



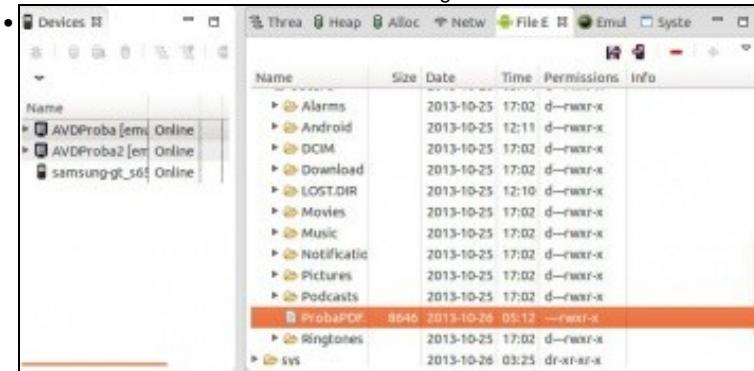
Observar os 3 dispositivos conectados: 2 AVDs e 1 físico.



Seleccionar un dispositivo e ir á lapela **File Explorer**. Vemos o contido da raíz.



Observar como /sdcard/ é un enlace a /storage/sdcard.



En /storage/sdcard/ está o ficheiro PDF de proba usado anteriormente.

- 

Name	Size	Date	Time	Permissions
▶ ⌂ Alarms		2013-10-25	17:02	d-rw xr-x
▶ ⌂ Android		2013-10-25	12:11	d-rw xr-x
▶ ⌂ DCIM		2013-10-25	17:02	d-rw xr-x
▶ ⌂ Download		2013-10-25	17:02	d-rw xr-x
▶ ⌂ LOST.DIR		2013-10-25	12:10	d-rw xr-x
▶ ⌂ Movies		2013-10-25	17:02	d-rw xr-x
▶ ⌂ Music		2013-10-25	17:02	d-rw xr-x
▶ ⌂ Notificacio		2013-10-25	17:02	d-rw xr-x
▶ ⌂ Pictures		2013-10-25	17:02	d-rw xr-x
▶ ⌂ Podcasts		2013-10-25	17:02	d-rw xr-x
▶ ProbaPDF.pdf	8646	2013-10-26	05:12	—rwxr-x
▶ ⌂ Ringtones		2013-10-25	17:02	d-rw xr-x
▶ ⌂ SYS		2013-10-26	03:25	dr-xr-xr-x

Premendo no botón **Pull** poderíase extraer ese ficheiro ...

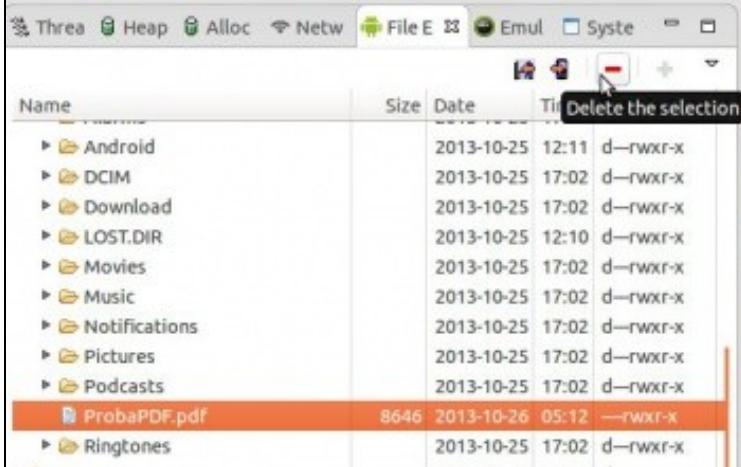
- 

... ao ordenador.

