



# Debugging.TV

Frame 0x33

Presenter: Dmitry Vostokov

SOFTWARE DIAGNOSTICS SERVICES

[www.PatternDiagnostics.com](http://www.PatternDiagnostics.com)

Including Memory Dump and Software Trace Analysis Audit, Seminars,  
Certification and Training

Sponsors

**OPENTASK**  
Iterative and Incremental Publishing

A stylized world map in shades of blue and black, serving as a background for the Opentask logo.

# Topics

- Spiking Thread pattern (Android)
- Deadlock pattern (Android)
- ADB
- Paratext pattern (Android)

# Spiking Thread

```
(new Thread(new Runnable() { // Thread #11
    public void run() {
        while (true) {
            double num = Math.random();
        }
    }
})) .start();
```

```
09-08 22:38:54.629: I/System.out(21804): Thread[main,5,main]:
09-08 22:38:54.629: I/System.out(21804):     android.os.MessageQueue.nativePollOnce(Native Method)
09-08 22:38:54.629: I/System.out(21804):     android.os.MessageQueue.next(MessageQueue.java:119)
09-08 22:38:54.629: I/System.out(21804):     android.os.Looper.loop(Looper.java:117)
09-08 22:38:54.629: I/System.out(21804):     android.app.ActivityThread.main(ActivityThread.java:3687)
09-08 22:38:54.639: I/System.out(21804):     java.lang.reflect.Method.invokeNative(Native Method)
09-08 22:38:54.639: I/System.out(21804):     java.lang.reflect.Method.invoke(Method.java:507)
09-08 22:38:54.639: I/System.out(21804):     com.android.internal.os.ZygoteInit$MethodAndArgsCaller.run(ZygoteInit.java:867)
09-08 22:38:54.639: I/System.out(21804):     com.android.internal.os.ZygoteInit.main(ZygoteInit.java:625)
09-08 22:38:54.639: I/System.out(21804):     dalvik.system.NativeStart.main(Native Method)
09-08 20:12:03.779: I/System.out(20147): Thread[Thread-11,5,main]:
09-08 20:12:03.789: I/System.out(20147):     java.util.Random.next(Random.java:90)
09-08 20:12:03.789: I/System.out(20147):     java.util.Random.nextDouble(Random.java:122)
09-08 20:12:03.789: I/System.out(20147):     java.lang.Math.random(Math.java:965)
09-08 20:12:03.789: I/System.out(20147):     com.example.spikingthread.FullscreenActivity$6$1.run(FullscreenActivity.java:150)
09-08 20:12:03.789: I/System.out(20147):     java.lang.Thread.run(Thread.java:1019)
```

# Deadlock

```
public void run() { // Thread #11
    synchronized(cs1) {
        try { Thread.sleep(2000); } catch (InterruptedException e) {}
        synchronized(cs2) {
            try { Thread.sleep(1000); } catch (InterruptedException e) {}
        }
    }
}
```

```
public void run() { // Thread #12
    synchronized(cs2) {
        try { Thread.sleep(4000); } catch (InterruptedException e) {}
        synchronized(cs1) {
            try { Thread.sleep(1000); } catch (InterruptedException e) {}
        }
    }
}
```

```
09-08 22:38:54.629: I/System.out(21804): Thread[Thread-11,5,main]:
09-08 22:38:54.629: I/System.out(21804):
com.example.deadlock.FullscreenActivity$6$1.run(FullscreenActivity.java:157)
09-08 22:38:54.629: I/System.out(21804):     java.lang.Thread.run(Thread.java:1019)
09-08 22:38:54.639: I/System.out(21804): Thread[Thread-12,5,main]:
09-08 22:38:54.639: I/System.out(21804):
com.example.deadlock.FullscreenActivity$6$2.run(FullscreenActivity.java:177)
09-08 22:38:54.639: I/System.out(21804):     java.lang.Thread.run(Thread.java:1019)
```

# ADB session..



# Paratext

```
$ ps -t
[...]
```

<b>app_57</b>	21600	93	146880	19352	ffffffff	00000000	S	<b>com.example.spikingthread</b>
app_57	21601	21600	146880	19352	ffffffff	00000000	S	HeapWorker
app_57	21602	21600	146880	19352	ffffffff	00000000	S	GC
app_57	21603	21600	146880	19352	ffffffff	00000000	S	Signal Catcher
app_57	21604	21600	146880	19352	ffffffff	00000000	S	JDWP
app_57	21605	21600	146880	19352	ffffffff	00000000	S	Compiler
app_57	21606	21600	146880	19352	ffffffff	00000000	S	Binder Thread #
app_57	21607	21600	146880	19352	ffffffff	00000000	S	Binder Thread #
app_57	21608	21600	146880	19352	ffffffff	00000000	S	Thread-10
app_57	21611	21600	146880	19352	ffffffff	00000000	R	<b>Thread-11</b>
shell	21619	143	800	336	c00a6fc8	afd0c3bc	S	/system/bin/sh
root	21631	2	0	0	ffffffff	00000000	S	flush-138:13
<b>app_68</b>	21635	93	151092	18456	ffffffff	00000000	S	<b>com.example.deadlock</b>
app_68	21636	21635	151092	18456	ffffffff	00000000	S	HeapWorker
app_68	21637	21635	151092	18456	ffffffff	00000000	S	GC
app_68	21638	21635	151092	18456	ffffffff	00000000	S	Signal Catcher
app_68	21639	21635	151092	18456	ffffffff	00000000	S	JDWP
app_68	21640	21635	151092	18456	ffffffff	00000000	S	Compiler
app_68	21641	21635	151092	18456	ffffffff	00000000	S	Binder Thread #
app_68	21642	21635	151092	18456	ffffffff	00000000	S	Binder Thread #
app_68	21643	21635	151092	18456	ffffffff	00000000	S	Thread-10
app_68	21650	21635	151092	18456	ffffffff	00000000	S	<b>Thread-11</b>
app_68	21652	21635	151092	18456	ffffffff	00000000	S	<b>Thread-12</b>

```
[...]
```

# **!Ad** Hardcore Software Diagnostics Training

Sep - Nov, 2013	<u>Mobile Software Diagnostics</u> (FREE) <u>Psychology of Software Diagnostics</u> (FREE) <u>Semiotics of Debugging</u> (FREE) <u>Generative Software Narratology</u> (FREE) <u>Software Diagnostics: Requirements, Architecture, Design, Implementation and Improvement</u> (FREE)
October 25-28, 2013	<u>Accelerated Disassembly, Reconstruction and Reversing</u>
November 15-25, 2013	<u>Accelerated Windows Memory Dump Analysis</u>

# Debugging.TV

Now on YouTube!

<http://www.youtube.com/DebuggingTV>