Gkil1.com

안드로이드 개발의 다른점

-다른 어플리케이션 개발과 차이점

정승일
지킬닷컴 대표
국가SW마에스트로 멘토

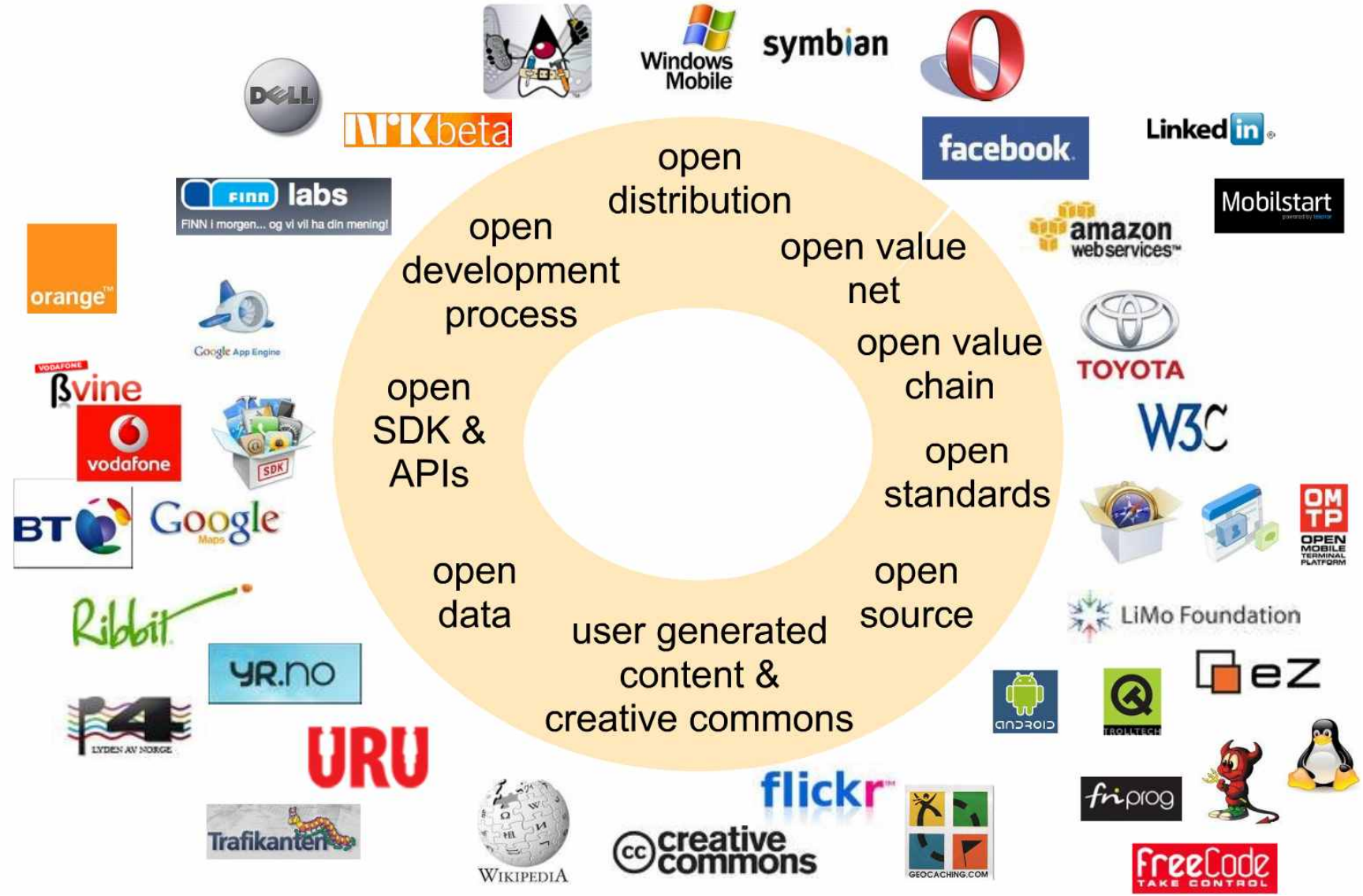
Google

"We want the next killer application to be written for cell phones"

(Andy Rubin, Google)