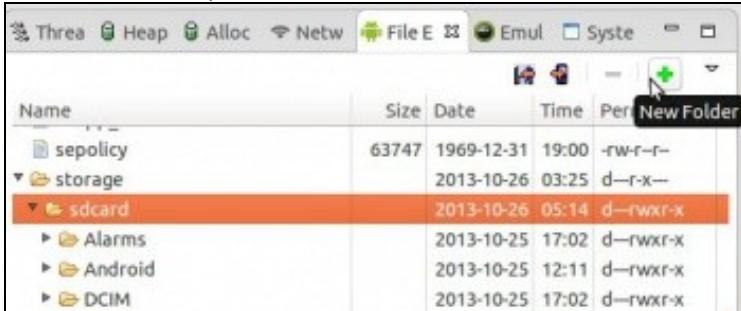
- 

Name	Size	Date	Time	Permissions
▶ ⌂ Alarms		2013-10-25	17:02	d-rw xr-x
▶ ⌂ Android		2013-10-25	12:11	d-rw xr-x
▶ ⌂ DCIM		2013-10-25	17:02	d-rw xr-x
▶ ⌂ Download		2013-10-25	17:02	d-rw xr-x
▶ ⌂ LOST.DIR		2013-10-25	12:10	d-rw xr-x
▶ ⌂ Movies		2013-10-25	17:02	d-rw xr-x

Premendo en **Push** pódense meter ficheiros do ordenador no dispositivo.

- 

Premendo en **Delete** pódese eliminar.

- 

Premendo en **New folder** pódese crear un novo cartafol.

## Parámetros dun proceso

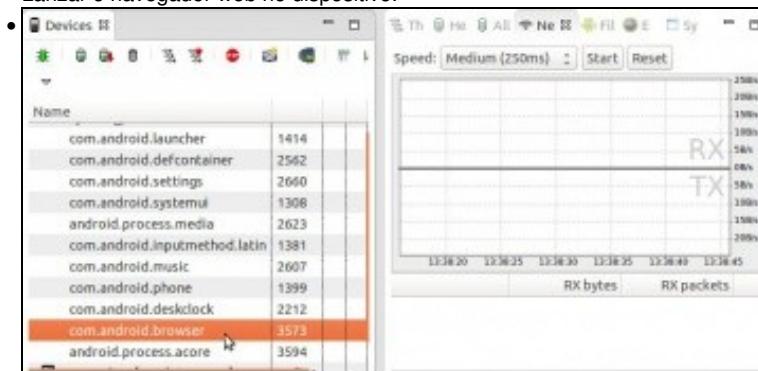
- Isto serve para comprobar os recursos que consume unha aplicación ou proceso.
- A continuación vaise lanzar o navegador web no dispositivo e vaise monitorizar.

**NOTA:** Nas imaxes INTEL vai dar un erro á hora de capturar o tráfico. En caso de querer probar esta funcionalidade baixar unha imaxe ARM dende o Android SDK Manager e crear un novo dispositivo virtual.

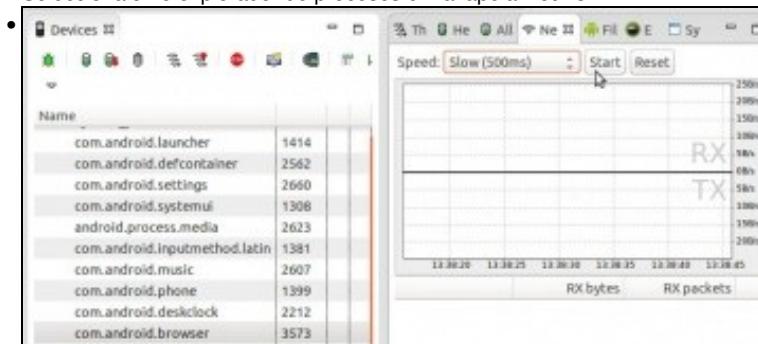
- Parámetros dun proceso



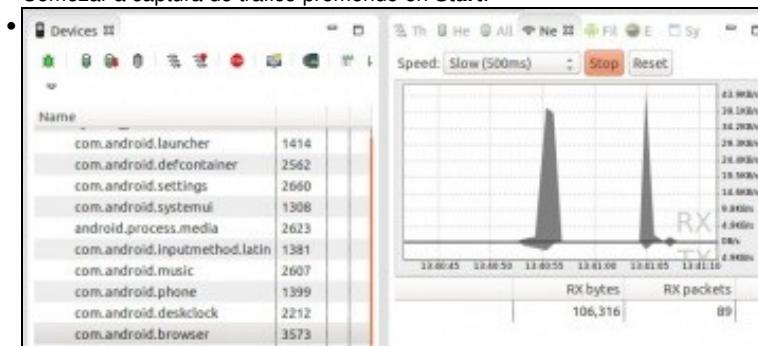
Lanzar o navegador web no dispositivo.



