High-level introduction to Android low-level

Jollen's Consulting
專業課程供應

與優質的課程平臺商合作, 專注課程研究與開發,致力創造教育訓練的價值

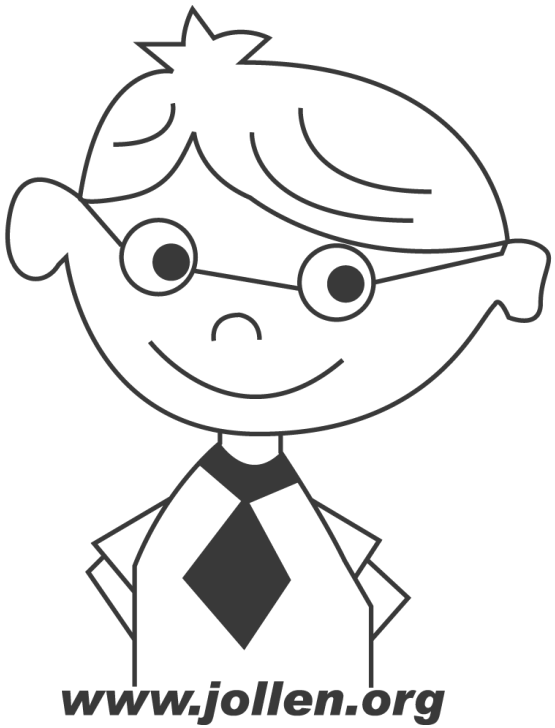
課程供應與開發者,致力於價值創造
詳細資訊: www.jollen.org/consulting

歡迎參加由 Jollen's Consulting 所規劃與主講的課程，參加我們任何課程，您皆可透過 ta@jollen.org 信箱登錄個人資料，即可加入郵件列表；郵件主旨請填寫 [加入郵件列表]、郵件內文空白即可

我們將不定時提供以下資訊：

- 公開的講義電子檔
- 題庫系統使用帳號
- 實作測試題目（提供學員練功題目）

Jollen's Consulting Team



Jollen Chen (陳俊宏)

Email：jollen@jollen.org

Blog：jollen.org/blog

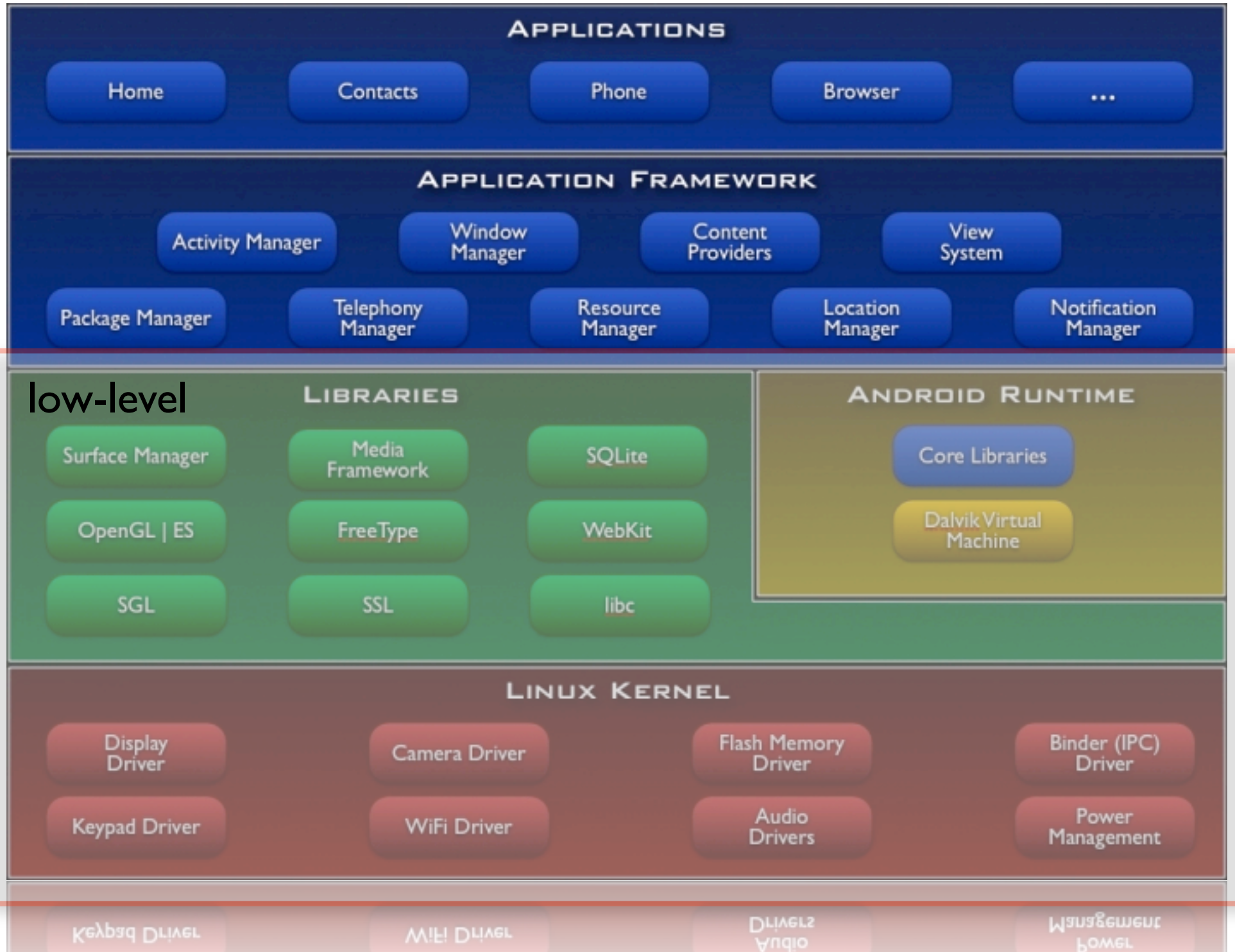
演講時間：2009.09.18,14:00~17:00 (3hr)

演講地點：台北市．佳士達T00會議室

High-level introduction to Android low-level

課程形式： ☑演講（Presentation） ☐訓練（Training） ☐實作指導（Coaching）

請使用投影片
放映方式閱讀


Applications
Home
Contacts
Phone
Browser
...
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Package Manager
Telephony Manager
Resource Manager
Location Manager
Notification Manager
low-level
Libraries
Surface Manager
Media Framework
SQLite
OpenGL | ES
FreeType
WebKit
SGL
SSL
libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Flash Memory Driver
Binder (IPC) Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management

Android Kernel Features

Android Kernel Features

哪裡不一樣？

Android Kernel

- Binder
- Ashmem (Android shared memory)
- PMEM (Processor memory allocator)
- logcat (Android Logger)
- wakelock (Android Power Management)
- Alarm

	Key Features	2.6.23	2.6.25	2.6.27	2.6.29
1	Alarm Driver	O	O	O	O
2	Android Logger	O	O	O	O
3	Low Memory Killer	O	O	O	O
4	Power Management	O	O	O	O
5	USB Gadget	O	O	O	O
6	ASHMEM	X	O	O	O
7	PMEM	X	X	O	O
8	x86 Support	X	X	O	O
9	./drivers/staging/Android/	X	X	X	O

/pub/scm / linux/kernel/git/torvalds/linux-2.6.git / tree

summary | shortlog | log | commit | commitdiff | tree
snapshot

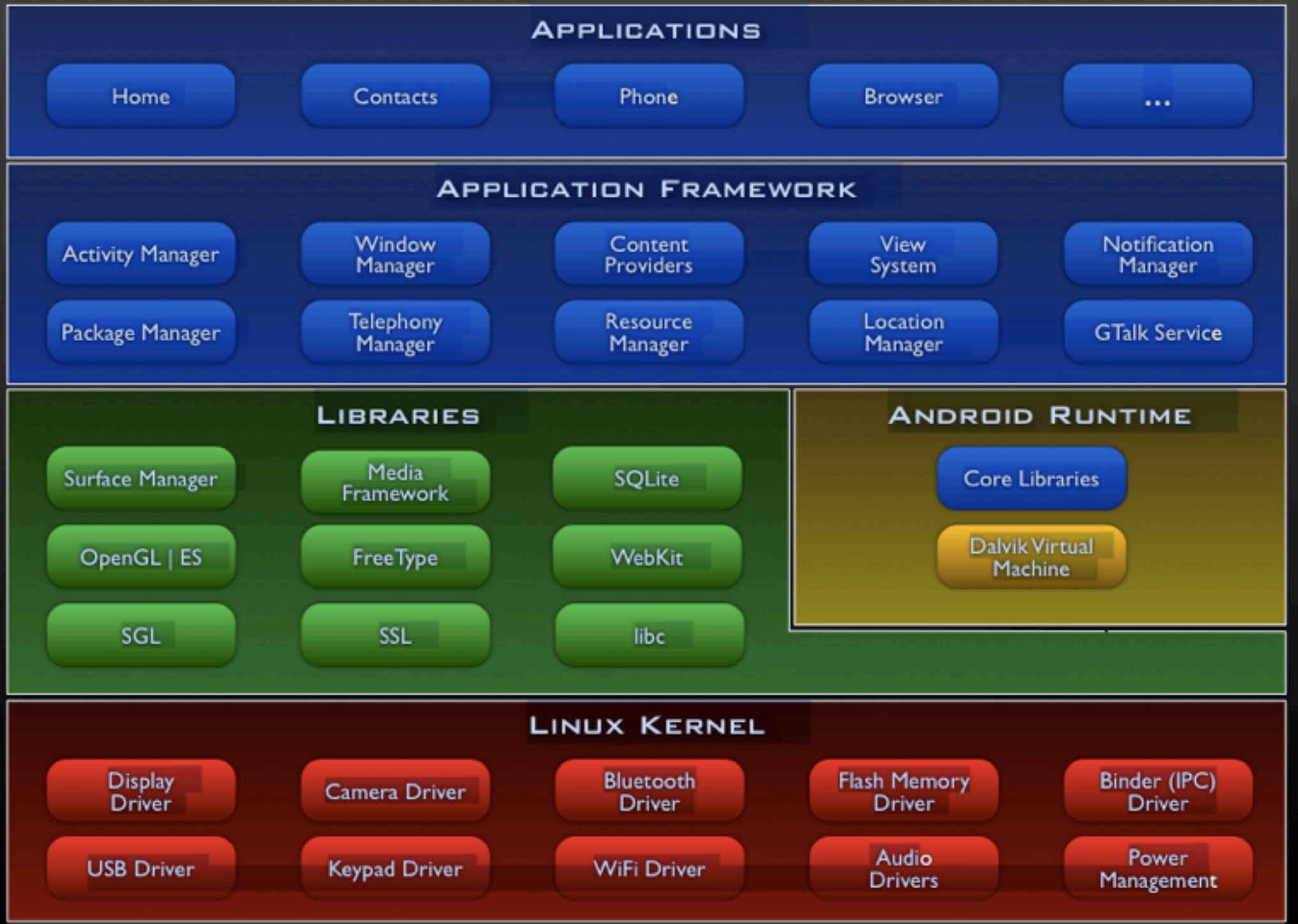
Merge branch 'for-linus' of git://git.kernel.org/pub/scm/linux/kernel/git/drzeus/mmc

[linux/kernel/git/torvalds/linux-2.6.git] /
[linux/kernel/git/torvalds/linux-2.6.git] / drivers / staging / android /

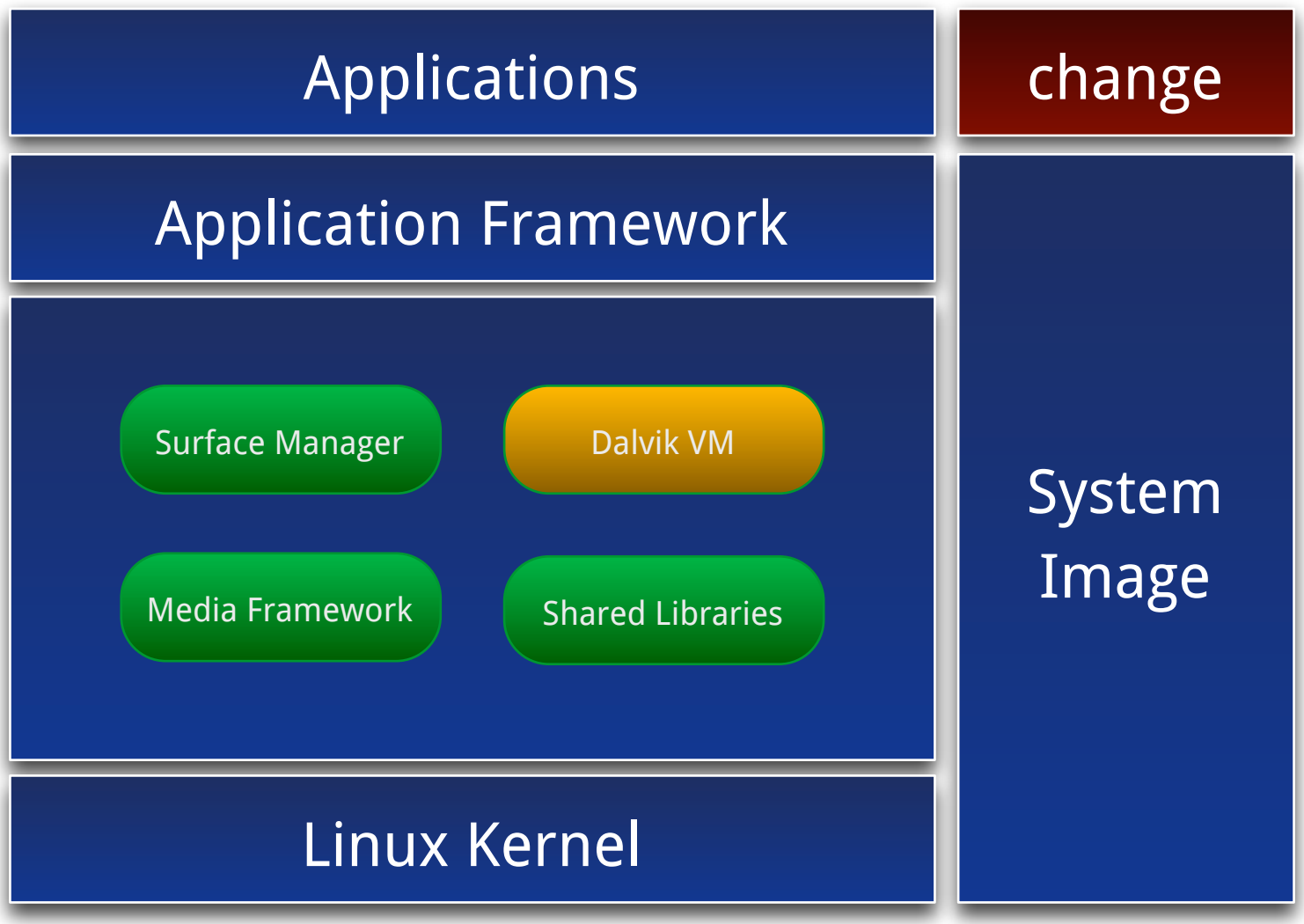
-rw-r--r--	.gitignore	blob | history | raw
-rw-r--r--	.mailmap	blob | history | raw
-rw-r--r--	COPYING	blob | history | raw
-rw-r--r--	CREDITS	blob | history | raw
drwxr-xr-x	Documentation	tree | history
-rw-r--r--	Kbuild	blob | history | raw
-rw-r--r--	MAINTAINERS	blob | history | raw
-rw-r--r--	Makefile	blob | history | raw
-rw-r--r--	README	blob | history | raw
-rw-r--r--	REPORTING-BUGS	blob | history | raw
drwxr-xr-x	arch	tree | history
drwxr-xr-x	block	tree | history
drwxr-xr-x	crypto	tree | history
drwxr-xr-x	drivers	tree | history

Android Toolchain

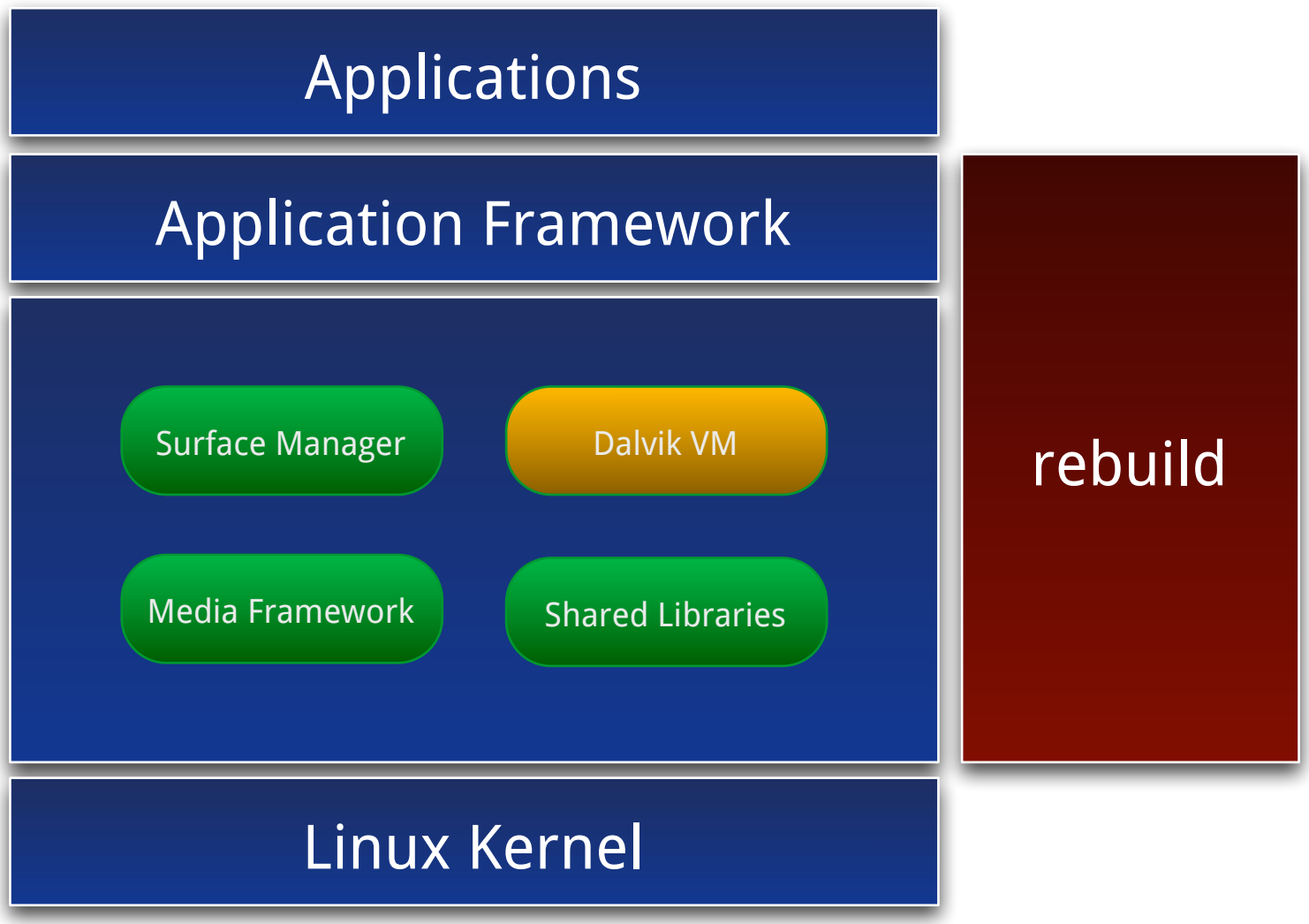
- android-toolchain-20081019
  - http://android.git.kernel.org/pub/
- gcc 4.2.1 / binutils 2.17 / gdb 6.6
- ARMV5te+
- ABI: EABI, AAPCS
- --with-float=soft --with=fpu-vfp
- --enable-threads (single)