Open

Open에 대한 다양한 관점



형용사 – 19개

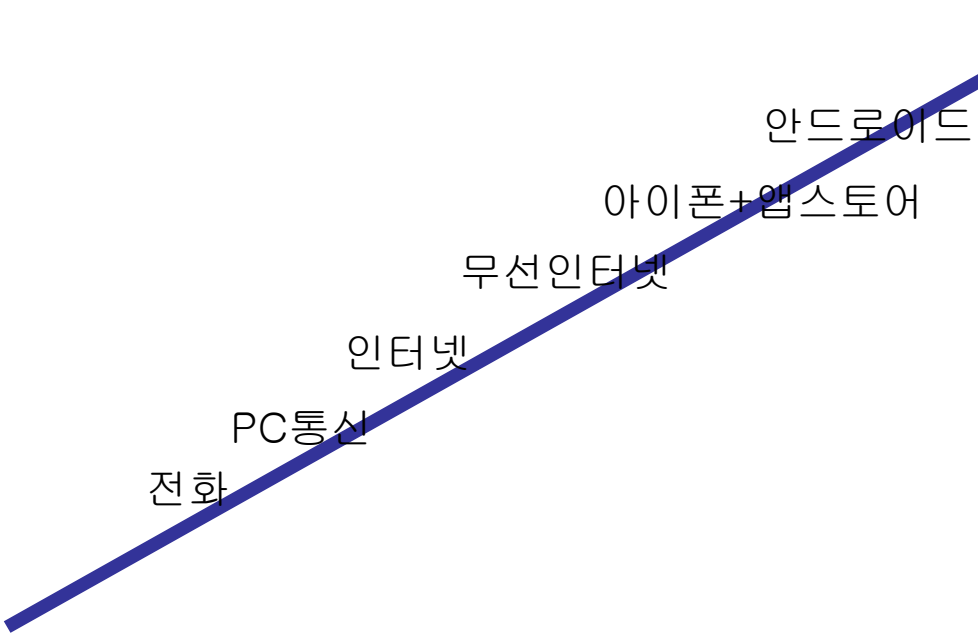
동사 – 14개

명사 -2 개

Open은 Open되어있다.

Open은 상대적인 개념으로 진화하고, 역사는 반복한다.

오픈은 진화의 한 단계이고, 역사적 큰줄기임

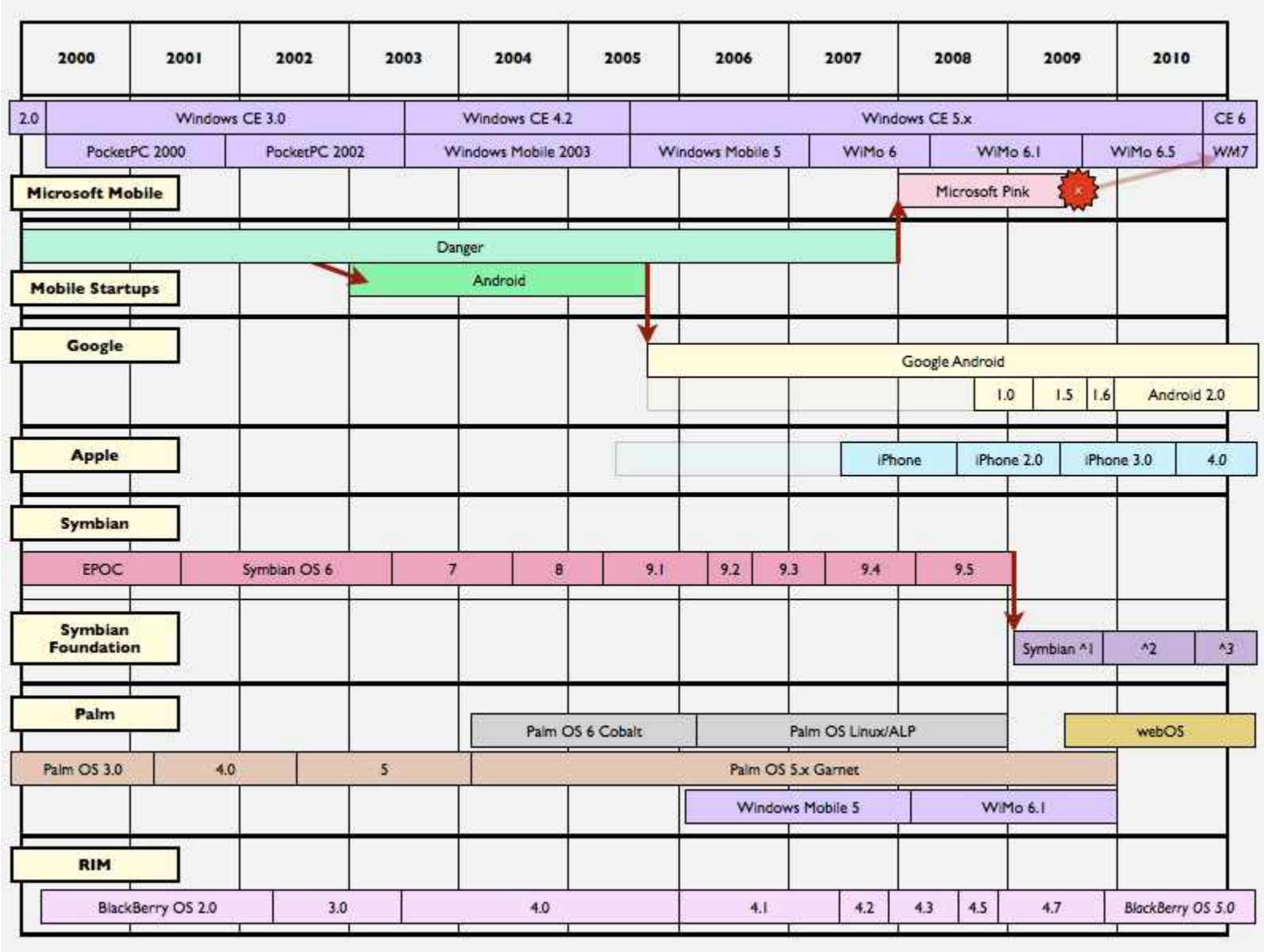


안드로이드적인 Open은....