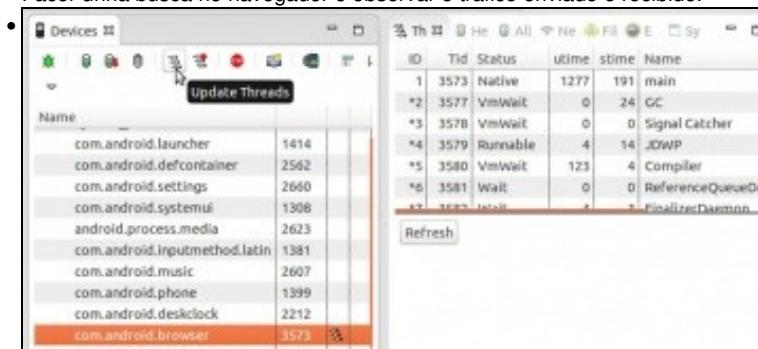
Seleccionalo no explorador de procesos e ir á lapela Network.



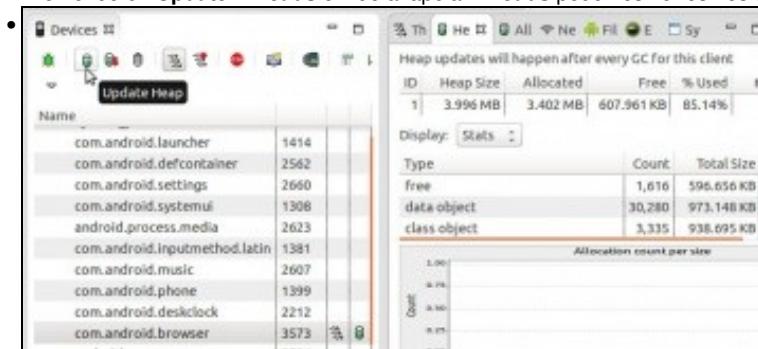
Comezar a captura de tráfico premendo en Start.



Facer unha busca no navegador e observar o tráfico enviado e recibido.



Premendo en **Update Threads** e indo á lapela **Threads** podemos ver os fíos que ten abertos ese proceso.



Premendo en **Update Heap** e indo á lapela **Heap** podemos ver os consumos de memoria.

The screenshot shows the Android Device Monitor with the 'Heap' tab selected. The main pane displays a table of processes with their IDs and memory usage. A chart below the table shows the allocation count per size, with the y-axis ranging from 0.00 to 1.00. The data points are clustered around 1.00, indicating high allocation counts for most memory sizes.

Name	ID	Heap Size	Allocated	Free	% Used
com.android.launcher	1414				
com.android.defcontainer	2562				
com.android.settings	2660				
com.android.systemui	1308				
android.process.media	2623				
com.android.inputmethod.latin	1381				
com.android.phone	1399				
com.android.deskclock	2212				
android.process.acore	3594				
com.android.browser	4893				

Volvendo premer en **Update Threads** deshabilitamos a monitorización de threads para ese proceso. O mesmo con **Update Heap**

The screenshot shows the Android Device Monitor with the 'Threads' tab selected. The main pane displays a table of processes with their IDs and memory usage. A button labeled 'Stop Process' is visible in the toolbar. The data table is identical to the one in the previous screenshot.

Name	ID	Heap Size	Allocated	Free	% Used
com.android.launcher	1414				
com.android.defcontainer	2562				
com.android.settings	2660				
com.android.systemui	1308				
android.process.media	2623				
com.android.inputmethod.latin	1381				
com.android.phone	1399				
com.android.deskclock	2212				
android.process.acore	3594				
com.android.browser	4893				

Podemos parar o proceso, neste caso o navegador.