Applications
Home
Contacts
Phone
Browser
...
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Notification Manager
Package Manager
Telephony Manager
Resource Manager
Location Manager
GTalk Service
Libraries
Surface Manager
Media Framework
SQLite
OpenGL | ES
FreeType
WebKit
SGL
SSL
libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Bluetooth Driver
Flash Memory Driver
Binder (IPC) Driver
USB Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management

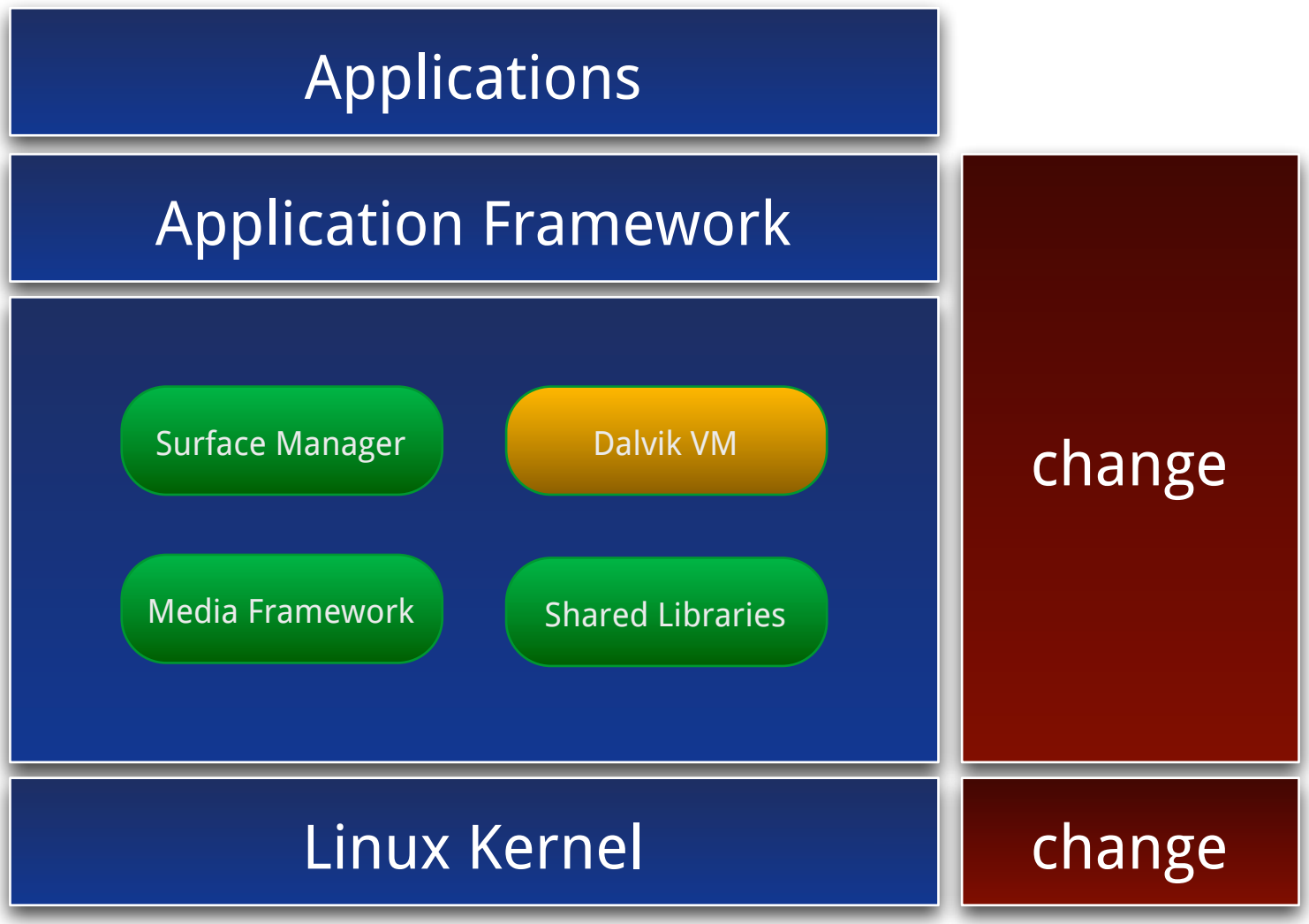
Application Developers



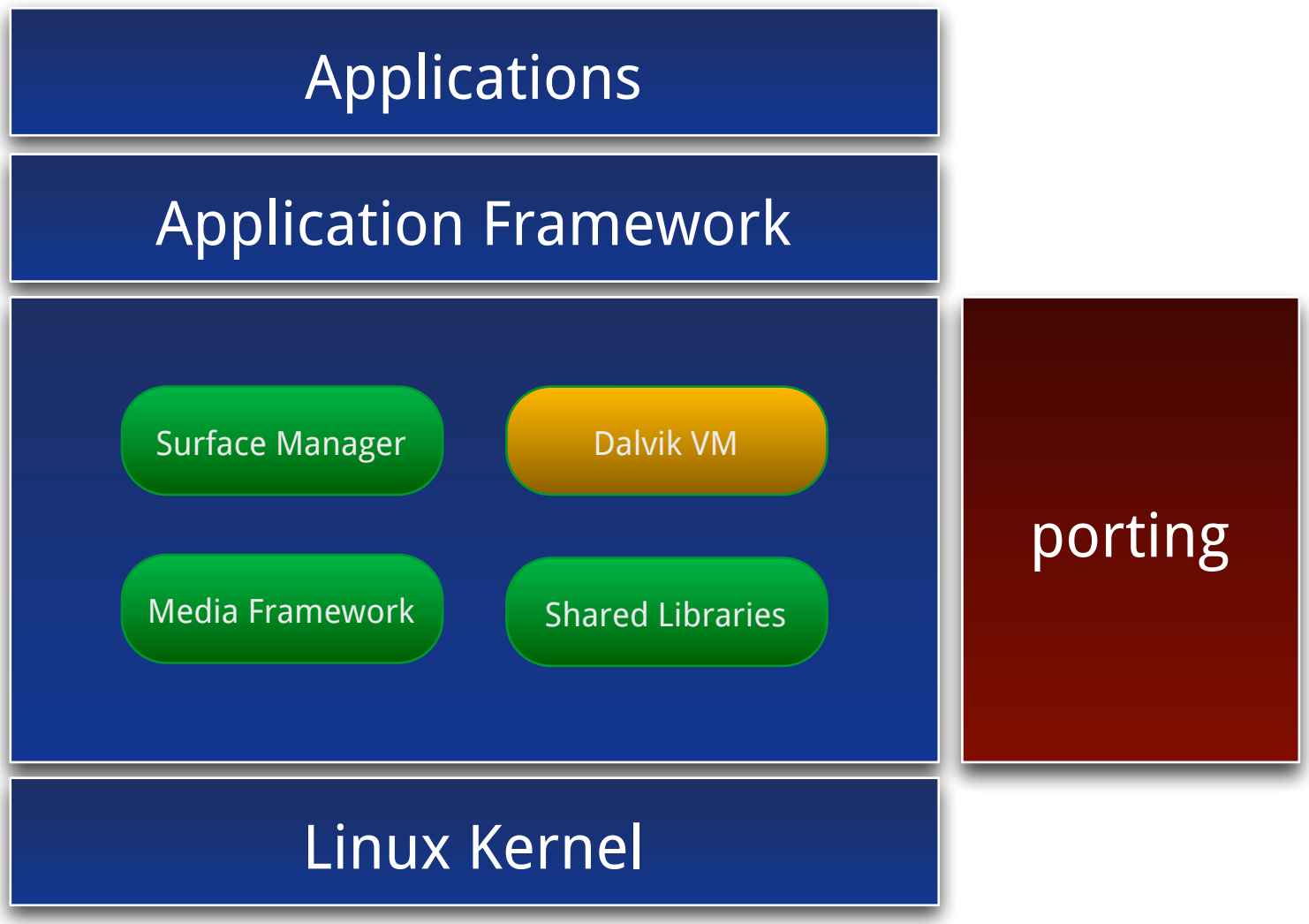
Product Branch Maintainer



Hardware Developing



Architecture Porting



- Application Developers

- Application Developers
- Product Branch Maintainer

- Application Developers
- Product Branch Maintainer
- Hardware Developing

- Application Developers
- Product Branch Maintainer
- Hardware Developing
- Architecture Porting

Android Framework 效能問題

Android Framework
效能問題

框架的重要性？

- The best way to write good, efficient code for embedded systems is to understand what the code you write really does. --Android Dev Guide

```
    private SensorManager mSensorManager;

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        mSensorManager = (SensorManager) getSystemService
(SENSOR_SERVICE);
    }
```

```
    private SensorManager mSensorManager;

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        mSensorManager = (SensorManager) getSystemService
(SENSOR_SERVICE);
    }



    private SensorManager mSensorManager;

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        mSensorManager = (SensorManager) getSystemService
(SENSOR_SERVICE);

        setContentView(R.layout.main);
    }
```

```
private SensorManager mSensorManager;

public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    畫UI;

    取得SensorManager;
}
```

```
private SensorManager mSensorManager;

public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    畫UI;

    取得SensorManager;
}
```

```
private SensorManager mSensorManager;

public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    取得SensorManager;

    畫UI;
}
```

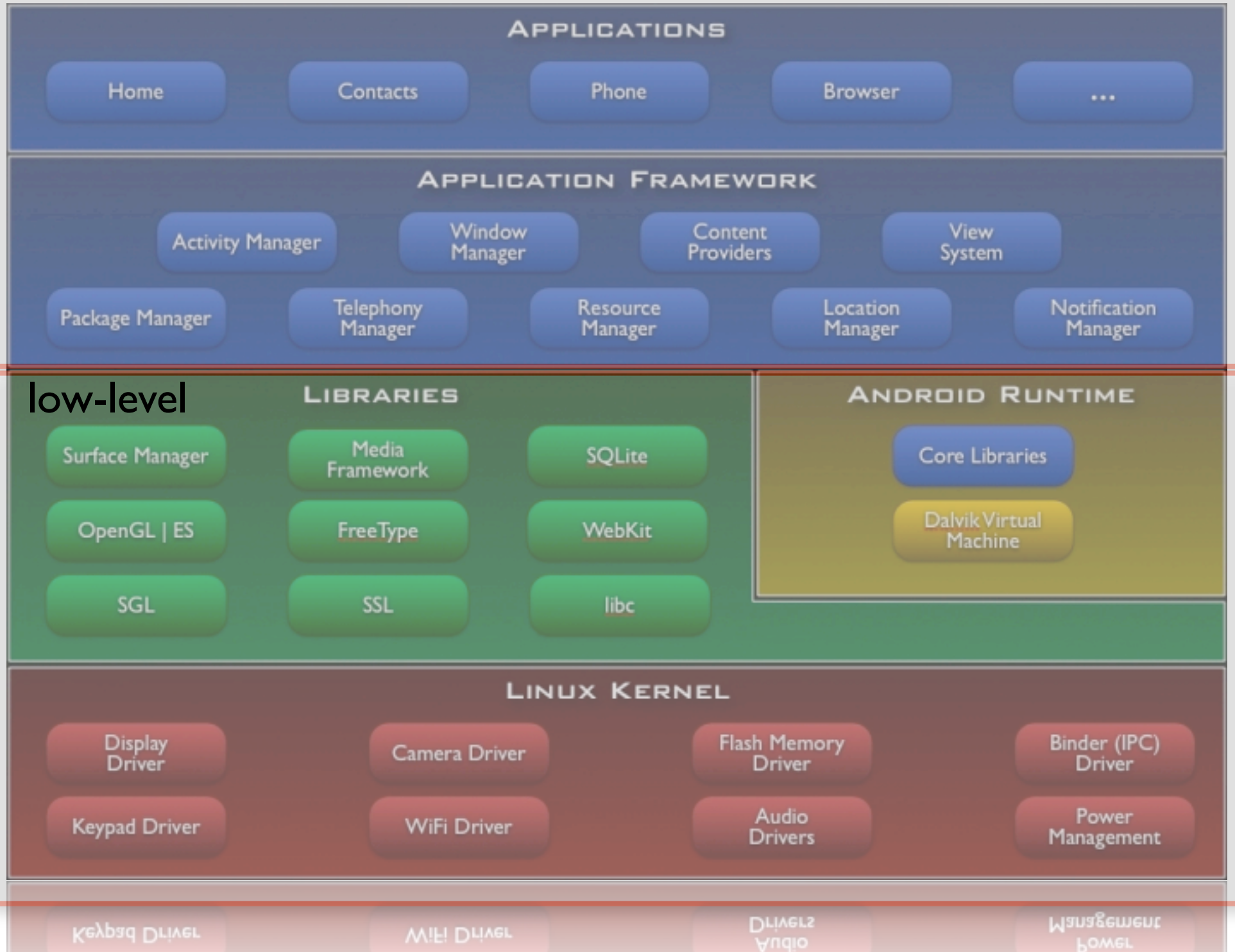
- 懂底層嗎？

```
public class TunitBill extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        Debug.startMethodTracing("TunitB");

        CharSequence time = Long.toString(SystemClock.elapsedRealtime());

        Debug.stopMethodTracing();
    }
}
```

```
public class TunitBill extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        Debug.startMethodTracing("TunitB");

        CharSequence time = Long.toString(SystemClock.elapsedRealtime());

        Debug.stopMethodTracing();
    }
}
```

- 在物件導向的模式裡的效能、設計快速或有效率的程式碼、多數工作落在 method-level

method-level



Java Profiler for Intelligent Code Profiling

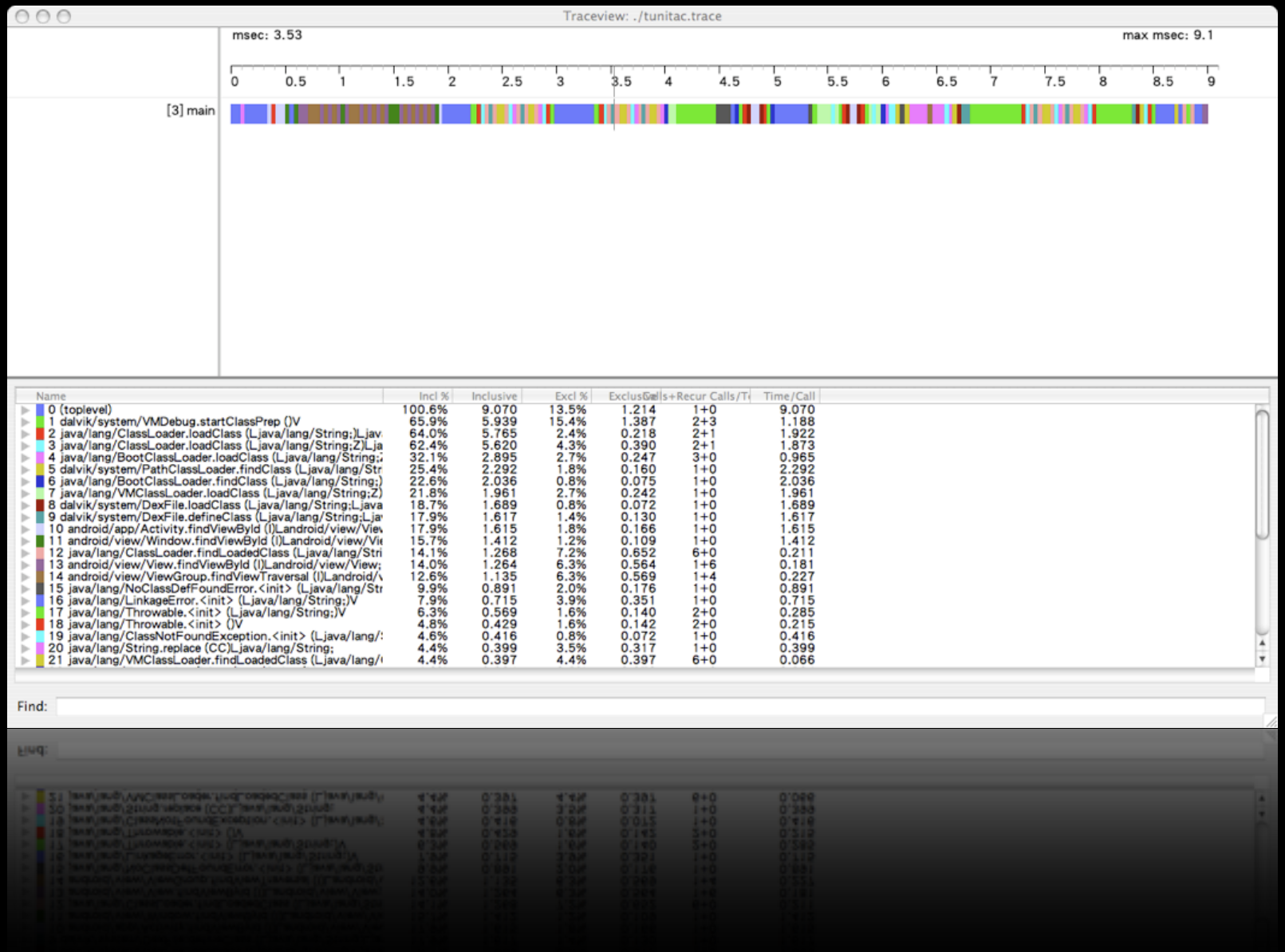
Resolve Performance Problems Faster with JProbe