Interconnected components

All apps are equal!

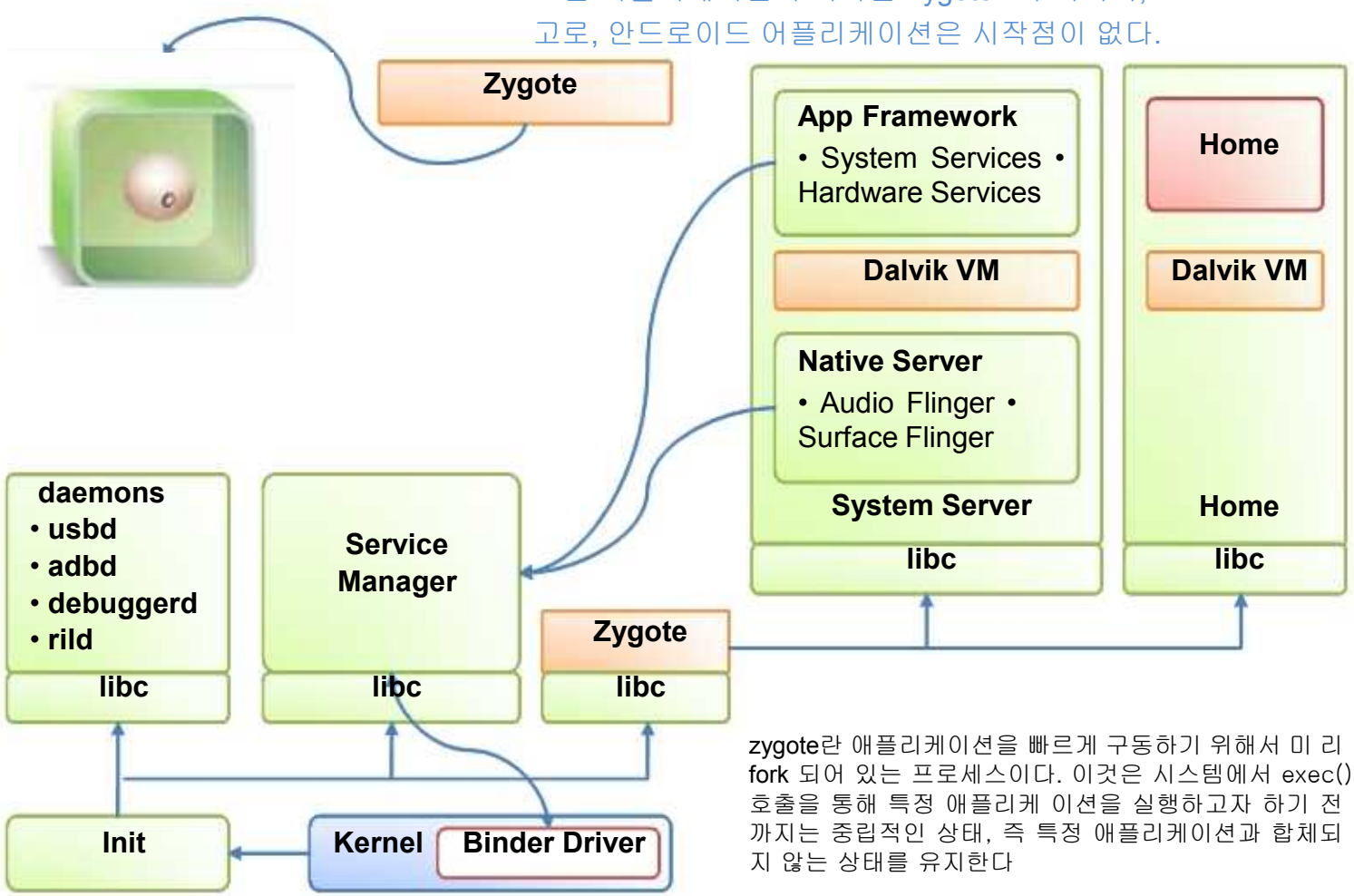
They didn't emphasize only Open Source.....



http://www.appleinsider.com/articles/09/11/21/inside_googles_android_and_apples_iphone_os_as_software_markets.html

Activity Internal : Android Start-up

안드로이드에서 **Zygote**의 존재는 개방을 고려해서 만들어졌다.
모든 어플리케이션의 시작은 Zygote로부터이며,
고로, 안드로이드 어플리케이션은 시작점이 없다.