The screenshot shows the Android Device Monitor with the 'Threads' tab selected. The main pane displays a table of processes with their IDs and memory usage. The process 'AVDProba [emulator-5554]' is selected and shown as 'Online'. The data table is identical to the one in the previous screenshots.

Name	ID	Heap Size	Allocated	Free	% Used
AVDProba [emulator-5554]	Online				
system_process	1239				
com.android.launcher	1414				
com.android.defcontainer	2562				
com.android.settings	2660				
com.android.systemui	1308				
android.process.media	2623				
com.android.inputmethod.latin	1381				
com.android.phone	1399				
com.android.deskclock	2212				
android.process.acore	3594				

Proceso parado ...

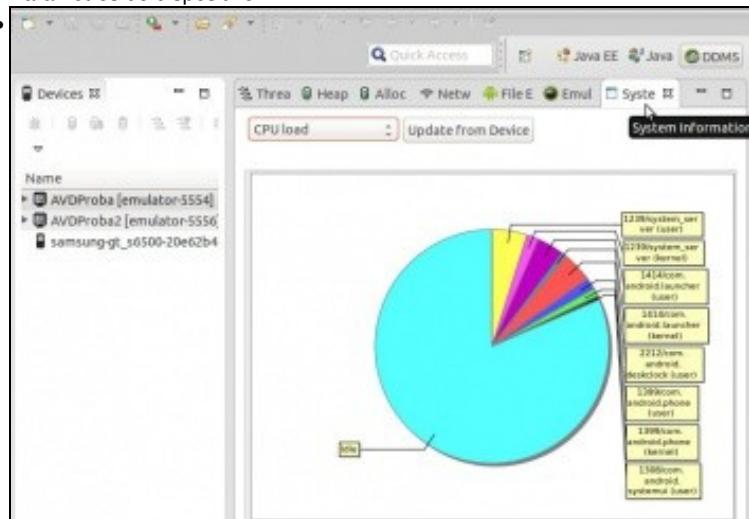


... e tamén no dispositivo.

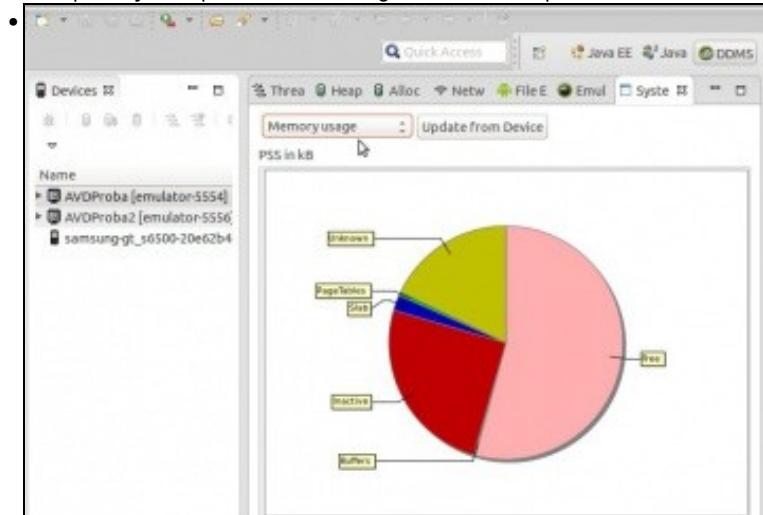
## Parámetros do dispositivo

- Podemos comprobar o uso de recursos que fan os procesos no dispositivo.

- Parámetros do dispositivo



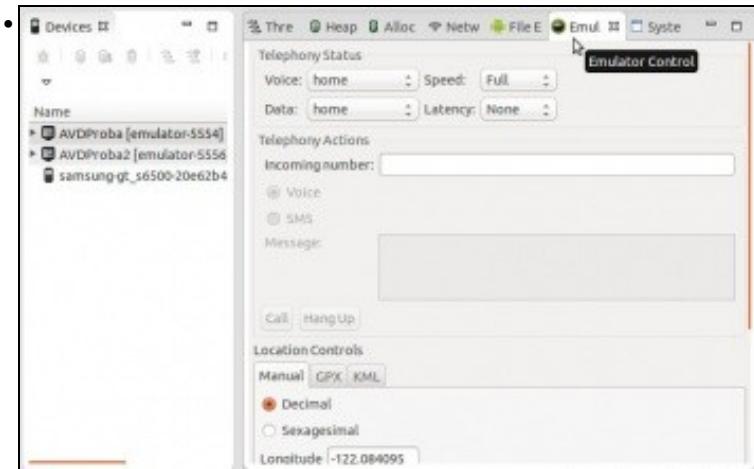
Na lapela **System** podemos ver a carga de CPU do dispositivo ...