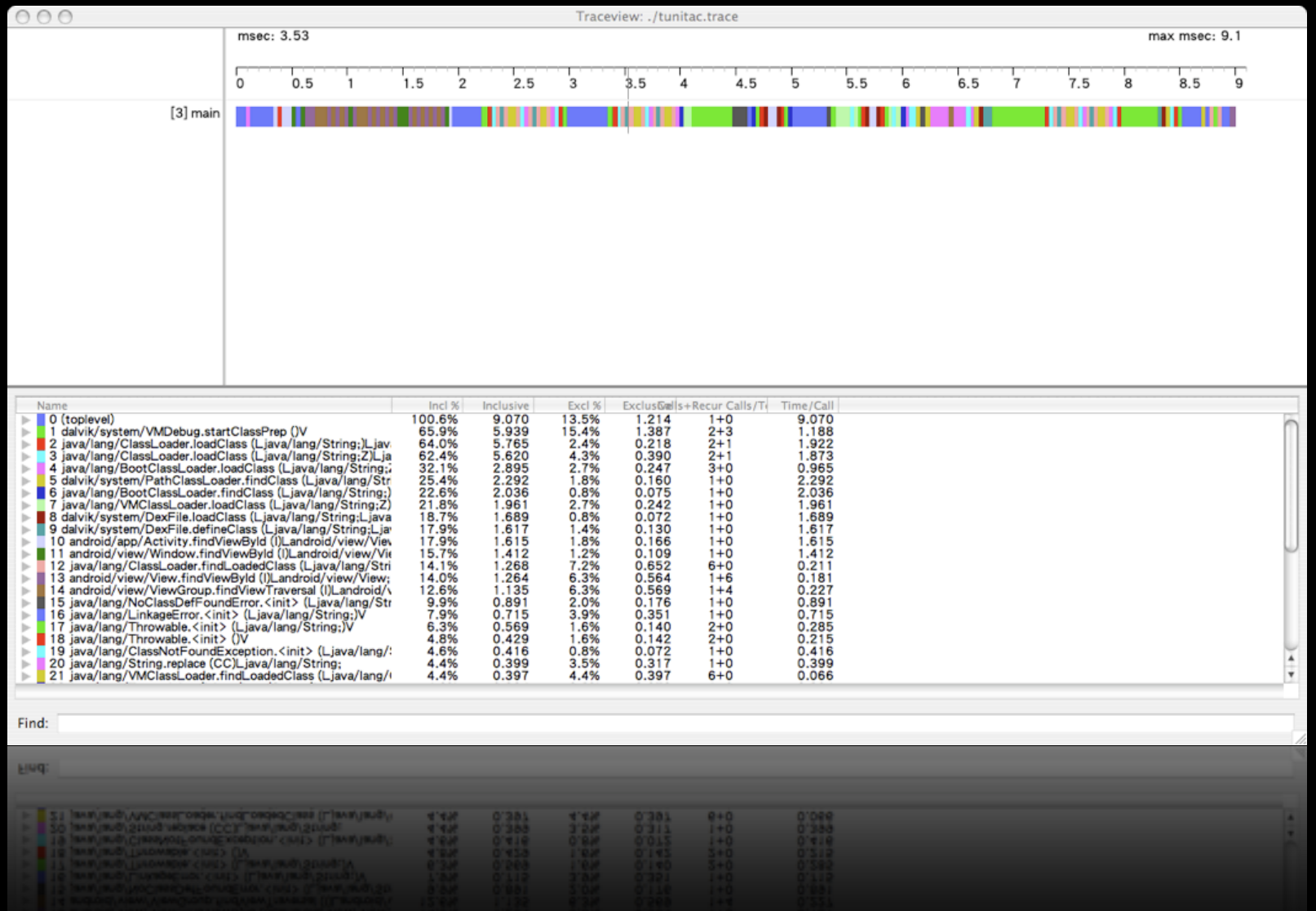
JProbe

JProbe is an enterprise-class **Java profiler** providing intelligent diagnostics on memory usage, performance and test coverage, allowing developers to quickly pinpoint and repair the root cause of application code performance and stability problems that obstruct component and integration integrity.

JProbe also offers an **Eclipse plug-in** that provides intelligent code performance analysis and problem resolution directly within the Eclipse Java IDE.

- **Memory Analysis** – Identify and resolve **Java memory leaks and object cycling** to ensure optimal program efficiency and stability. Learn more
- **Performance Analysis** – Identify and resolve **Java bottlenecks and deadlocks** to ensure optimal program performance and scalability. Learn more
- **Coverage Analysis** – Identify **unexecuted lines of code** during unit testing to ensure test coverage and program correctness. Learn more


Traceview: ./tunitac.trace
msec: 3.53
max msec: 9.1
0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5 5.5 6 6.5 7 7.5 8 8.5 9
[3] main
Name Incl % Inclusive Excl % Exclusive Calls+Recur Calls/T Time/Call
0 (toplevel) 100.6% 9.070 13.5% 1.214 1+0 9.070
1 dalvik/system/VMDebug.startClassPrep ()V 65.9% 5.939 15.4% 1.387 2+3 1.188
2 java/lang/ClassLoader.loadClass (Ljava/lang/String;)Ljav 64.0% 5.765 2.4% 0.218 2+1 1.922
3 java/lang/ClassLoader.loadClass (Ljava/lang/String;Z)Lja 62.4% 5.620 4.3% 0.390 2+1 1.873
4 java/lang/BootClassLoader.loadClass (Ljava/lang/String;; 32.1% 2.895 2.7% 0.247 3+0 0.965
5 dalvik/system/PathClassLoader.findClass (Ljava/lang/Str 25.4% 2.292 1.8% 0.160 1+0 2.292
6 java/lang/BootClassLoader.findClass (Ljava/lang/String;) 22.6% 2.036 0.8% 0.075 1+0 2.036
7 java/lang/VMClassLoader.loadClass (Ljava/lang/String;Z) 21.8% 1.961 2.7% 0.242 1+0 1.961
8 dalvik/system/DexFile.loadClass (Ljava/lang/String;Ljava 18.7% 1.689 0.8% 0.072 1+0 1.689
9 dalvik/system/DexFile.defineClass (Ljava/lang/String;Lja 17.9% 1.617 1.4% 0.130 1+0 1.617
10 android/app/Activity.findViewById (I)Landroid/view/Vie 17.9% 1.615 1.8% 0.166 1+0 1.615
11 android/view/Window.findViewById (I)Landroid/view/Vi 15.7% 1.412 1.2% 0.109 1+0 1.412
12 java/lang/ClassLoader.findLoadedClass (Ljava/lang/Stri 14.1% 1.268 7.2% 0.652 6+0 0.211
13 android/view/View.findViewById (I)Landroid/view/View; 14.0% 1.264 6.3% 0.564 1+6 0.181
14 android/view/ViewGroup.findViewTraversal (I)Landroid/ 12.6% 1.135 6.3% 0.569 1+4 0.227
15 java/lang/NoClassDefFoundError.<init> (Ljava/lang/Str 9.9% 0.891 2.0% 0.176 1+0 0.891
16 java/lang/LinkageError.<init> (Ljava/lang/String;)V 7.9% 0.715 3.9% 0.351 1+0 0.715
17 java/lang/Throwable.<init> (Ljava/lang/String;)V 6.3% 0.569 1.6% 0.140 2+0 0.285
18 java/lang/Throwable.<init> ()V 4.8% 0.429 1.6% 0.142 2+0 0.215
19 java/lang/ClassNotFoundException.<init> (Ljava/lang/ 4.6% 0.416 0.8% 0.072 1+0 0.416
20 java/lang/String.replace (CC)Ljava/lang/String; 4.4% 0.399 3.5% 0.317 1+0 0.399
21 java/lang/VMClassLoader.findLoadedClass (Ljava/lang/ 4.4% 0.397 4.4% 0.397 6+0 0.066
Find:



- 使用 TraceView 進行 method 追蹤 (method tracing)

Avoid Creating Objects
Use Native Methods
Prefer Virtual Over Interface
Prefer Static Over Virtual
Avoid Internal Getters/Setters
Cache Field Lookups
Declare Constants Final
Use Enhanced For Loop Syntax With Caution
Avoid Enums
Use Package Scope with Inner Classes
Avoid Float

```
Avoid Creating Objects
Use Native Methods
Prefer Virtual Over Interface
Prefer Static Over Virtual
Avoid Internal Getters/Setters
Cache Field Lookups
Declare Constants Final
Use Enhanced For Loop Syntax With Caution
Avoid Enums
Use Package Scope with Inner Classes
Avoid Float
```

- Designing for Performance -- Android Dev Guide

Android Tracing API

android.os.Debug

```
Debug.startMethodTracing("TunitB");

/* writing codes to profile */

Debug.stopMethodTracing();
```

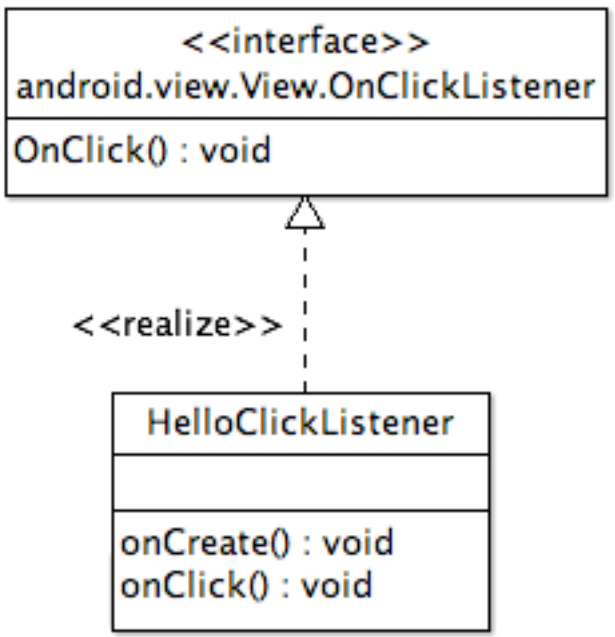
Android Tracing API

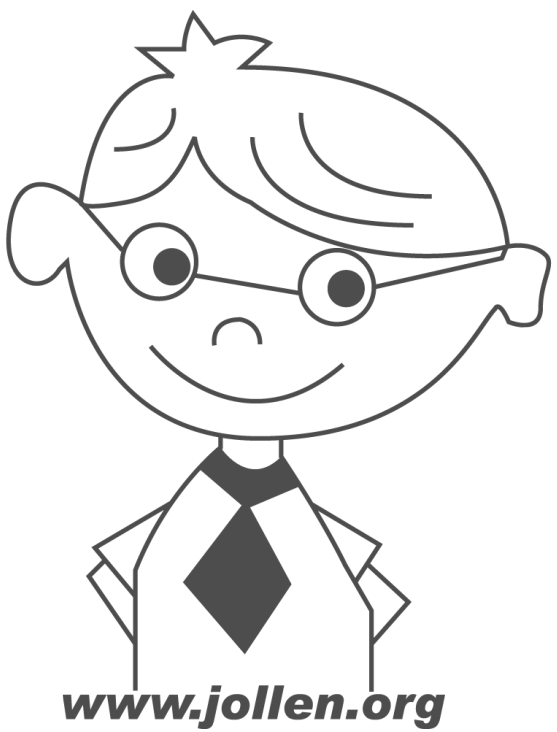
android.os.Debug

```
public class TunitChris extends Activity  {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        Button button = (Button)findViewById(R.id.btn);

        button.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                // Perform action on click
            }
        });

    }
}
```

- 避免建立物件

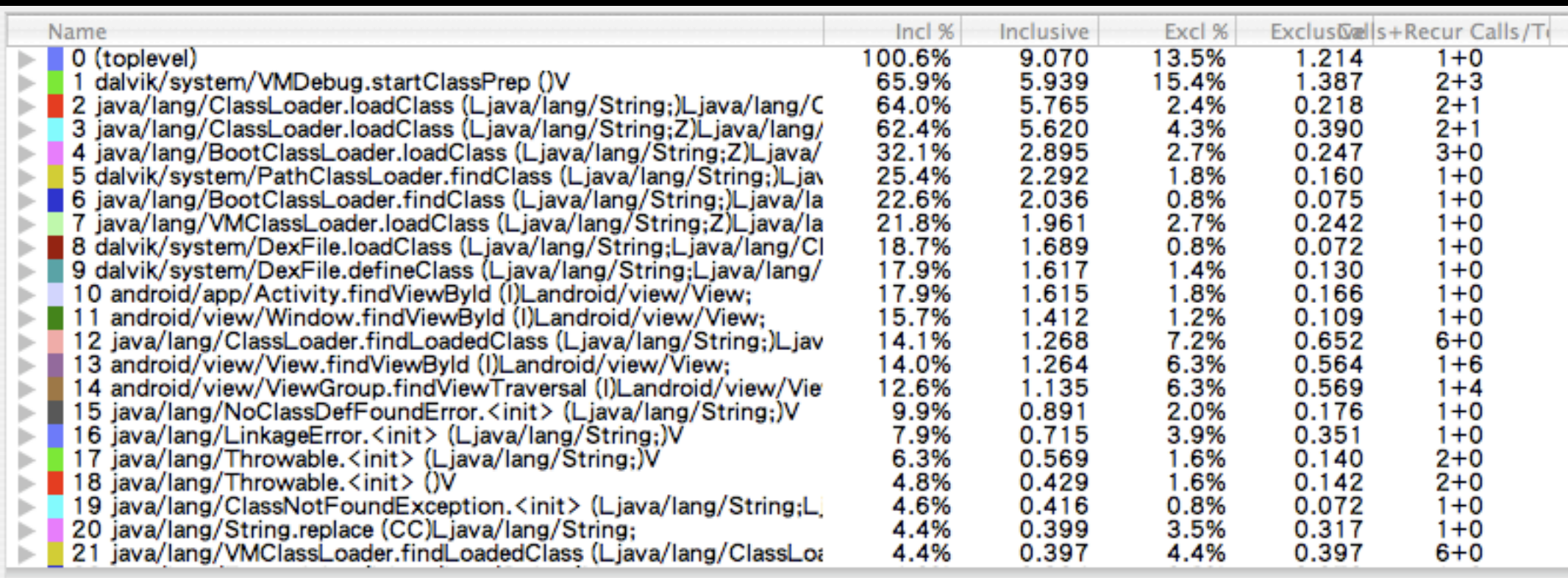
良好的Click Listener實作





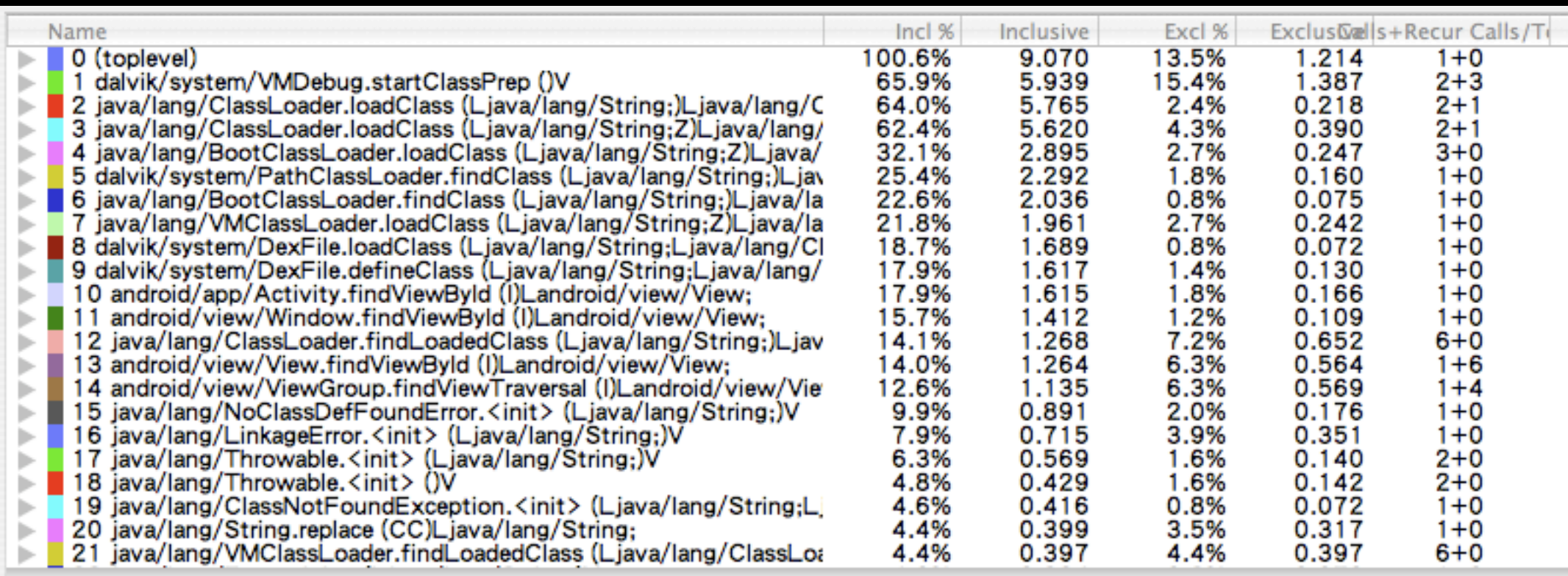
☑一個較為良好的實作方法是在我們的Acitivty類別裡實作View.OnClickListener介面

```
public class TunitChris extends Activity implements
View.OnClickListener {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        Button button = (Button)findViewById(R.id.btn);
        button.setOnClickListener(this);
    }

    public void onClick(View v) {
        Toast.makeText(
                this,
                "Yes.",
                Toast.LENGTH_LONG).show();
    }
}
```