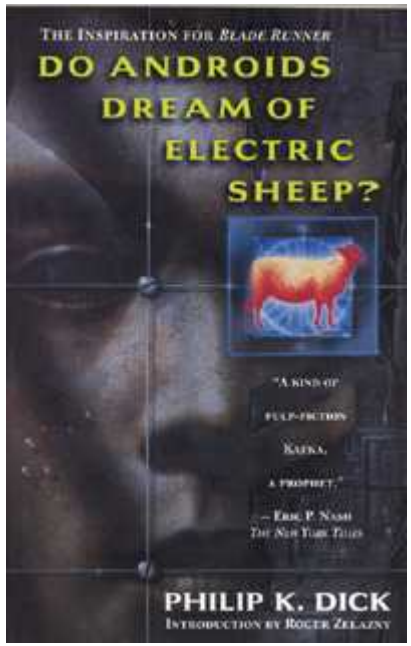
zygote란 애플리케이션을 빠르게 구동하기 위해서 미 리 fork 되어 있는 프로세스이다. 이것은 시스템에서 exec() 호출을 통해 특정 애플리케 이션을 실행하고자 하기 전까지는 중립적인 상태, 즉 특정 애플리케이션과 합체되지 않는 상태를 유지한다

Zygote and Nexus 어원

Do Androids Dream of Electric Sheep? - **Philip K. Dick**

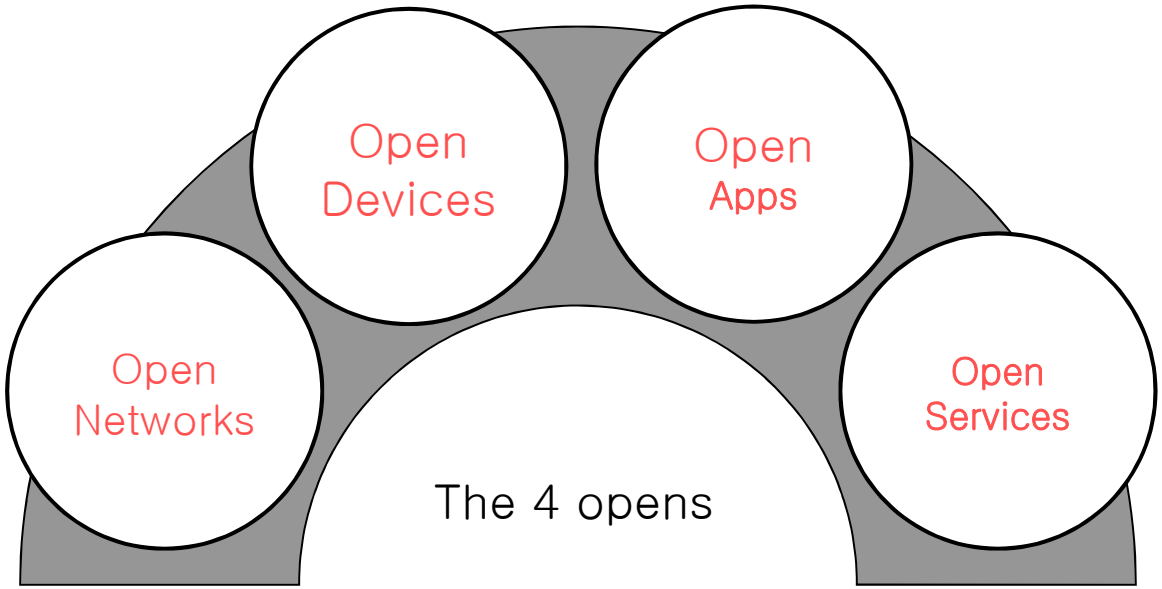
"블레이드러너"의 영감을 준 책
Zygote는 **Nexus** 유형(결과물)을 만들어내는 **Bath**(모태)이며,
Nexus1은 그 첫번째 유형

구글이 왜**?** 자신의 브랜드 첫 작품을 **nexus one**이라고 했는지 짐작이감**..**

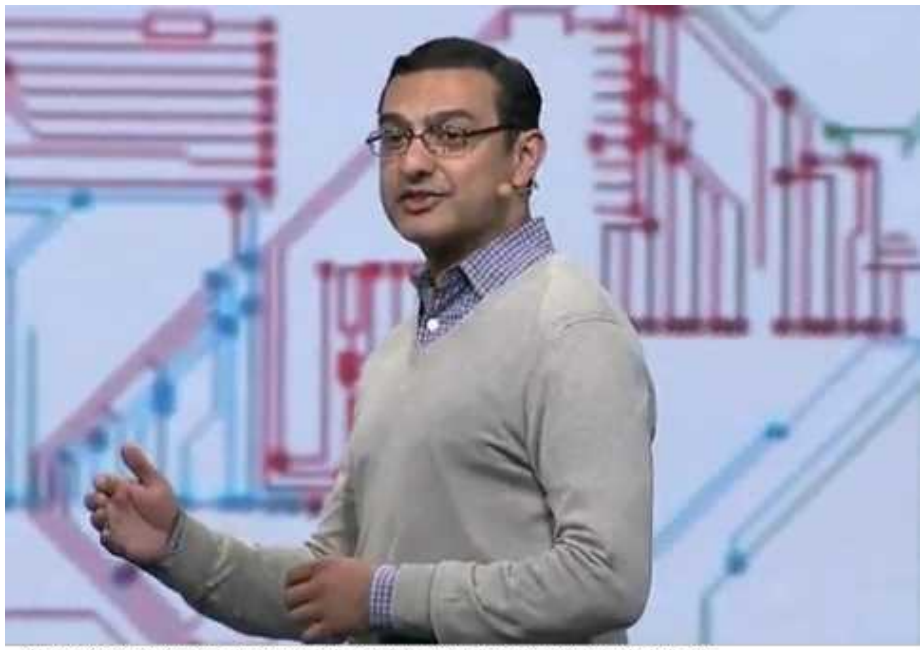


구글의 Open Ecosystem

The FOUR OPENS of Successful Open Access. The letter of GOOGLE to FCC, July 18, 2007



Android and Open Ecosystem



◇기조연설 중인 빅 군도트라 구글 엔지니어담당 부사장[사진=구글]

"If Google did not act, we faced a Draconian future. **A future where one man, one company, one device, one carrier would be our only choice. That's a future we don't want.**
So if you believe in openness..if you believe in choice..if you believe in innovation.. from everyone, then welcome to Android. Now, let's get started."