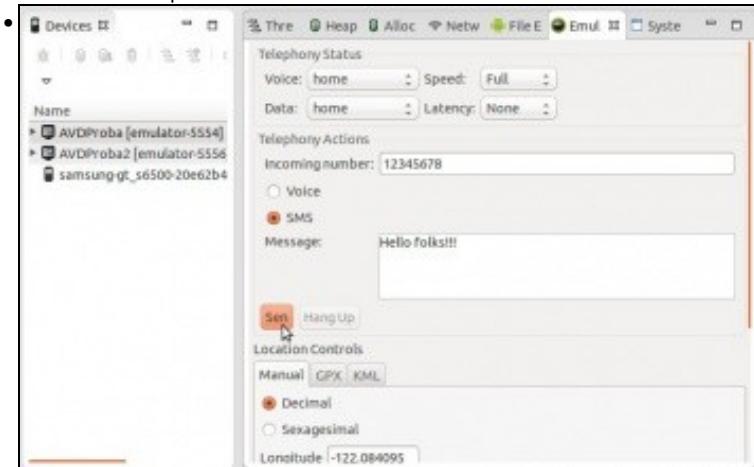
O consumo de memoria, etc.

## Enviar datos ao dispositivo

- Para poder probar as aplicacións nos AVDs ás veces é preciso que estas reciban datos do exterior: unha chamada, un sms, coordenadas gps, etc.
- Enviar datos



Seleccionar a lapela **Emulador**.



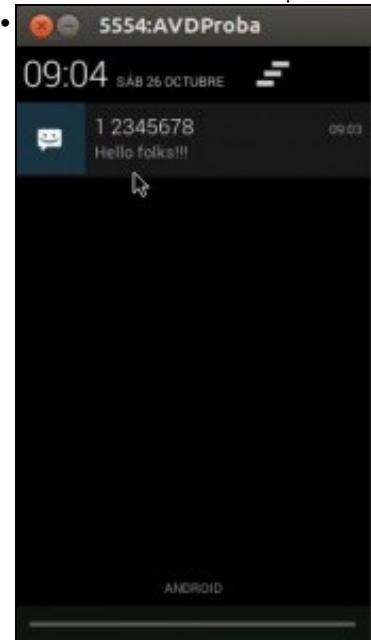
Imos enviar un sms ao AVD. Poñer un número de teléfono, marcar **SMS**, escribir a mensaxe e premer en **Send(d)**.



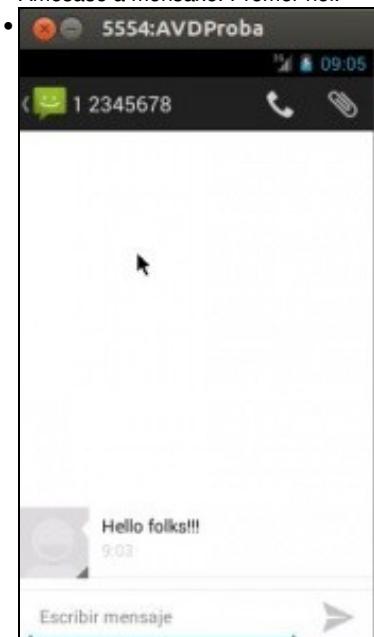
No dispositivo recíbe a notificación de SMS entrante.