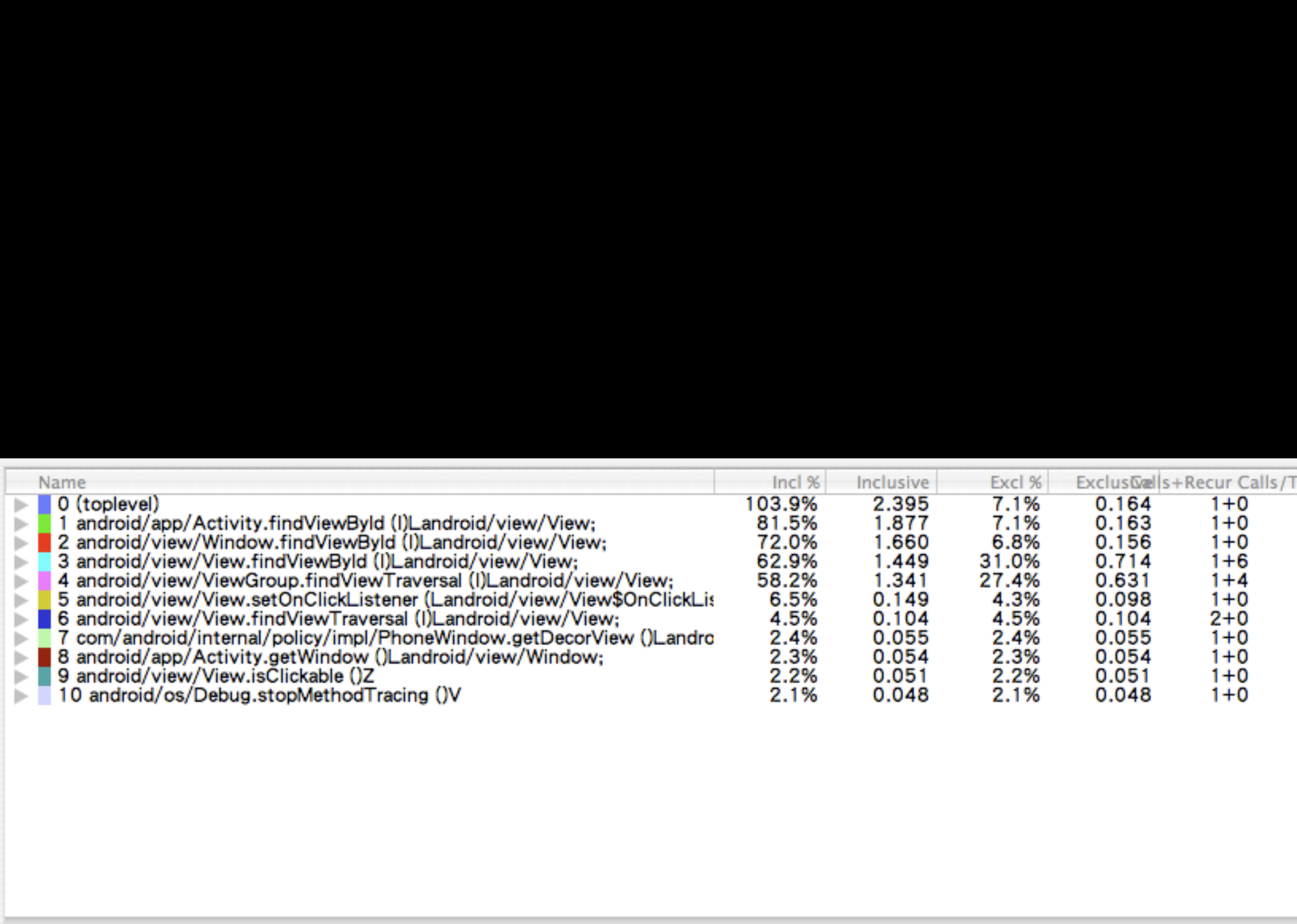


Name	Incl %	Inclusive	Excl %	Exclusive	Calls+Recur Calls/T
0 (toplevel)	100.6%	9.070	13.5%	1.214	1+0
1 dalvik/system/VMDebug.startClassPrep ()V	65.9%	5.939	15.4%	1.387	2+3
2 java/lang/ClassLoader.loadClass (Ljava/lang/String;)Ljava/lang/C	64.0%	5.765	2.4%	0.218	2+1
3 java/lang/ClassLoader.loadClass (Ljava/lang/String;Z)Ljava/lang/	62.4%	5.620	4.3%	0.390	2+1
4 java/lang/BootClassLoader.loadClass (Ljava/lang/String;Z)Ljava/	32.1%	2.895	2.7%	0.247	3+0
5 dalvik/system/PathClassLoader.findClass (Ljava/lang/String;)Ljav	25.4%	2.292	1.8%	0.160	1+0
6 java/lang/BootClassLoader.findClass (Ljava/lang/String;)Ljava/la	22.6%	2.036	0.8%	0.075	1+0
7 java/lang/VMClassLoader.loadClass (Ljava/lang/String;Z)Ljava/la	21.8%	1.961	2.7%	0.242	1+0
8 dalvik/system/DexFile.loadClass (Ljava/lang/String;Ljava/lang/Cl	18.7%	1.689	0.8%	0.072	1+0
9 dalvik/system/DexFile.defineClass (Ljava/lang/String;Ljava/lang/	17.9%	1.617	1.4%	0.130	1+0
10 android/app/Activity.findViewById (I)Landroid/view/View;	17.9%	1.615	1.8%	0.166	1+0
11 android/view/Window.findViewById (I)Landroid/view/View;	15.7%	1.412	1.2%	0.109	1+0
12 java/lang/ClassLoader.findLoadedClass (Ljava/lang/String;)Ljav	14.1%	1.268	7.2%	0.652	6+0
13 android/view/View.findViewById (I)Landroid/view/View;	14.0%	1.264	6.3%	0.564	1+6
14 android/view/ViewGroup.findViewTraversal (I)Landroid/view/Vie	12.6%	1.135	6.3%	0.569	1+4
15 java/lang/NoClassDefFoundError.<init> (Ljava/lang/String;)V	9.9%	0.891	2.0%	0.176	1+0
16 java/lang/LinkageError.<init> (Ljava/lang/String;)V	7.9%	0.715	3.9%	0.351	1+0
17 java/lang/Throwable.<init> (Ljava/lang/String;)V	6.3%	0.569	1.6%	0.140	2+0
18 java/lang/Throwable.<init> ()V	4.8%	0.429	1.6%	0.142	2+0
19 java/lang/ClassNotFoundException.<init> (Ljava/lang/String;L	4.6%	0.416	0.8%	0.072	1+0
20 java/lang/String.replace (CC)Ljava/lang/String;	4.4%	0.399	3.5%	0.317	1+0
21 java/lang/VMClassLoader.findLoadedClass (Ljava/lang/ClassLoa	4.4%	0.397	4.4%	0.397	6+0

```
button.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        // Perform action on click
    }
});
```



Name	Incl %	Inclusive	Excl %	Exclusive	Calls+Recur Calls/T
0 (toplevel)	100.6%	9.070	13.5%	1.214	1+0
1 dalvik/system/VMDebug.startClassPrep ()V	65.9%	5.939	15.4%	1.387	2+3
2 java/lang/ClassLoader.loadClass (Ljava/lang/String;)Ljava/lang/C	64.0%	5.765	2.4%	0.218	2+1
3 java/lang/ClassLoader.loadClass (Ljava/lang/String;Z)Ljava/lang/	62.4%	5.620	4.3%	0.390	2+1
4 java/lang/BootClassLoader.loadClass (Ljava/lang/String;Z)Ljava/	32.1%	2.895	2.7%	0.247	3+0
5 dalvik/system/PathClassLoader.findClass (Ljava/lang/String;)Ljav	25.4%	2.292	1.8%	0.160	1+0
6 java/lang/BootClassLoader.findClass (Ljava/lang/String;)Ljava/la	22.6%	2.036	0.8%	0.075	1+0
7 java/lang/VMClassLoader.loadClass (Ljava/lang/String;Z)Ljava/la	21.8%	1.961	2.7%	0.242	1+0
8 dalvik/system/DexFile.loadClass (Ljava/lang/String;Ljava/lang/Cl	18.7%	1.689	0.8%	0.072	1+0
9 dalvik/system/DexFile.defineClass (Ljava/lang/String;Ljava/lang/	17.9%	1.617	1.4%	0.130	1+0
10 android/app/Activity.findViewById (I)Landroid/view/View;	17.9%	1.615	1.8%	0.166	1+0
11 android/view/Window.findViewById (I)Landroid/view/View;	15.7%	1.412	1.2%	0.109	1+0
12 java/lang/ClassLoader.findLoadedClass (Ljava/lang/String;)Ljav	14.1%	1.268	7.2%	0.652	6+0
13 android/view/View.findViewById (I)Landroid/view/View;	14.0%	1.264	6.3%	0.564	1+6
14 android/view/ViewGroup.findViewTraversal (I)Landroid/view/Vie	12.6%	1.135	6.3%	0.569	1+4
15 java/lang/NoClassDefFoundError.<init> (Ljava/lang/String;)V	9.9%	0.891	2.0%	0.176	1+0
16 java/lang/LinkageError.<init> (Ljava/lang/String;)V	7.9%	0.715	3.9%	0.351	1+0
17 java/lang/Throwable.<init> (Ljava/lang/String;)V	6.3%	0.569	1.6%	0.140	2+0
18 java/lang/Throwable.<init> ()V	4.8%	0.429	1.6%	0.142	2+0
19 java/lang/ClassNotFoundException.<init> (Ljava/lang/String;L	4.6%	0.416	0.8%	0.072	1+0
20 java/lang/String.replace (CC)Ljava/lang/String;	4.4%	0.399	3.5%	0.317	1+0
21 java/lang/VMClassLoader.findLoadedClass (Ljava/lang/ClassLoa	4.4%	0.397	4.4%	0.397	6+0



Name	Incl %	Inclusive	Excl %	Exclusive	Calls+Recur Calls/T
0 (toplevel)	103.9%	2.395	7.1%	0.164	1+0
1 android/app/Activity.findViewByld (I)Landroid/view/View;	81.5%	1.877	7.1%	0.163	1+0
2 android/view/Window.findViewByld (I)Landroid/view/View;	72.0%	1.660	6.8%	0.156	1+0
3 android/view/View.findViewByld (I)Landroid/view/View;	62.9%	1.449	31.0%	0.714	1+6
4 android/view/ViewGroup.findViewTraversal (I)Landroid/view/View;	58.2%	1.341	27.4%	0.631	1+4
5 android/view/View.setOnClickListener (Landroid/view/View$OnClickLis	6.5%	0.149	4.3%	0.098	1+0
6 android/view/View.findViewTraversal (I)Landroid/view/View;	4.5%	0.104	4.5%	0.104	2+0
7 com/android/internal/policy/impl/PhoneWindow.getDecorView ()Landro	2.4%	0.055	2.4%	0.055	1+0
8 android/app/Activity.getWindow ()Landroid/view/Window;	2.3%	0.054	2.3%	0.054	1+0
9 android/view/View.isClickable ()Z	2.2%	0.051	2.2%	0.051	1+0
10 android/os/Debug.stopMethodTracing ()V	2.1%	0.048	2.1%	0.048	1+0

```
public class TunitChris extends Activity implements
View.OnClickListener  { }
```

Name	Incl %	Inclusive	Excl %	Exclusive	Calls+Recur Calls/T
0 (toplevel)	103.9%	2.395	7.1%	0.164	1+0
1 android/app/Activity.findViewByld (I)Landroid/view/View;	81.5%	1.877	7.1%	0.163	1+0
2 android/view/Window.findViewByld (I)Landroid/view/View;	72.0%	1.660	6.8%	0.156	1+0
3 android/view/View.findViewByld (I)Landroid/view/View;	62.9%	1.449	31.0%	0.714	1+6
4 android/view/ViewGroup.findViewTraversal (I)Landroid/view/View;	58.2%	1.341	27.4%	0.631	1+4
5 android/view/View.setOnClickListener (Landroid/view/View$OnClickLis	6.5%	0.149	4.3%	0.098	1+0
6 android/view/View.findViewTraversal (I)Landroid/view/View;	4.5%	0.104	4.5%	0.104	2+0
7 com/android/internal/policy/impl/PhoneWindow.getDecorView ()Landro	2.4%	0.055	2.4%	0.055	1+0
8 android/app/Activity.getWindow ()Landroid/view/Window;	2.3%	0.054	2.3%	0.054	1+0
9 android/view/View.isClickable ()Z	2.2%	0.051	2.2%	0.051	1+0
10 android/os/Debug.stopMethodTracing ()V	2.1%	0.048	2.1%	0.048	1+0

- 與硬體相關

如何使用 SensorManager ?

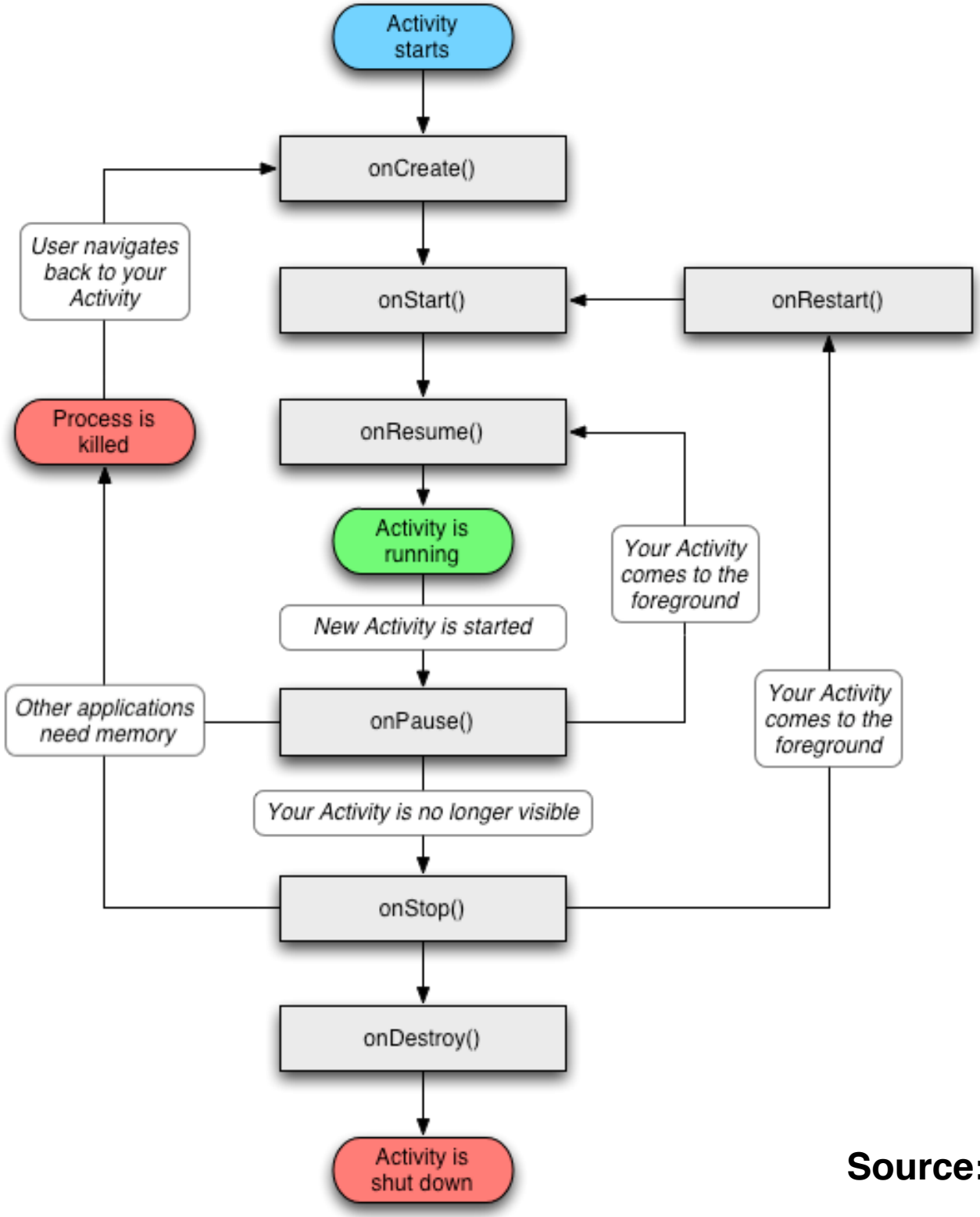
```
    private SensorManager mSensorManager;

     public void onCreate(Bundle savedInstanceState) {
         super.onCreate(savedInstanceState);
         setContentView(R.layout.main);

         mSensorManager = (SensorManager) getSystemService
(SENSOR_SERVICE);
     }
```

如何使用 SensorManager ?

```
    private SensorManager mSensorManager;

     public void onCreate(Bundle savedInstanceState) {
         super.onCreate(savedInstanceState);
         setContentView(R.layout.main);
     }

     public void onStop() {
         super.onStop();

         mSensorManager = (SensorManager) getSystemService
(SENSOR_SERVICE);
     }
```

```
    private SensorManager mSensorManager;

     public void onCreate(Bundle savedInstanceState) {
         super.onCreate(savedInstanceState);
         setContentView(R.layout.main);
     }

     public void onStop() {
         super.onStop();

         mSensorManager = (SensorManager) getSystemService
(SENSOR_SERVICE);
     }
```

better ?