-Vic Gundotra quoting Andy Rubin (both are VPs of Engineering at Google)

Dalvik 가상 머신 (Virtual Machine)- 모바일환경을 위한

다음과 같은 환경을 고려한 Bytecode Interpreter

- Slow CPU (250-500 MHz)
- RAM Usage : Low-level : 20M, High-level : 24M (system library : 10M)
- Little RAM (64MB) : Available RAM : 20M
- Bus speed : 100MHz
- Data Cache : 16~32K
- No swap space, Battery power

Dalvik이란 이름은**, Dalvik**의 창시자인 본스타인이 자신의 조상이 살던 아이슬란드의 한 어촌의 이름을 따서 만든 것임.

Stats

- 30% fewer instructions and code units
- 35% more bytes in the instruction stream

Dalvik 가상 머신 (Virtual Machine) is not Java VM

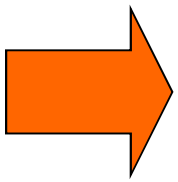
- 모바일환경의 최고 성능을 위해 고안됨

```
public class Demo {
    private static final char[] DATA =
        { 'N','o','s','e','r',
        'k','n','o','w','s',
        'A','n','d','r','o','i','d'
    };
}
```

Java VM(JVM) bytecode

```
 0: bipush 17
 2: newarray char
 4: dup
 5: iconst_0
 6: bipush 78
 8: castore

94: dup
95: bipush 16
97: bipush 100
99: castore
100: putstatic DATA
103: return
```