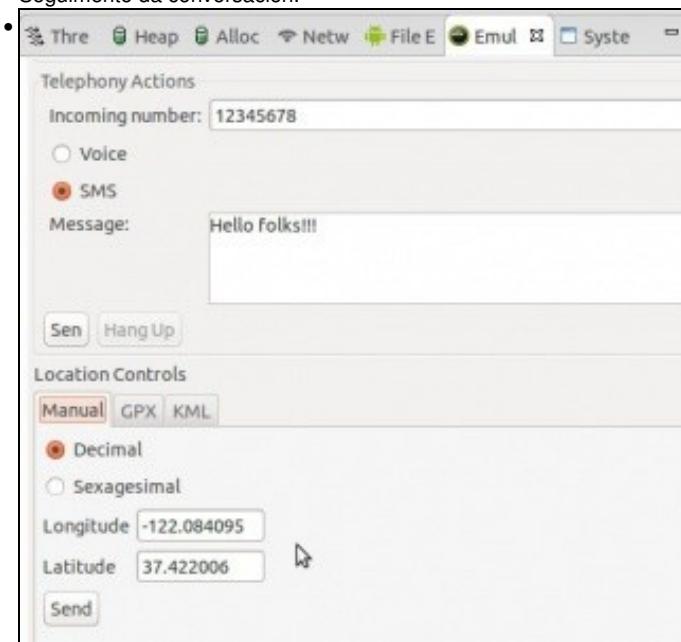
Premer na ícone de aviso superior e arrastrar a barra para abaixo.



Amósase a mensaxe. Premer nel.



Seguimento da conversación.



Tamén se poderían enviar coordenadas GPS, ben de xeito manual ou ben cargadas dun ficheiro.

## DDMS sen Eclipse

- Tamén se pode iniciar o utilidade DDMS sen facer uso de Eclipse.
- A utilidade DDMS está no cartafol **tools**.
- **Importante:** Nas novas versións do SDK para lanzar o monitor este chámase: **monitor**

- Utilidade DDMS

```
● ladmin@ubase: ~
ladmin@ubase:~$ ddms
The standalone version of DDMS is deprecated.
Please use Android Device Monitor (tools/monitor) instead.
```

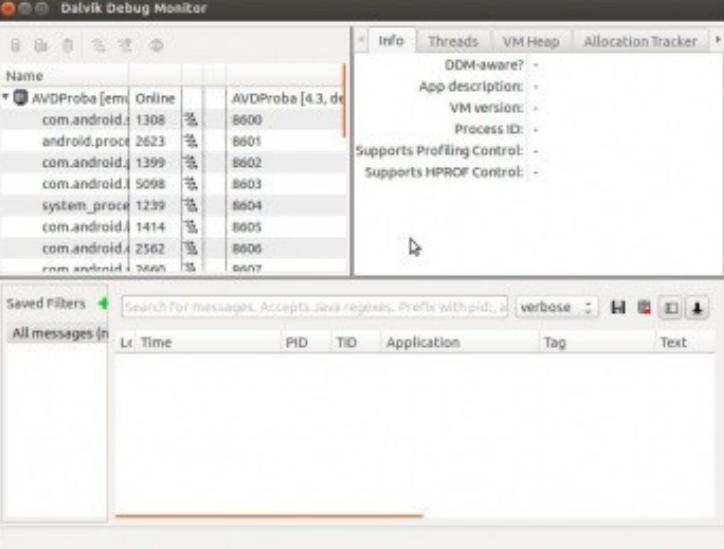
Executamos **ddms** e indícanos que está obsoleta, que usemos a utilidade **monitor**.

```
● ladmin@ubase: ~
ladmin@ubase:~$ monitor
```