Source: Android SDK

實際應用上、可以對Launcher做最佳化

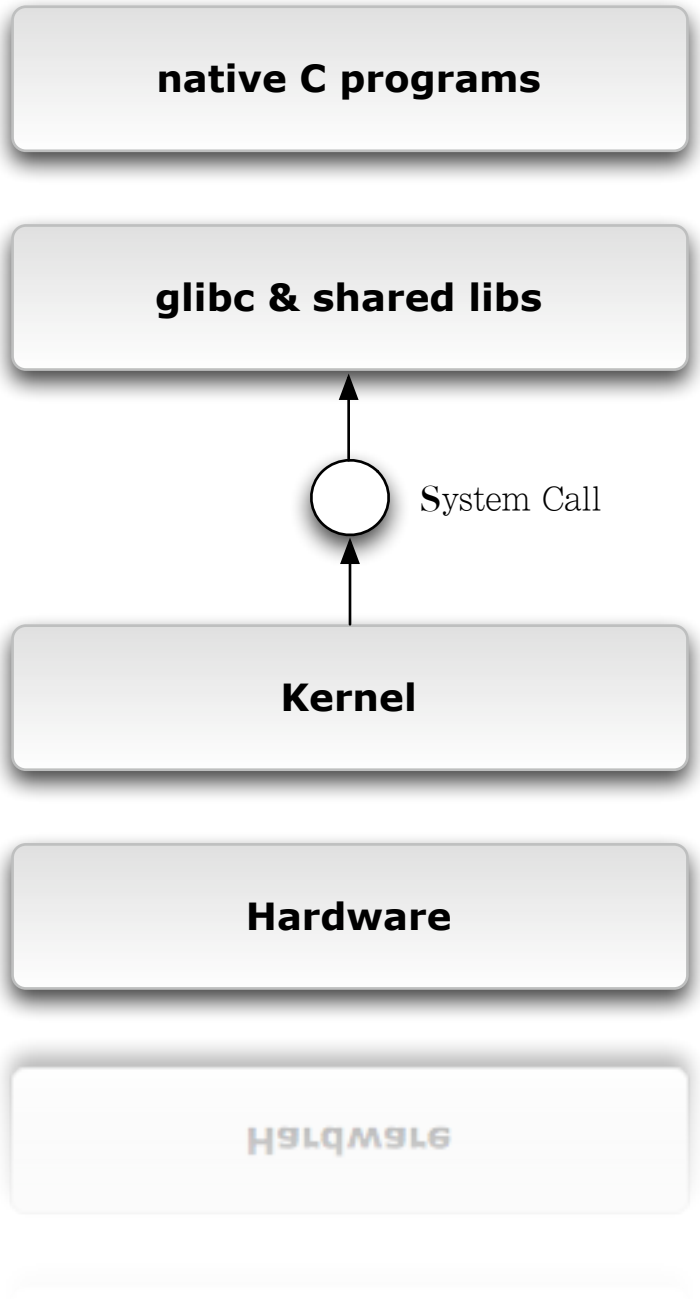
框架怎麼用

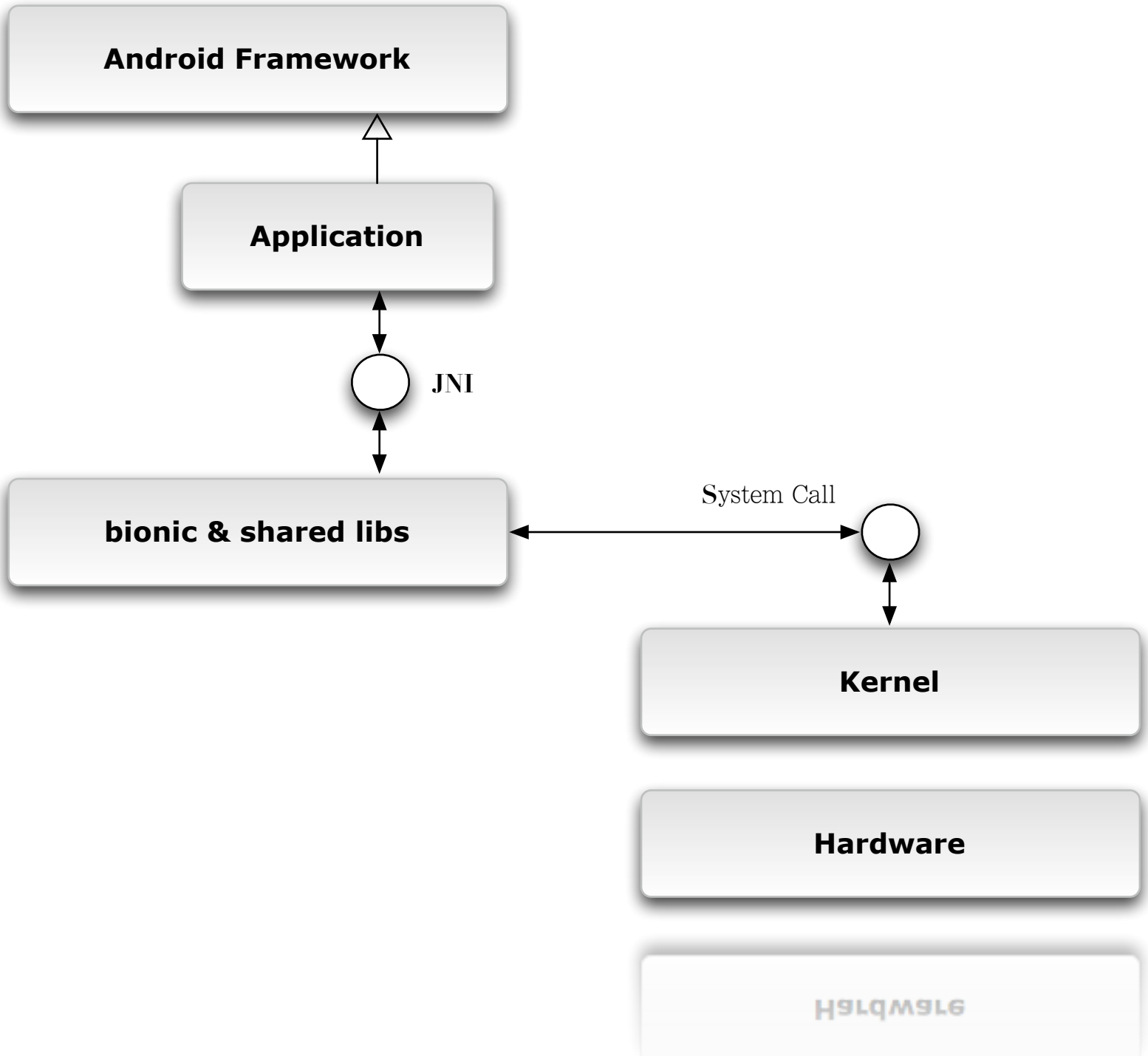
做好 method-level 的工作？
做好軟硬整合的工作？

Android Driver Model 與HAL

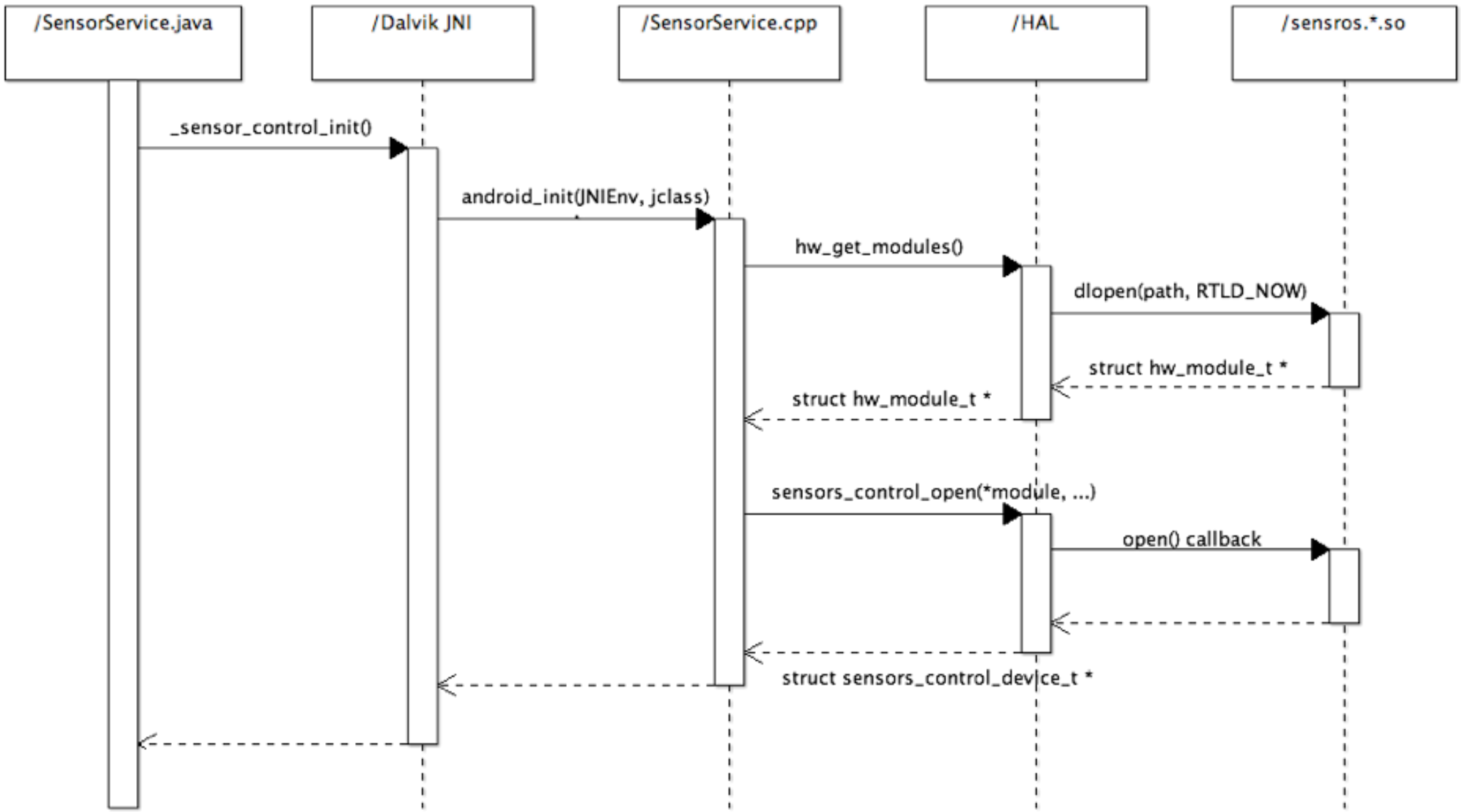
Android Driver Model 與HAL

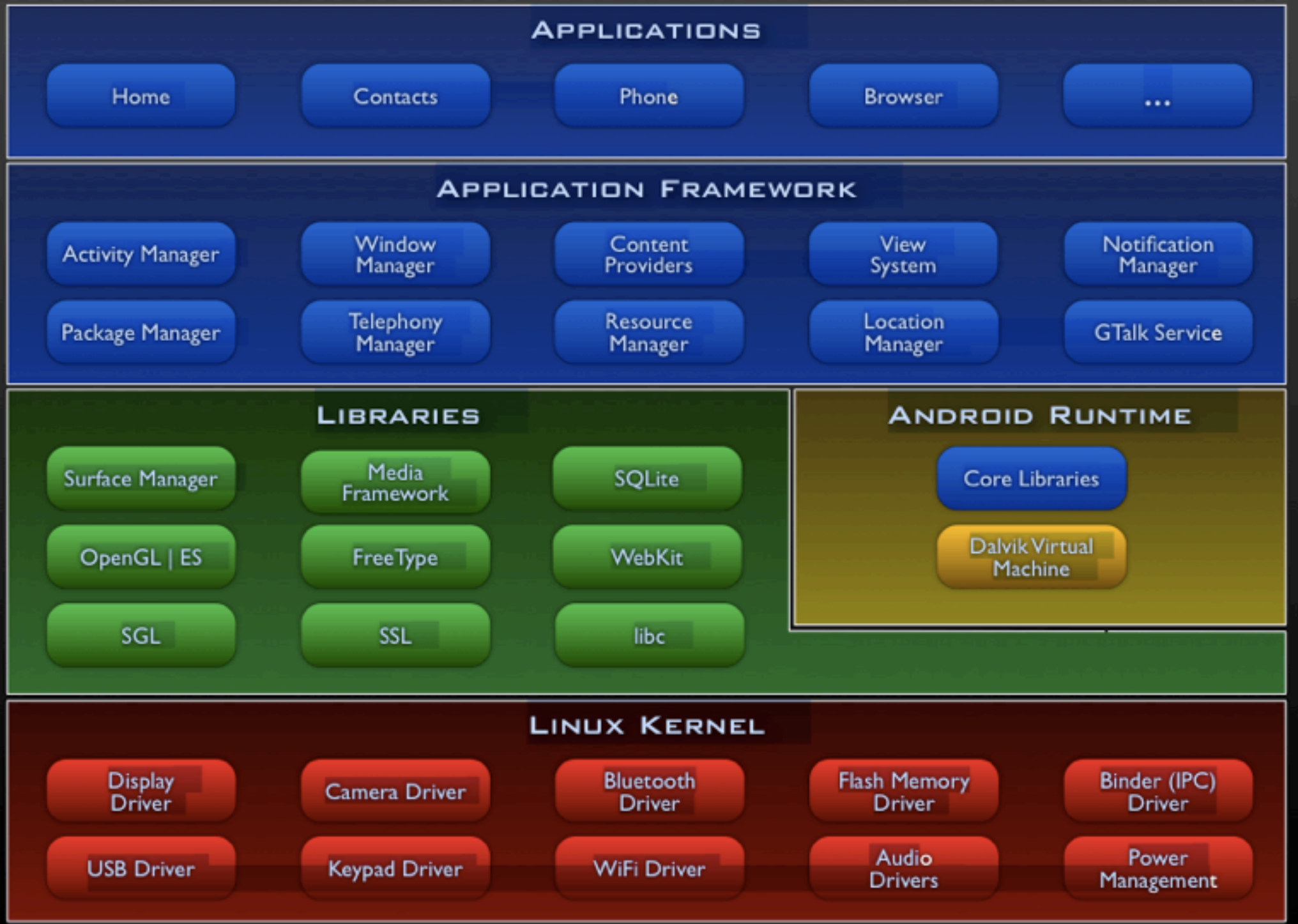
典型的學科整合


native C programs
glibc & shared libs
System Call
Kernel
Hardware


Android Framework
Application
JNI
bionic & shared libs
System Call
Kernel
Hardware

```
public class Sensors extends Activity {
    ...
    private SensorManager mSensorManager;
    ...
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        // Be sure to call the super class.
        super.onCreate(savedInstanceState);

        mSensorManager = (SensorManager) getSystemService
(SENSOR_SERVICE);
    }
    ...
}
```


/SensorService.java
/Dalvik JNI
/SensorService.cpp
/HAL
/sensros.*.so
_sensor_control_init()
android_init(JNIEnv, jclass)
hw_get_modules()
dlopen(path, RTLD_NOW)
struct hw_module_t *
struct hw_module_t *
sensors_control_open(*module, ...)
open() callback
struct sensors_control_device_t *


Applications
Home
Contacts
Phone
Browser
...
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Notification Manager
Package Manager
Telephony Manager
Resource Manager
Location Manager
GTalk Service
Libraries
Surface Manager
Media Framework
SQLite
OpenGL | ES
FreeType
WebKit
SGL
SSL
libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Bluetooth Driver
Flash Memory Driver
Binder (IPC) Driver
USB Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management

```
class SensorService extends ISensorService.Stub {
    ...
    private static native int _sensors_control_init();
    private static native ParcelFileDescriptor
_sensors_control_open();
    private static native boolean
_sensors_control_activate(int sensor, boolean activate);
    private static native int _sensors_control_set_delay
(int ms);
}
```

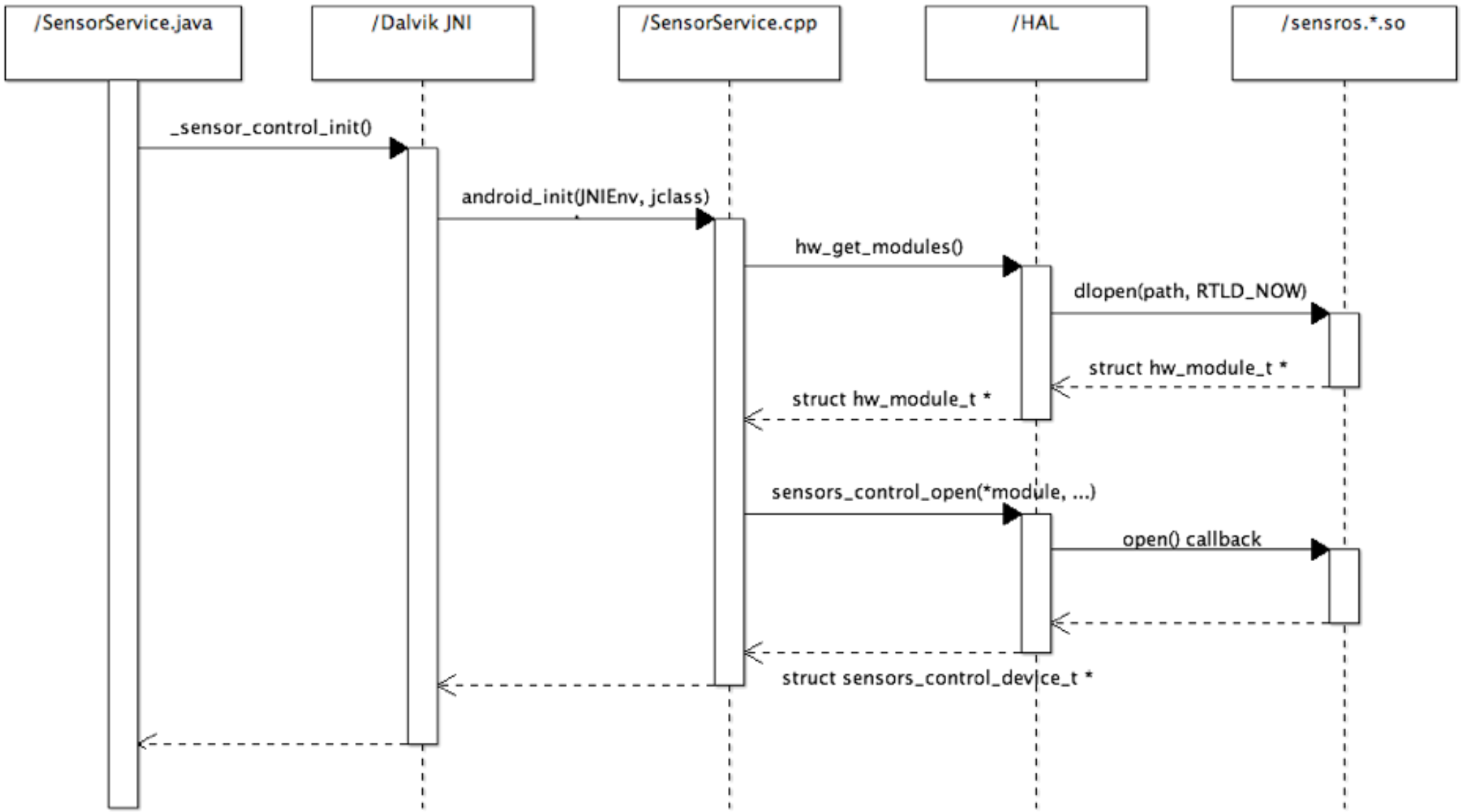
at framework/base/services/java/SensorService.java

```
class SensorService extends ISensorService.Stub {
    ...
    public SensorService(Context context) {
        if (localLOGV) Log.d(TAG, "SensorService
startup");
        _sensors_control_init();
        mNotificationManager = (NotificationManager)
context.getSystemService
(Context.NOTIFICATION_SERVICE);
        mContext = context;
    }
   ...
}
```