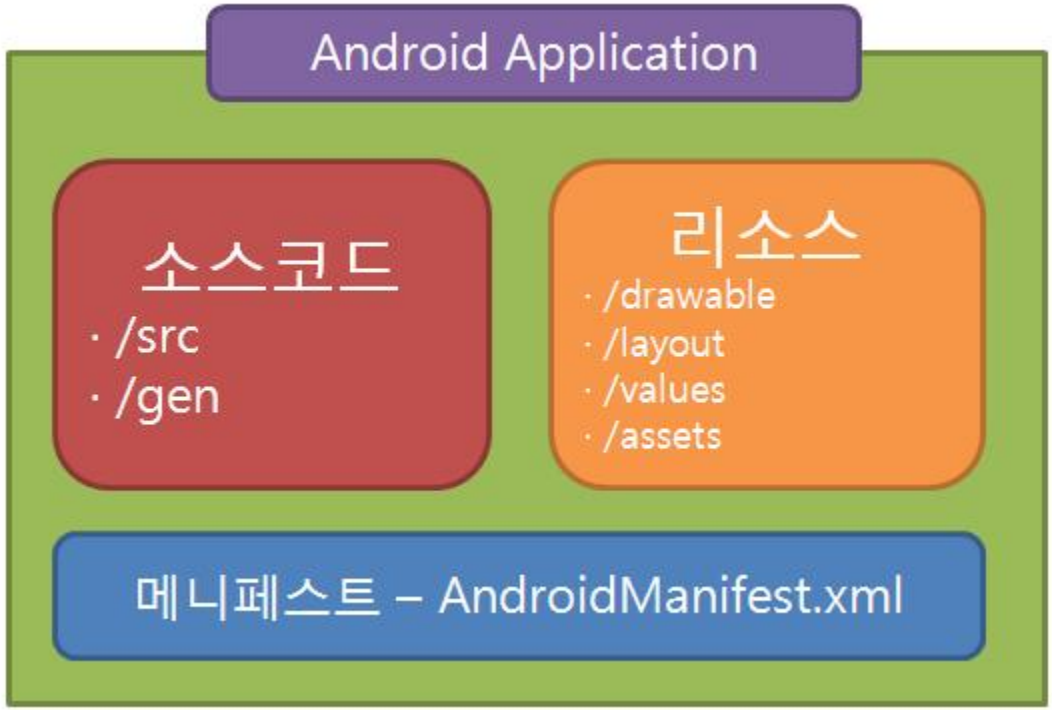


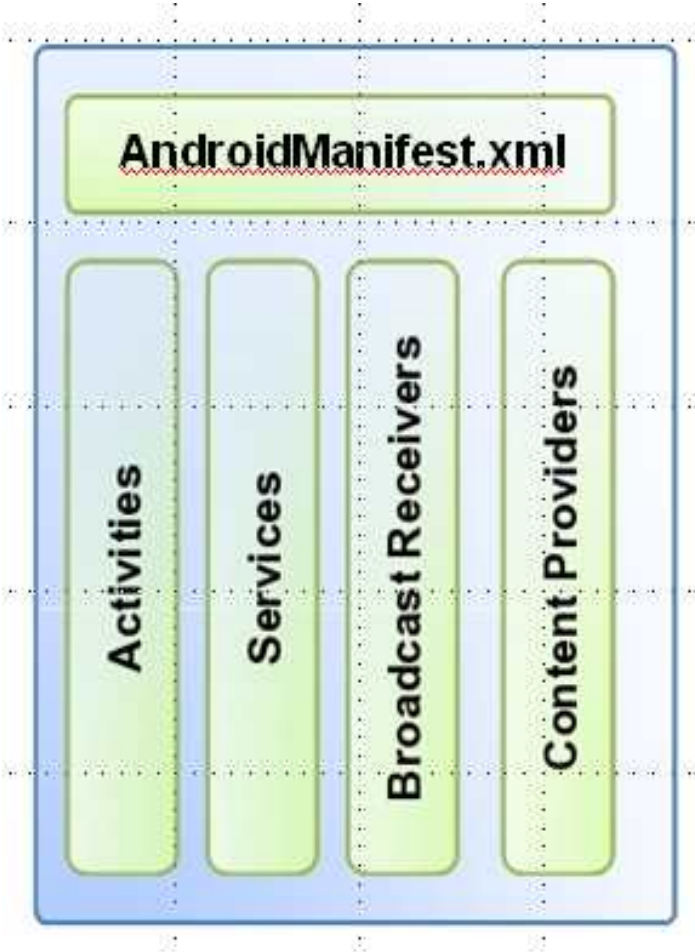
35% fewer Instruction

Dalvik VM(DVM) bytecode

```
0000: const/16 v0, #int 17
0002: new-array v0, v0, [C
0004: fill-array-data v0, 0a
0007: sput-object v0, DATA:[C
0009: return-void
000a: array-data (21 units)
        'N', 'o', 's', 'e', 'r' ...
```