Lanzamos **monitor**

```
● Dalvik Debug Monitor
```

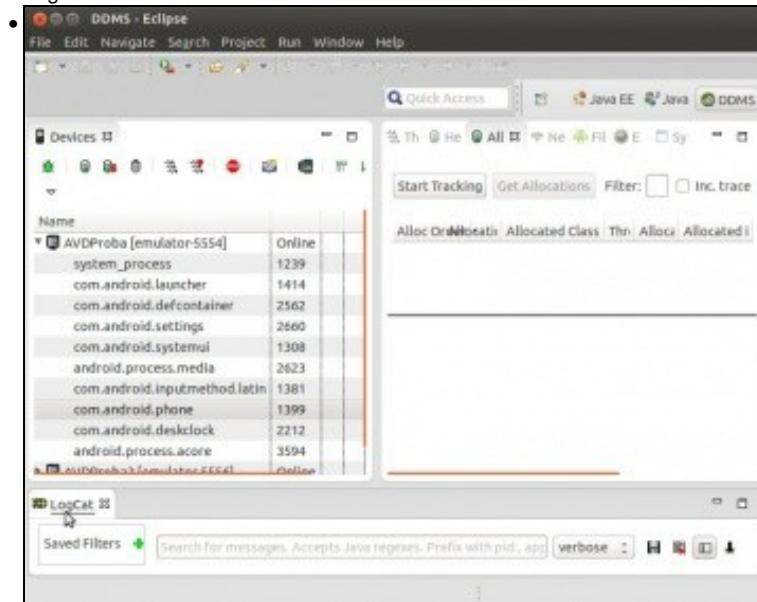


E vemos que temos a mesma ventá que cando usábamos DDMS dende Eclipse.

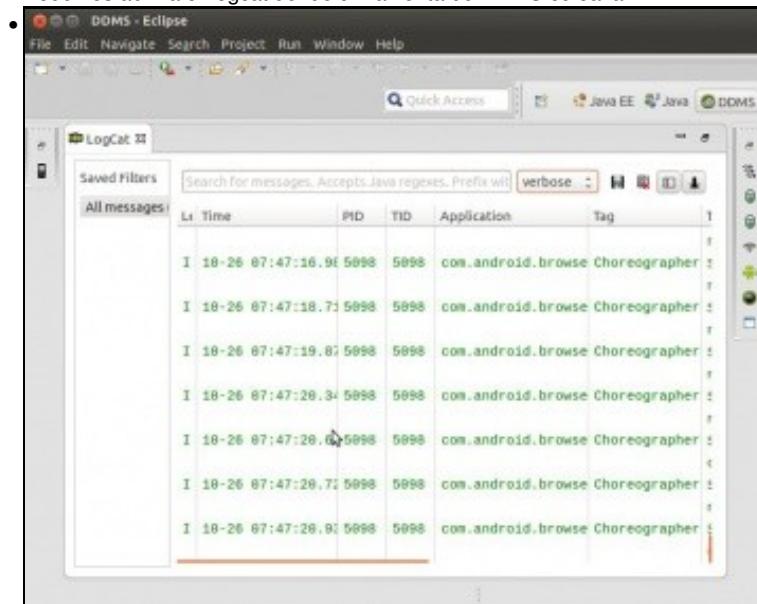
## Logcat

- O sistema de log de Android proporciona un mecanismo polo cal vai recollendo toda a información de saída do dispositivo.
- Esa información pódese capturar e ver con **Logcat** e tamén se pode filtrar polo tipo de mensaxe.
- Na UNIDADE 2 do curso usaremos esta utilidade para controlar unha aplicación.
- Para máis información: <http://developer.android.com/tools/help/logcat.html>

- Logcat



Podemos abrir o Logcat dende unha ventá de DDMS ou Java.



Rexistro de saída do dispositivo.

```
• ladmin@ubase: ~
ladmin@ubase:~$ adb -s emulator-5554 logcat
```

Tamén se pode usar **adb logcat** para ver ...

```
• ladmin@ubase: ~
I/Choreographer( 1414): Skipped 56 frames!  The application may be doing too muc
h work on its main thread.
D/dalvikvm( 5893): GC_FOR_ALLOC freed 261K, 15% free 2915K/3416K, paused 450ms,
total 455ms
I/Choreographer( 1239): Skipped 264 frames!  The application may be doing too mu
ch work on its main thread.
I/Choreographer( 1239): Skipped 67 frames!  The application may be doing too muc
h work on its main thread.
W/ActivityManager( 1239): Launch timeout has expired, giving up wake lock!
W/ActivityManager( 1239): Activity idle timeout for ActivityRecord{b43c7560 u0 c
on.android.camera/.Camera}
```

... por consola as mensaxes que está enviando a o dispositivo.

-- Ángel D. Fernández González e Carlos Carrión Álvarez -- (2017).