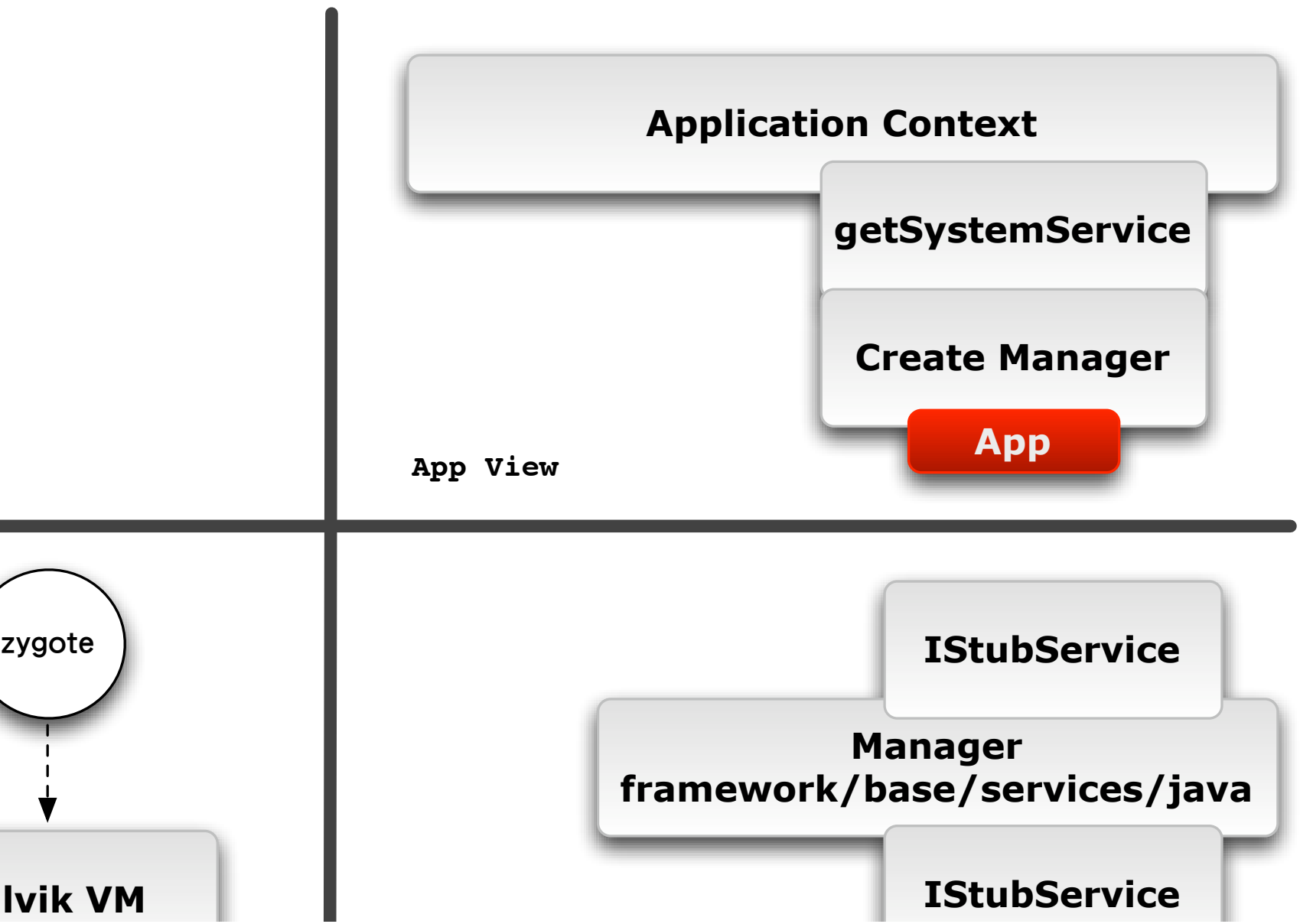
at framework/base/services/java/SensorService.java

```
static JNINativeMethod gMethods[] = {
    {"_sensors_control_init",     "()I",   (void*) android_init },
    {"_sensors_control_open",     "()Landroid/os/
ParcelFileDescriptor;",
                                           (void*) android_open },
    {"_sensors_control_activate", "(IZ)Z", (void*)
android_activate },
    {"_sensors_control_wake",     "()I", (void*)
android_data_wake },
    {"_sensors_control_set_delay","(I)I", (void*)
android_set_delay },
};
```

```
static jint
android_init(JNIEnv *env, jclass clazz)
{
    sensors_module_t* module;
    if (hw_get_module(SENSORS_HARDWARE_MODULE_ID, (const hw_module_t**)
&module) == 0) {
        if (sensors_control_open(&module->common, &sSensorDevice) == 0)
{
            const struct sensor_t* list;
            int count = module->get_sensors_list(module, &list);
            return count;
        }
    }
    return 0;
}



static inline int sensors_control_open(const struct hw_module_t*
module, struct sensors_control_device_t** device) {
    return module->methods->open(module,
            SENSORS_HARDWARE_CONTROL, (struct hw_device_t**)
device);
}
```


/SensorService.java
/Dalvik JNI
/SensorService.cpp
/HAL
/sensros.*.so
_sensor_control_init()
android_init(JNIEnv, jclass)
hw_get_modules()
dlopen(path, RTLD_NOW)
struct hw_module_t *
struct hw_module_t *
sensors_control_open(*module, ...)
open() callback
struct sensors_control_device_t *

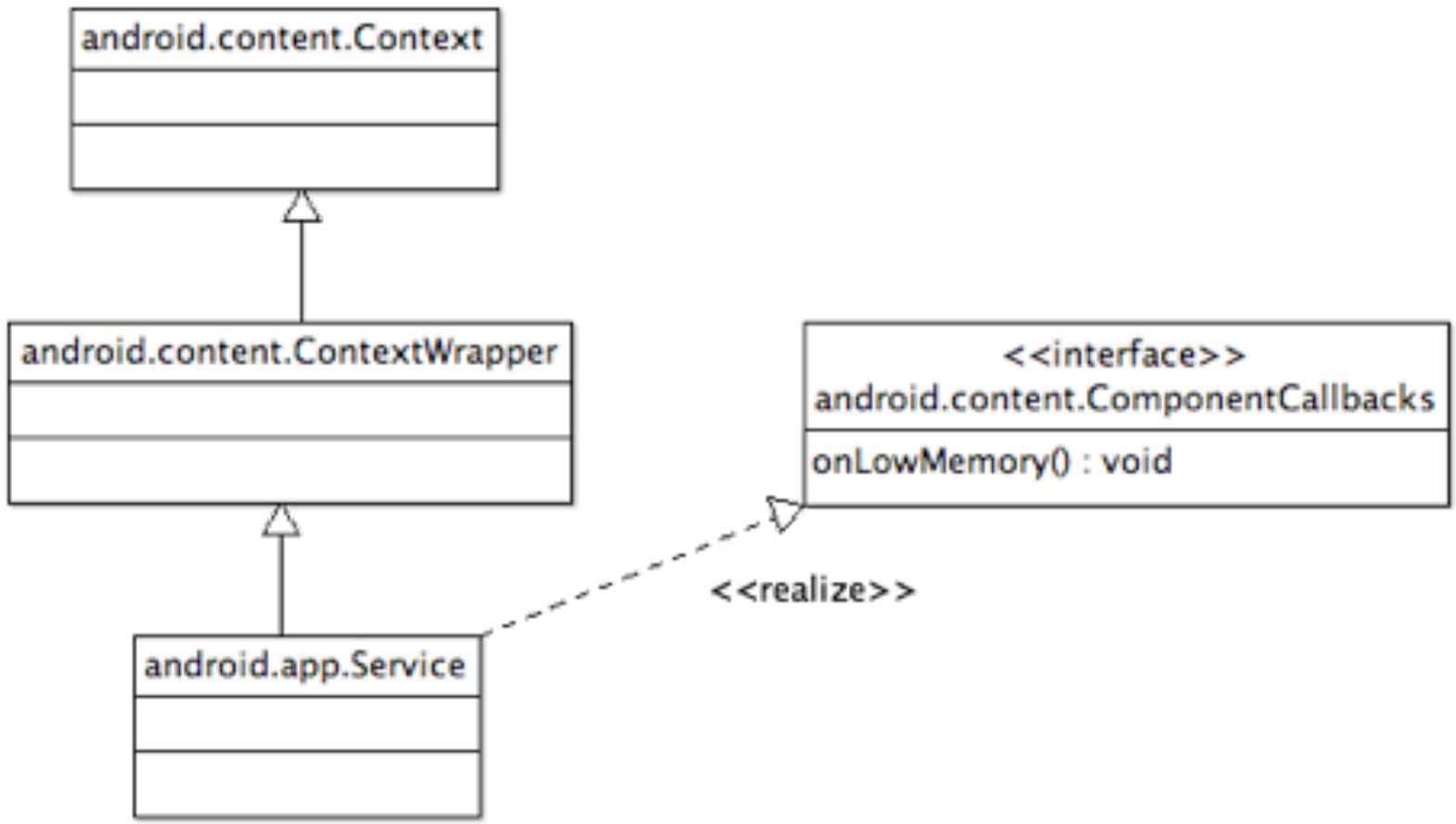
第一次比較慢


Application Context
getSystemService
Create Manager
App
App View
zygote
lvik VM
IStubService
Manager
framework/base/services/java
IStubService

如何使用 SensorManager ?

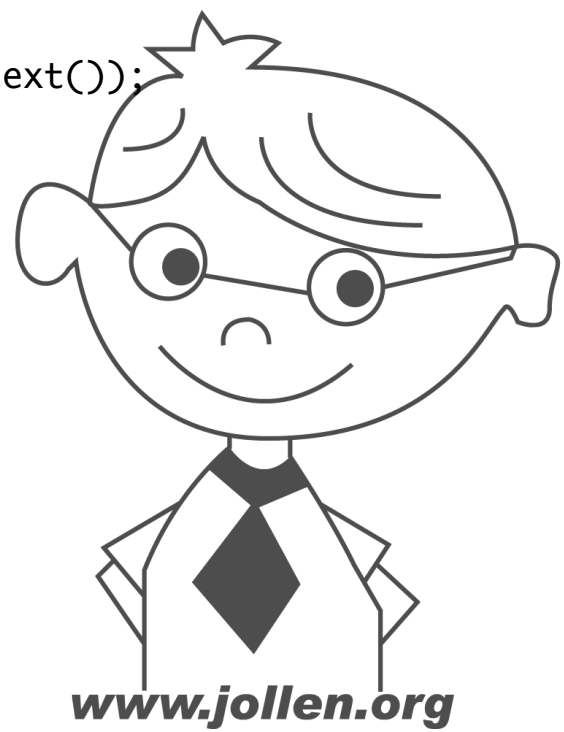
```
    private SensorManager mSensorManager;

     public void onCreate(Bundle savedInstanceState) {
         super.onCreate(savedInstanceState);
         setContentView(R.layout.main);

         mSensorManager = (SensorManager) getSystemService
(SENSOR_SERVICE);
     }
```

如何使用 SensorManager ?


android.content.Context
android.content.ContextWrapper
android.app.Service
<<interface>>
android.content.ComponentCallbacks
onLowMemory() : void
<<realize>>

Context.getSystemService

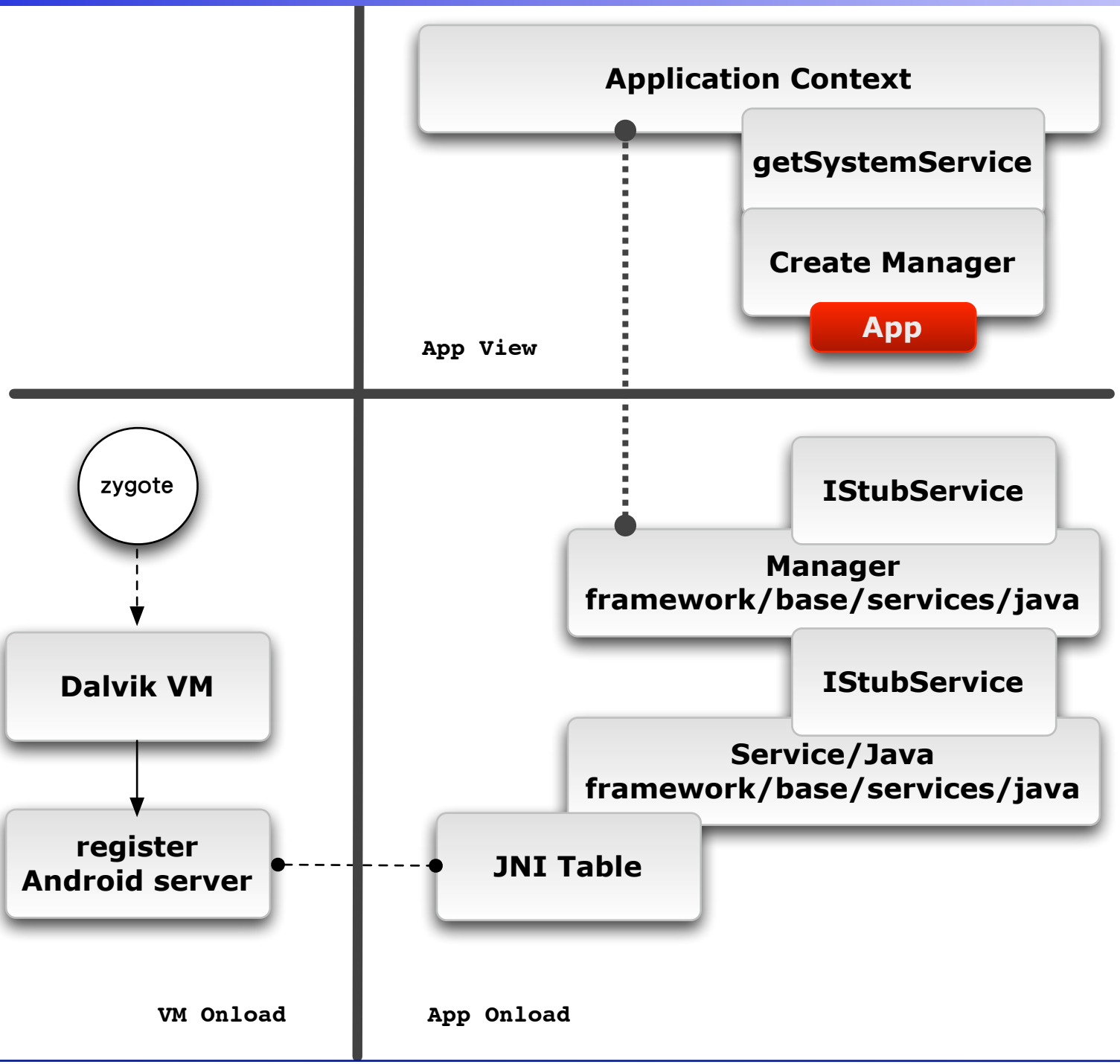
```
public Object getSystemService(String name) {
    if (WINDOW_SERVICE.equals(name)) {
        return WindowManagerImpl.getDefault();
    } else if (LAYOUT_INFLATER_SERVICE.equals(name)) {
        synchronized (mSync) {
            LayoutInflater inflater = mLayoutInflater;
            if (inflater != null) {
                return inflater;
            }
            mLayoutInflater = inflater =
                PolicyManager.makeNewLayoutInflater(getOuterContext());
            return inflater;
        }
    } else if (ACTIVITY_SERVICE.equals(name)) {
        return getActivityManager();
    } else if (ALARM_SERVICE.equals(name)) {
        return getAlarmManager();
    ..
    } else if ( SENSOR_SERVICE.equals(name)) {
        return getSensorManager();
    }
   ...
}
```



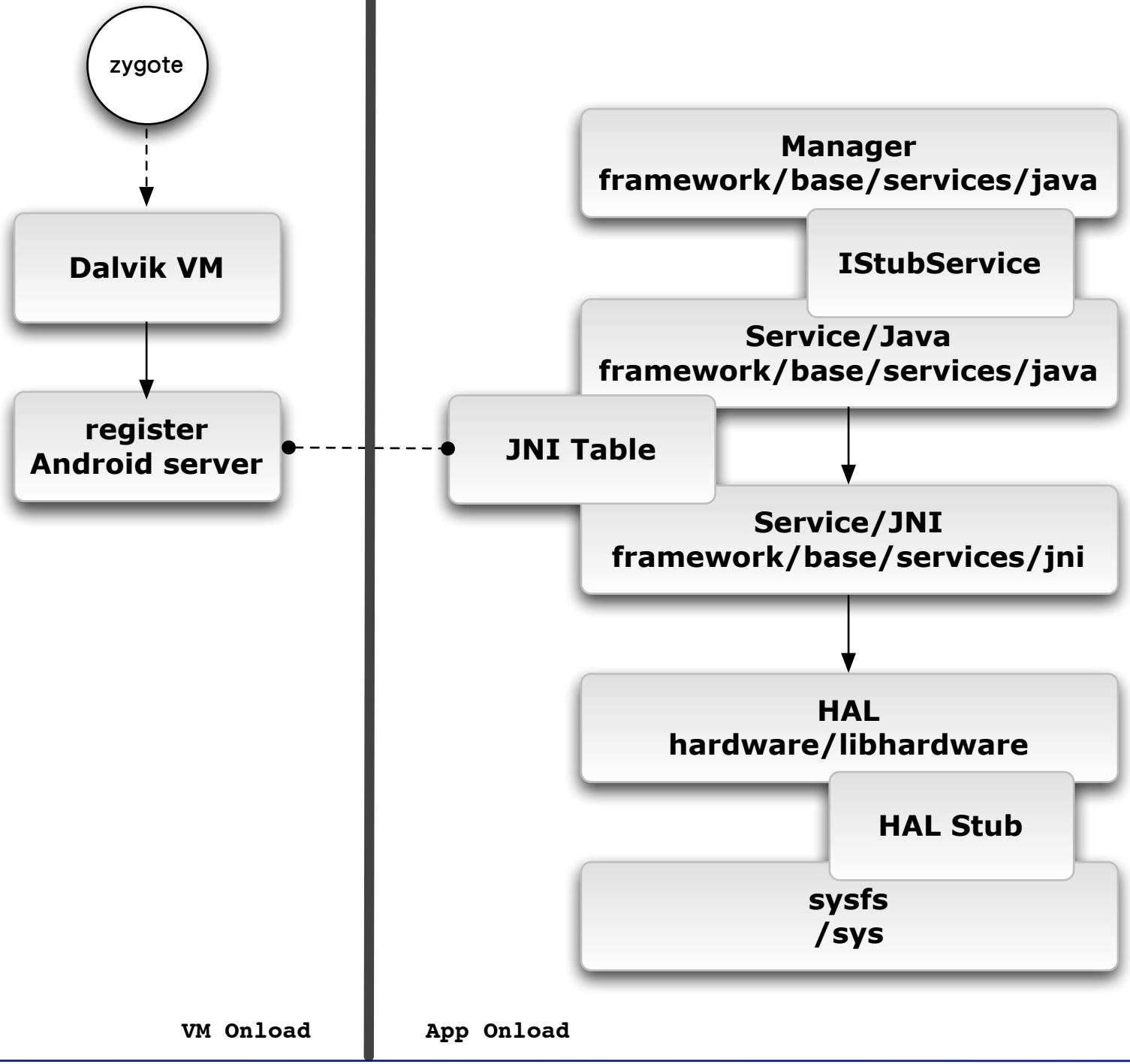
第二次比較快

```
    private SensorManager getSensorManager() {
        synchronized (mSync) {
            if (mSensorManager == null) {
                mSensorManager = new SensorManager
(mMainThread.getHandler().getLooper());
            }
        }
        return mSensorManager;
    }
```