Android Application Structure





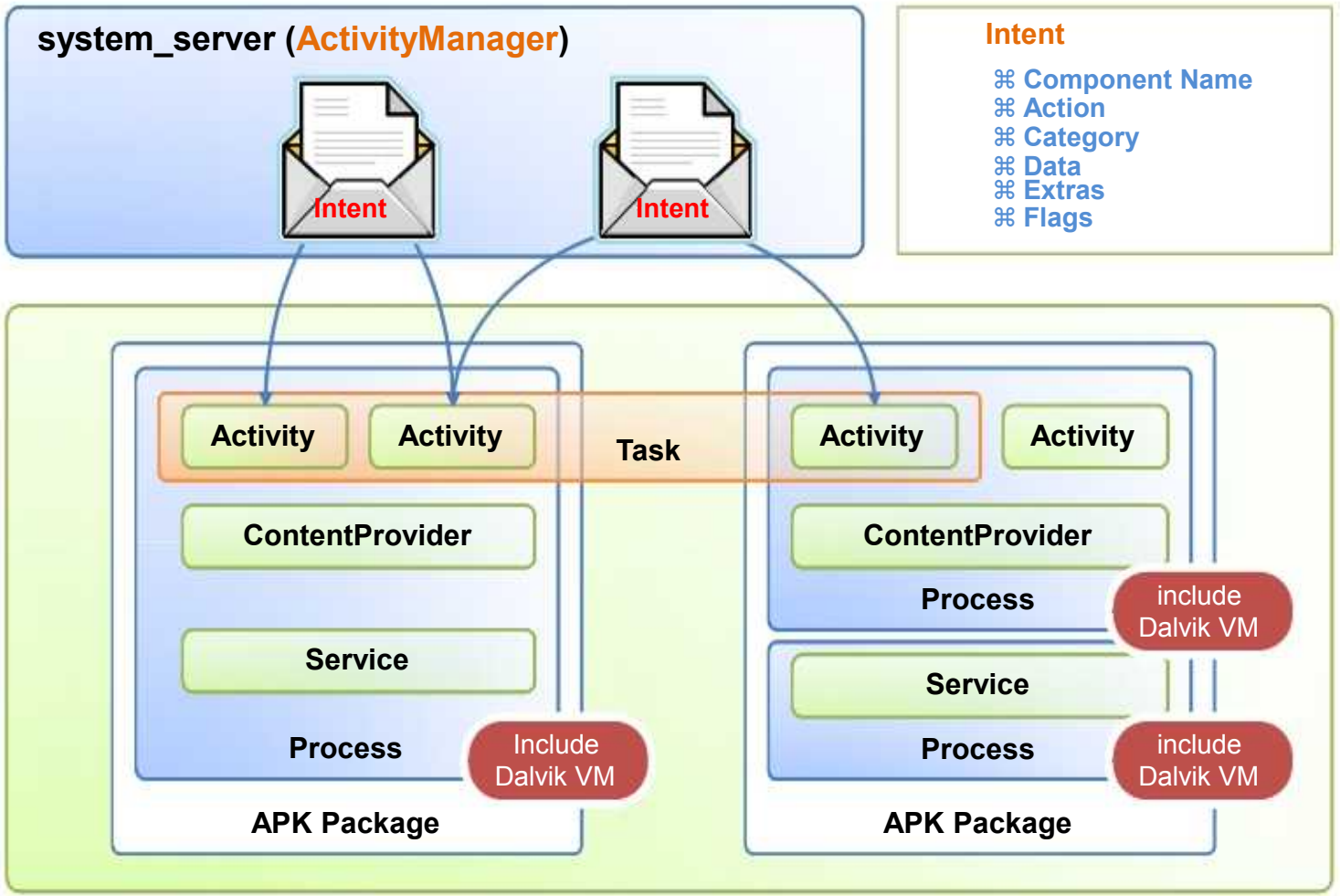
안드로이드 Apps 분자구조

- **Activity** UI for one focused endeavor
- **Service** no UI, used for long running processes
- **Broadcast receiver** receives announcements
- **Content provider** makes data available

Intents

• “an Intent is a simple message object that represents an ‘intention’ to do something”
• “an intent is an abstract description of an operation to be performed”

Overview


system_server (ActivityManager)
Intent
Intent
Intent
⌘ Component Name
⌘ Action
⌘ Category
⌘ Data
⌘ Extras
⌘ Flags
Activity
Activity
Task
Activity
Activity
ContentProvider
ContentProvider
Process
include
Dalvik VM
Service
Service
Process
Include
Dalvik VM
Process
include
Dalvik VM
APK Package
APK Package

Intent

Web browser

Intents:

VIEW + www.google.com

Android

Intents:

VIEW + www.google.com

VIEW + Contact

VIEW + Image

DIAL + 123

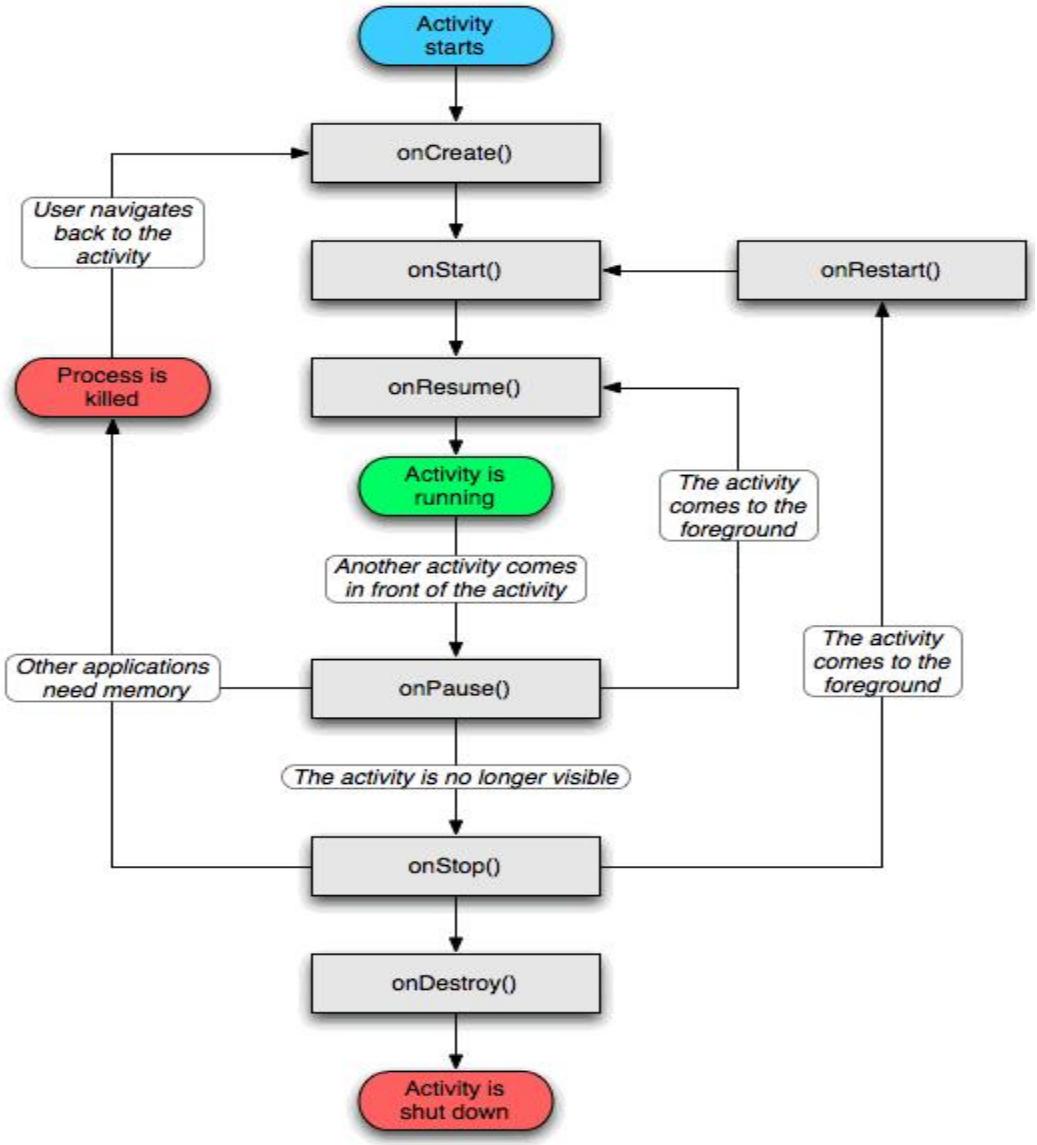
PICK + Image

EDIT + Contact

SEND + Note

.....