第二次比較快



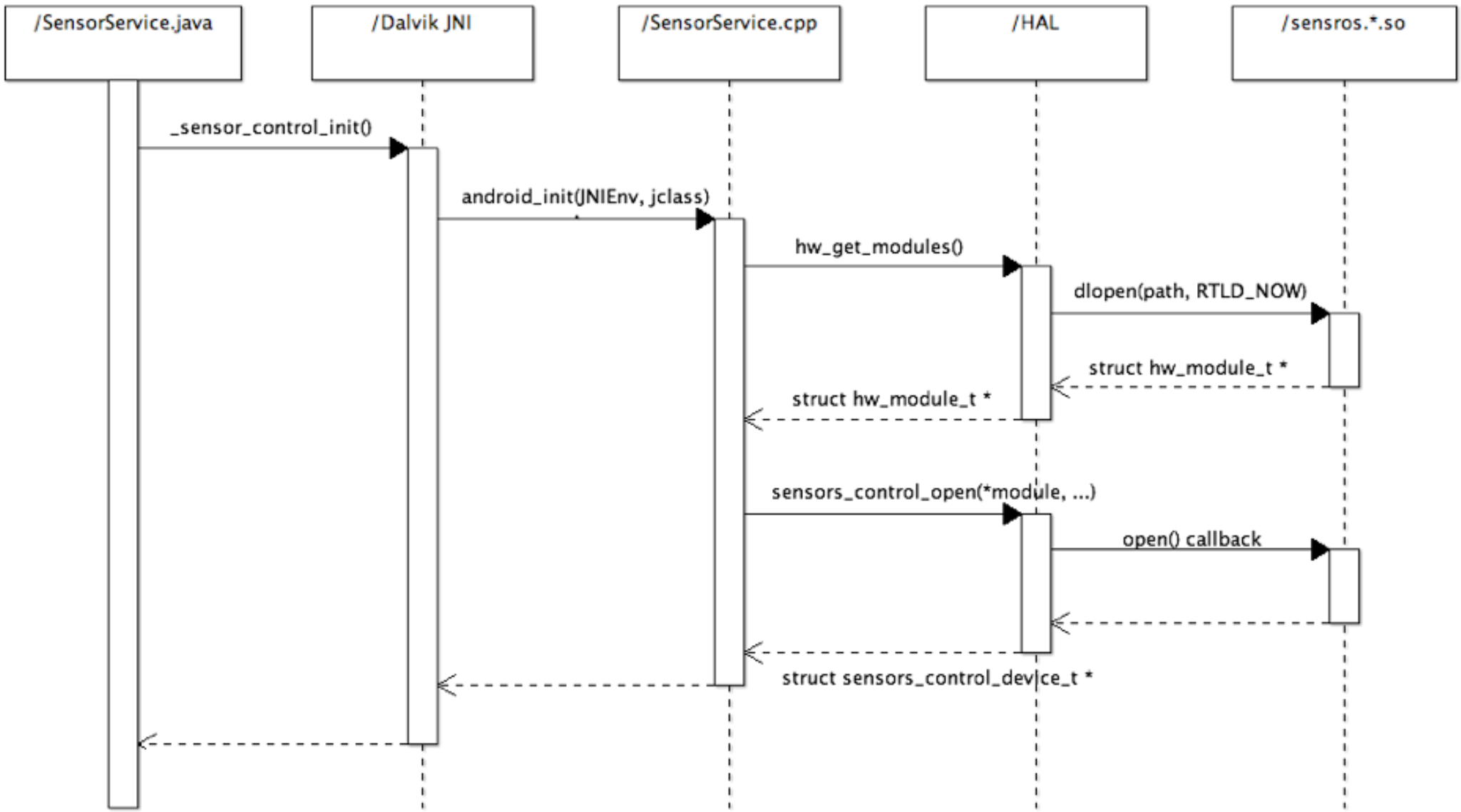
- device driver部份



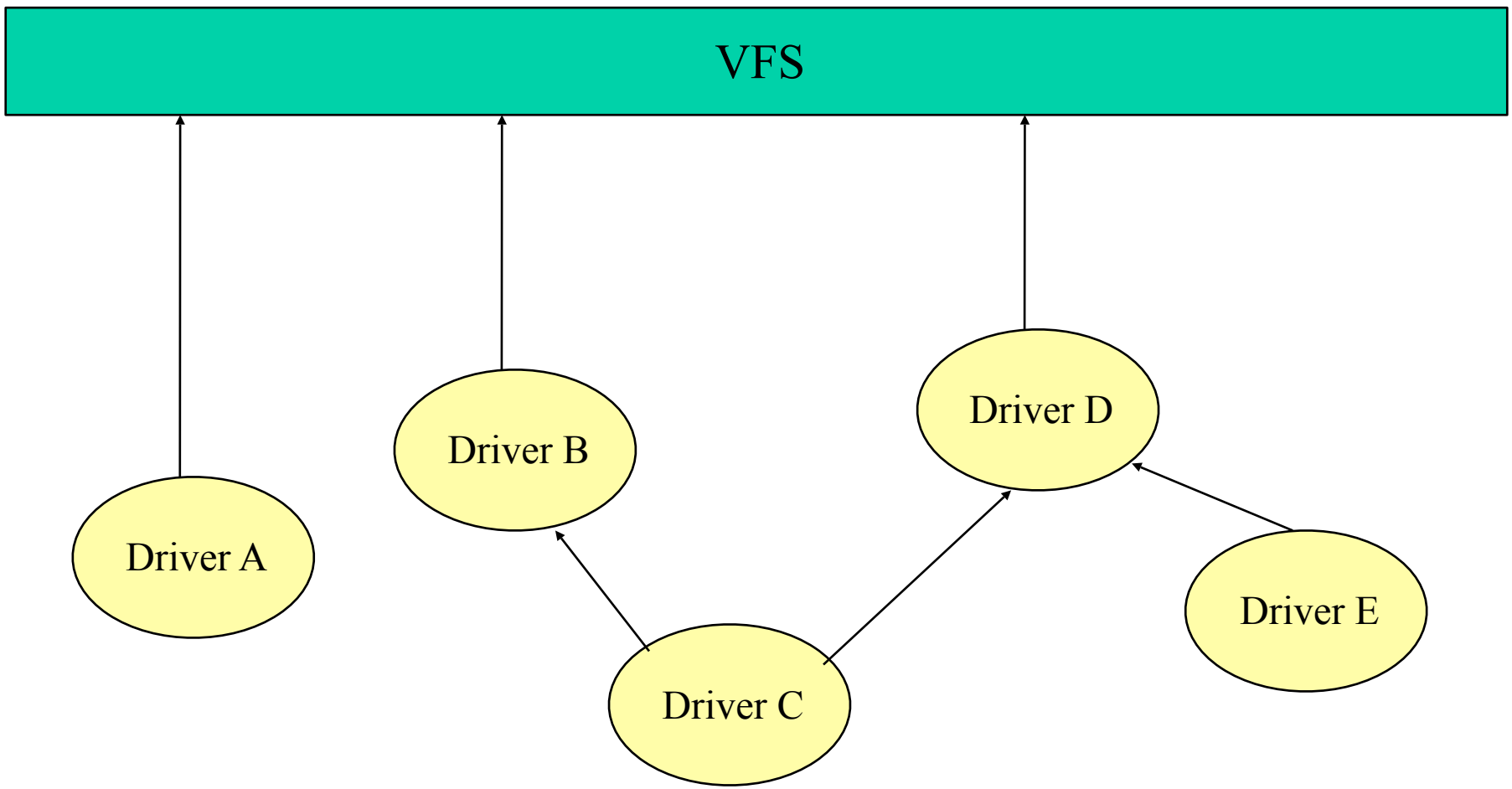
at framework/base/services/java/SensorService.java

```
    class SensorService extends ISensorService.Stub {
                                ...
     public SensorService(Context context) {
         if (localLOGV) Log.d(TAG, "SensorService
startup");
         _sensors_control_init();
         mNotificationManager = (NotificationManager)
context.getSystemService(Context.NOTIFICATION_SERVICE);
         mContext = context;
     }
    ...
}
```

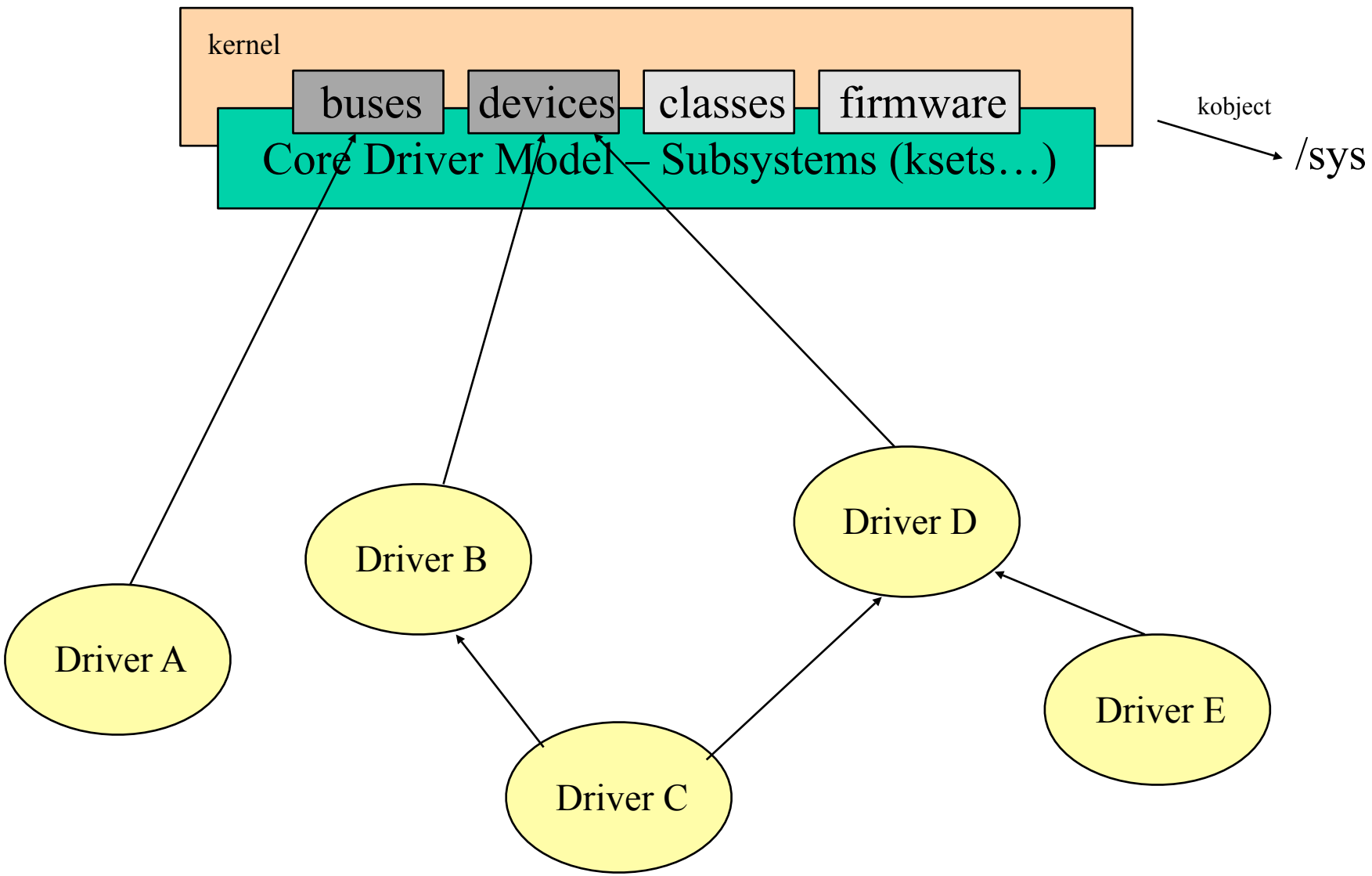
at framework/base/services/java/SensorService.java



◆2.4以VFS為中心: 實體集權模式



◆2.6以kobject為中心：虛擬集權模式

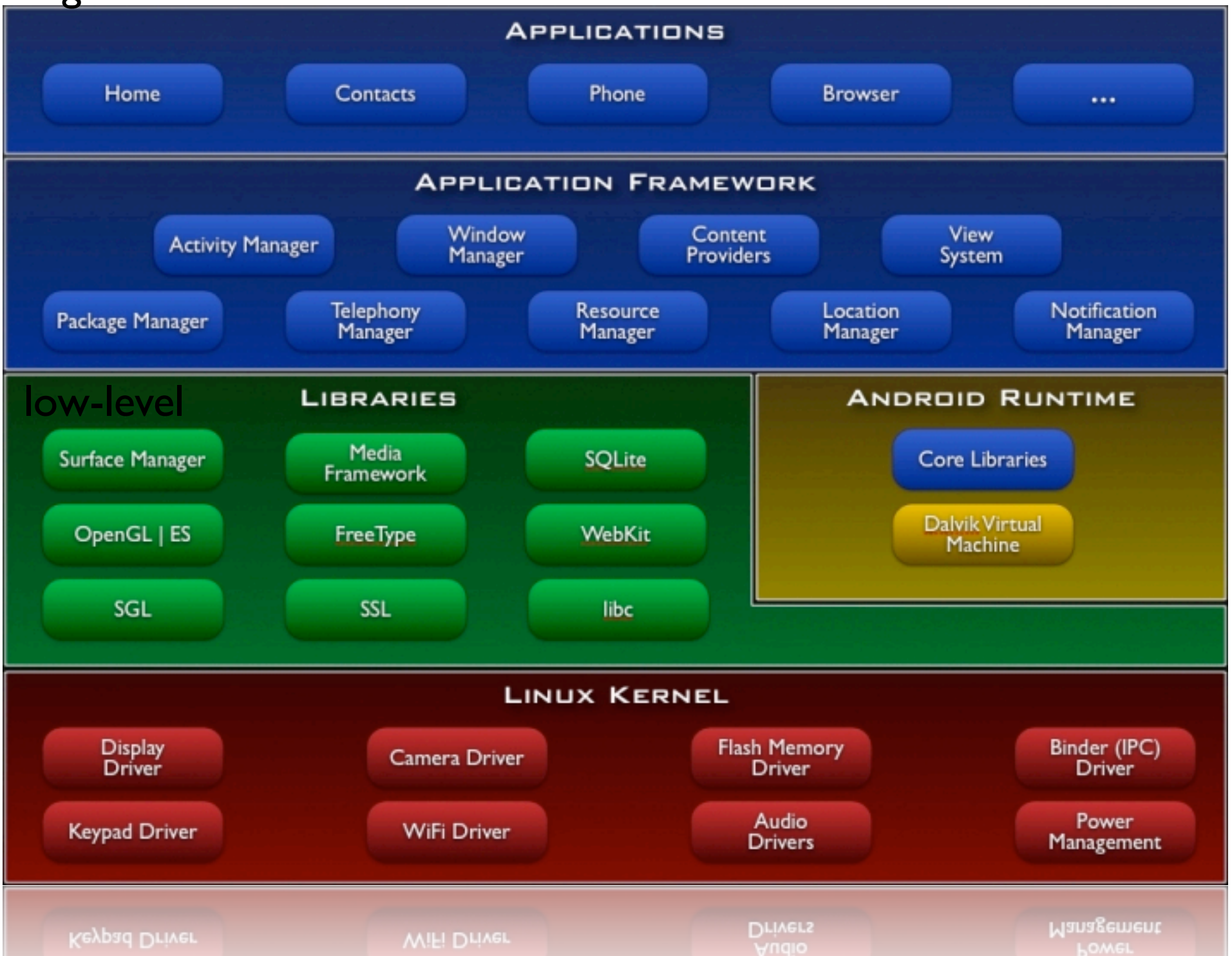


Android 效能最佳化
幾個策略舉例

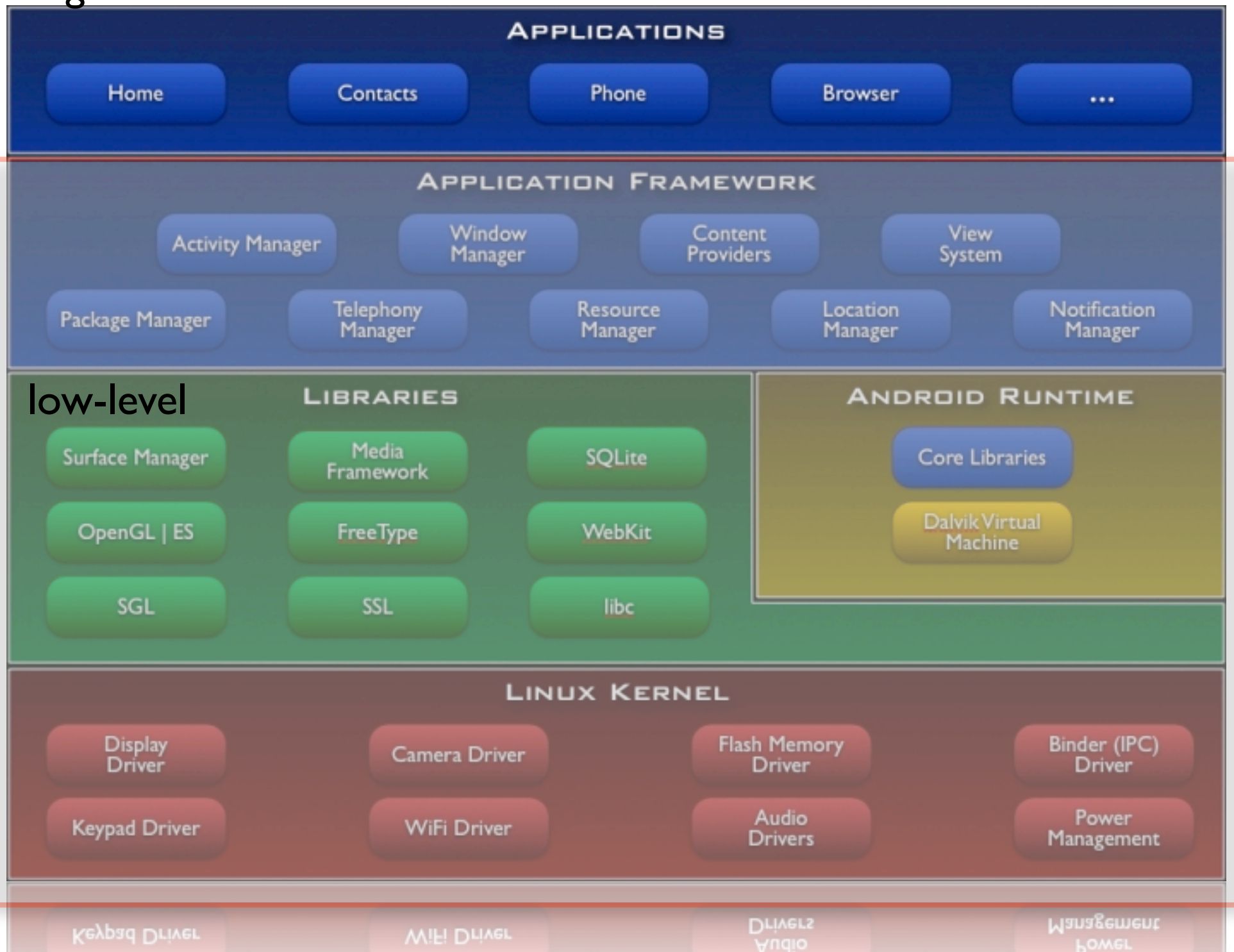
Android 效能最佳化 幾個策略舉例

- Launcher Model 的改進
- 啟動時getSystemService
- Framework本身設計的改善
- HAL module善用sysfs

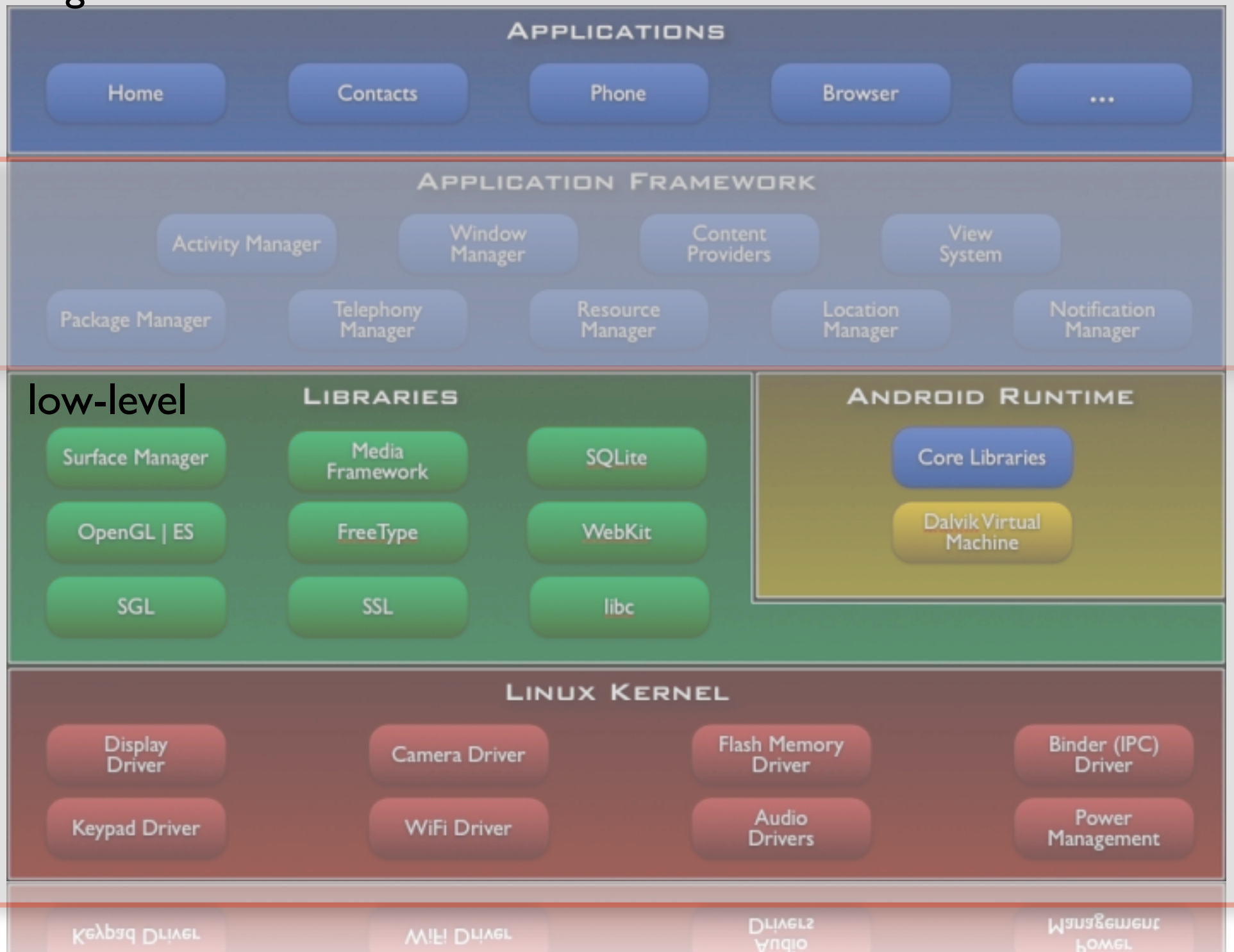
high-level


Applications
Home
Contacts
Phone
Browser
...
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Package Manager
Telephony Manager
Resource Manager
Location Manager
Notification Manager
low-level
Libraries
Surface Manager
Media Framework
SQLite
OpenGL | ES
FreeType
WebKit
SGL
SSL
libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Flash Memory Driver
Binder (IPC) Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management

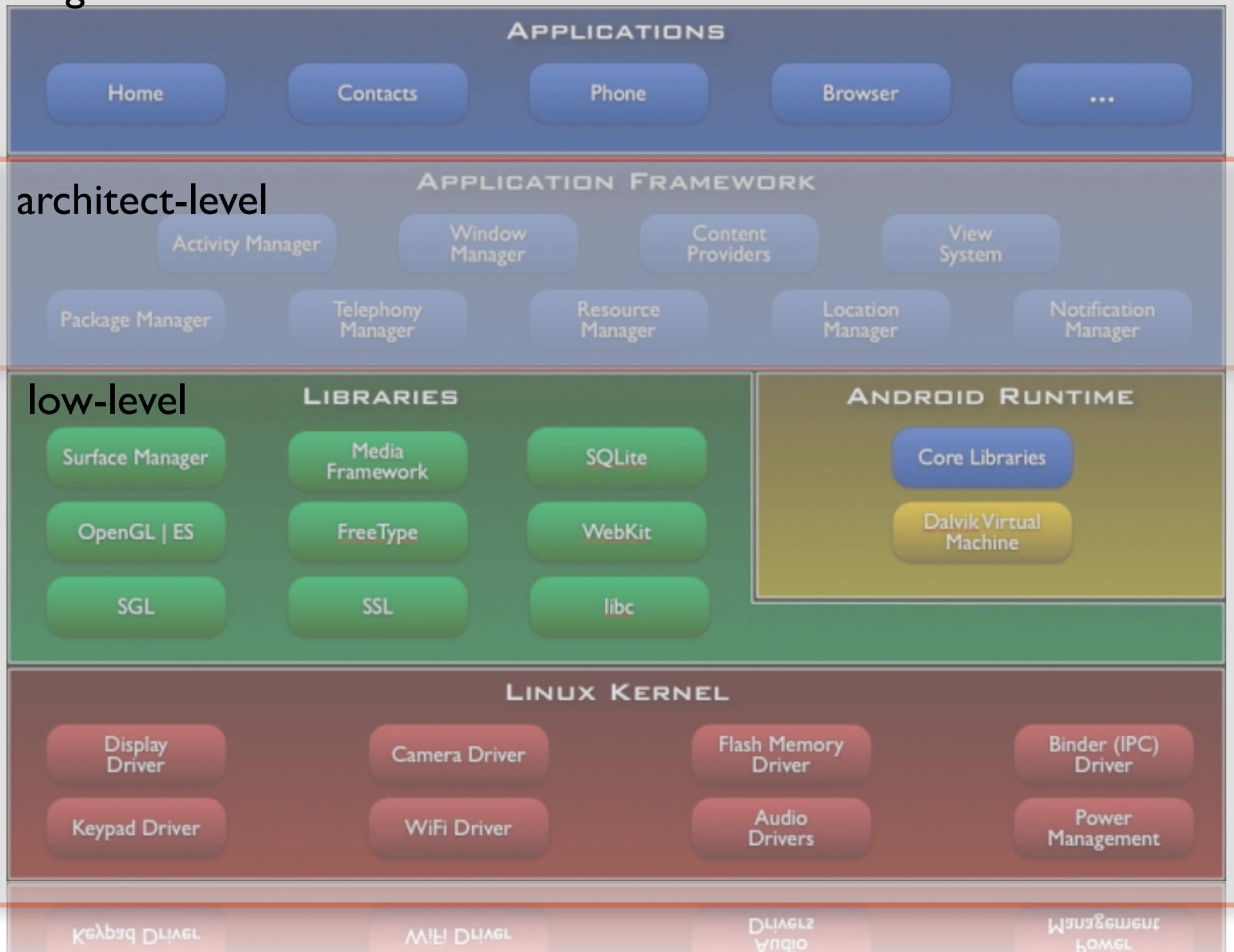
high-level


Applications
Home
Contacts
Phone
Browser
...
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Package Manager
Telephony Manager
Resource Manager
Location Manager
Notification Manager
low-level
Libraries
Surface Manager
Media Framework
SQLite
OpenGL | ES
FreeType
WebKit
SGL
SSL
libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Flash Memory Driver
Binder (IPC) Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management

high-level


Applications
Home
Contacts
Phone
Browser
...
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Package Manager
Telephony Manager
Resource Manager
Location Manager
Notification Manager
low-level
Libraries
Surface Manager
Media Framework
SQLite
OpenGL | ES
FreeType
WebKit
SGL
SSL
libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Flash Memory Driver
Binder (IPC) Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management

high-level



architect-level

low-level

做驅動程式（底層）的人對框架（architect）不能置身事外

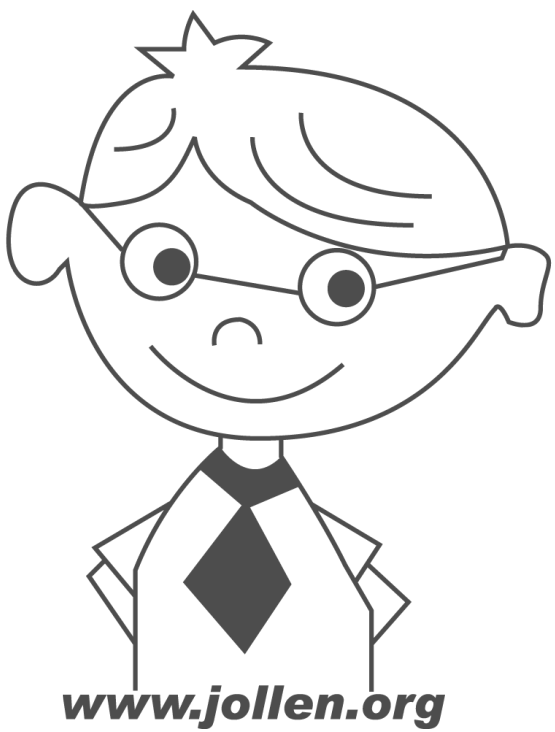
Android Source Tree

Android Source Tree

```
|-- Makefile          (全局的Makefile)
|-- bionic            (Bionic含义为仿生，这里面是一些基础的库的源代码)
|-- bootloader        (引导加载器)
|-- build             (build目录中的内容不是目标所用的代码，而是编译和配置所需
要的脚本和工具)
|-- dalvik             (JAVA虚拟机)
|-- development       (程序开发所需要的模板和工具)
|-- external           (目标机器使用的一些库)
|-- frameworks        (应用程序的框架层)
|-- hardware           (与硬件相关的库)
|-- kernel             (Linux2.6的源代码)
|-- packages           (Android的各种应用程序)
|-- prebuilt           (Android在各种平台下编译的预置脚本)
|-- recovery           (与目标的恢复功能相关)
`-- system            (Android的底层的一些库)
```

New Product File Tree

<company_name>
    <board_name>
        + Android.mk
        + product_config.mk
        + system.prop
    products
        + AndroidProducts.mk
        + <first_product_name>.mk
        + <second_product_name>.mk



1. Get Google Android

$ repo init -u git://android.git.kernel.org/platform/manifest.git -b cupcake
$ repo sync

2. Manifest file.

<manifest>
...
<project name="platform/vendor/asus/eee_701" \ path="vendor/asus/eee_701"/>
...
</manifest>

3. Get EeePC platform.

$ repo sync

4. Build Android image.

$ TARGET_ARCH=x86 TARGET_PRODUCT=eee_701 DISABLE_DEXPREOPT=true make -j2 installer_img

5. Create USB boot stick.

Use make-live script. https://review.source.android.com/Gerrit#change,6475

Build EeePC 701 Product Tips # Google API issue

```
$(call inherit-product, $(SRC_TARGET_DIR)/product/generic.mk)

PRODUCT_NAME := eee_701
PRODUCT_DEVICE := eee_701
PRODUCT_POLICY := android.policy_mid
PRODUCT_PROPERTY_OVERRIDES += \
        ro.com.android.dataroaming=true
```

☑ Cupcake将Google APIs放到add-ons

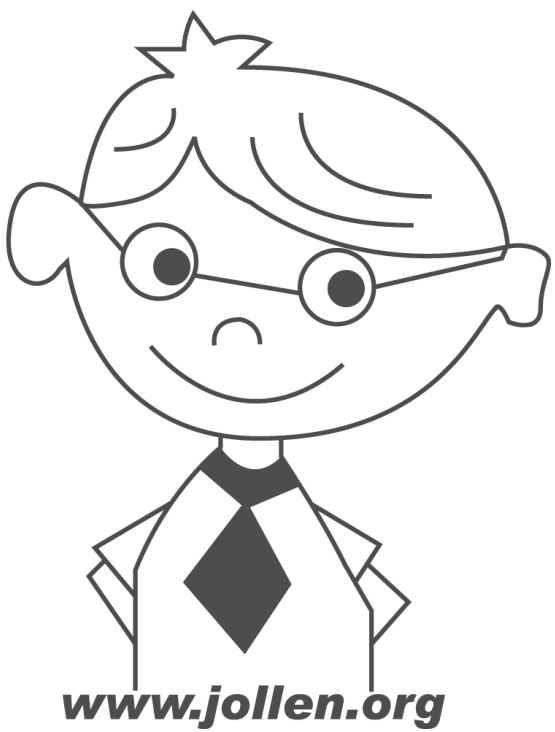
Build EeePC 701 Product Tips # e2fsprogs issue

external/e2fsprogs/Android.mk:

```
--- a/Android.mk
+++ b/Android.mk
@@ -1,3 +1,3 @@
ifneq ($(TARGET_SIMULATOR),true)
-# include $(call all-subdir-makefiles)
+ include $(call all-subdir-makefiles)
endif
```

Android Image Files

- boot.img
- installer.img
- ramdisk.img
- system.img
- userdata.img



建立Android开机随身碟

```
$ make-live
```

android
open source project

Change 6475: Adding a preliminary script that converts an USB installer.img into a "live" usb stick that can be b

▼ **Description**

Owner	Chris Elford
Project	platform/vendor/asus/eee_701
Branch	master
Uploaded	Dec 30, 2008 11:24 AM
Updated	Jan 13, 2009 7:21 AM
Status	Review in Progress

Permalink

Adding a preliminary script that converts an USB installer.img into a "live" usb stick that can be booted directly

▶ **Dependencies**

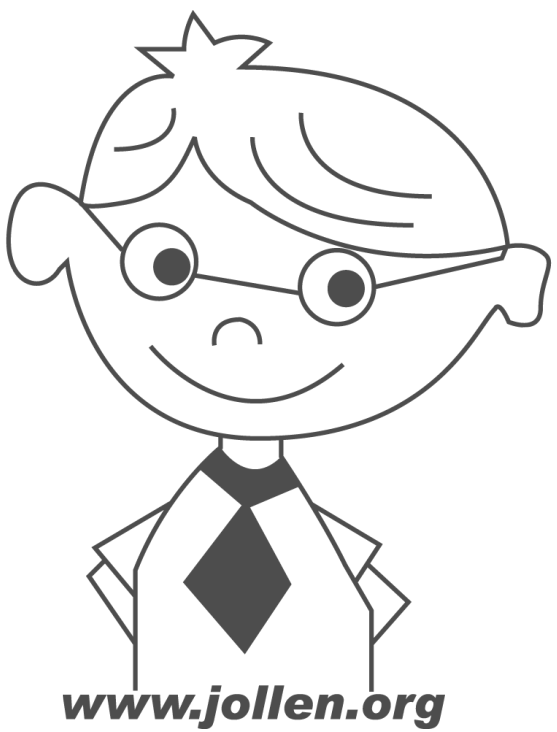
▼ **Approvals**

Reviewer	*Verified*	*Code Review*
Alexey Tarasov		

Need Verified
Need Code Review

使用 VirtualBox

- ☑ AMD PCnet32 PCI support
- ☑ VESA VGA graphics support
- ☑ VGA 8x8 font
- ☑ VGA 8x16 font

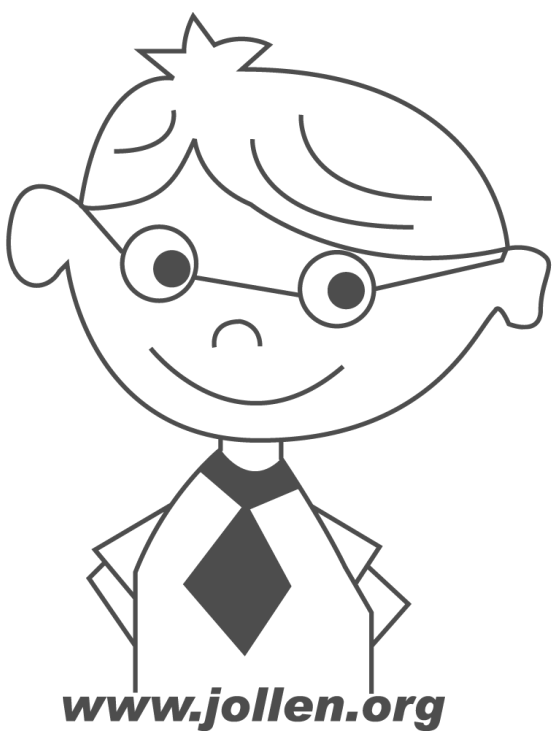


转换installer.img为VDI格式

```
$ VBoxManage convertromraw -format VDI ./installer.img ./android.vdi
```

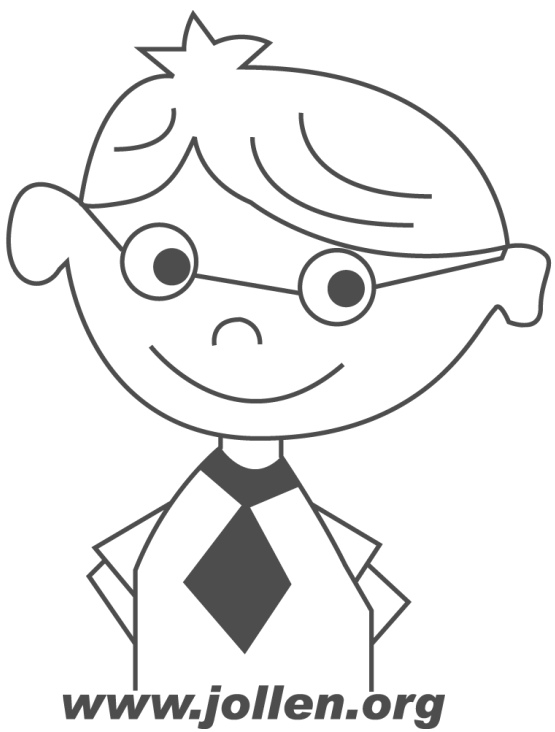
Android Init Process

- ☑ device/system/init
- ☑ device/system/init/init.c
- ☑ /etc/init.rc
- ☑ 自动mount file system (不需要/etc/fstab)



Running Applications

- /system/bin/logd
- /sbin/adbd
- /system/bin/usbd
- /system/bin/debuggerd
- /system/bin/rild
- /system/bin/app_process
- /system/bin/runtime
- /system/bin/dbus-daemon
- system_server



- Application Developers
- Product Branch Maintainer
- **Hardware Developing**
  寫一個LED驅動程式: kernel, hal, framework, app
- Architecture Porting

www.jollen.org/consulting

Jollen@jollen.org