Intent = Action + Data (+ Extras + Categories + Flags)



Android의 Activity LifeCycle은

Activity간 Activation과의 관계

안드로이드는 다른 Application의 Activity 에 대한 접근을 고려하여 만든 플랫폼으로,

메모리의 가용성에 대한 고민이 Activity Lifecycle이 중요해짐.

startActivityForResult()에 대한 고찰

- 어플리케이션간 이동이 가능하고, Task중심의 개발이 가능하도록해줌
- 호출된 Activity가 종료되면, 호출한 측의 onActivityResult 핸들러로..

```
public class MyActivity extends Activity { static
    final int PICK_CONTACT_REQUEST = 0;

    protected boolean onKeyDown(int keyCode, KeyEvent event)
    { if (keyCode == KeyEvent.KEYCODE_DPAD_CENTER)
    { startActivityForResult(
        new Intent(Intent.ACTION_PICK, new Uri("content://contacts")),
        PICK_CONTACT_REQUEST);

        return true;
      }
      return false;
    }

    protected void onActivityResult(int requestCode, int resultCode, Intent data) {

      if (requestCode == PICK_CONTACT_REQUEST)
        { if (resultCode == RESULT_OK) {
            startActivity(new Intent(Intent.ACTION_VIEW, data));
        }
      }
    }
}
```

Main 함수에 대하여 – C,C++

C

```
// HelloWorldApp.c : create source file
// gcc -o HelloWorldApp HelloWorldApp.c : compile c source
// HelloWorldApp : run c binary

#include <stdio.h>

int main()
{
        printf("hello,  world\n");
        return 0;
}
```

C++

```
// HelloWorldApp.cpp : create source file
// g++ -o HelloWorldApp HelloWorldApp.cpp : compile c++ source
// HelloWorldApp : run c++ binary

#include <iostream>
using namespace std;
int main()
{
        cout << "Hello World!" << endl;
        return 0;
}
```

메인함수에 대하여 - Java와 Java Applet

Java

```
// HelloWorldApp.java : create source file // javac
HelloWorldApp.java : compile java source // java
HelloWorldApp : run java class

class HelloWorldApp {
    public static void main(String[] args)
    { System.out.println("Hello World!"); }

}
```

Java Applet

```
// HelloWorldApp.java : create source file // javac
HelloWorldApp.java : compile java source
// <applet code="HelloWorldApp.class" width=400 height=300 ignore=""> </applet>
import java.applet.*;
import java.awt.*;
public class HelloWorldApp extends Applet
{
    public void init() {      }
    public void stop() {      }
    public void paint(Graphics g)
    {
        g.drawString("Hello World",20,40);
    }
}
```

메인함수에 대하여 – Android는 Main 함수가 없다!??

Android Activity

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

왜 안드로이드에 Main()이 없는 것일까?

- main()은 시작점이다.
  -안드로이드는 시작점이 없다.
  -어플리케이션의 어떠한 위치에서라도 시작이 가능
- 어플리케이션보다 Task의 개념이 중요하다.
- 어플리케이션은 복수의 Activity로 구성될수 있고,
  각 Activity는 어플리케이션 독립적인 수행이 가능하다.
- 각 Activity는 자신의 개방성을 자신이 속한 어플리케이션의 Manifest.xml에서 intent-filter에 등록하면, 다른 어떤 위치에서라도, intent라는 메시지에 의해서 실행(Activation)이 가능하다. 이부분이 안드로이드 스타일의 핵심이다.

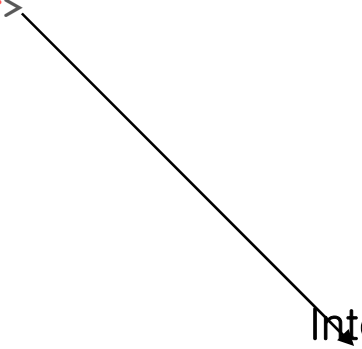
Manifest File

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
        xmlns:android="http://schemas.android.com/apk/res/android"
        package="com.gkill.helloandroid"
        android:versionCode="1"
        android:versionName="1.0.0">
  <application android:icon="@drawable/icon" android:label="@string/app_name">


    <activity android:name=".HelloAndroid" android:label="@string/app_name">


      <intent-filter>
      <action android:name="android.intent.action.MAIN" />
      <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>


    </activity>
  </application>
</manifest>
```



Intent-filter에 필터링 규칙을 등록하게 되면,
Activity에 접근이 가능하게 된다.

- 각 원은 하나의 어플리케이션이다.
- 원들이 서로를 공유하는 부분집합은 하나의 Activity이다.
- 전체의 어플리케이션끼리 개방되고, 융합되면서 Task를 이룬다.
- 기존의 프로그래밍 방식으로 안드로이드 어플리케이션을 만들면, 자칫 불필요한 코드의 과잉생산이 가능하다.

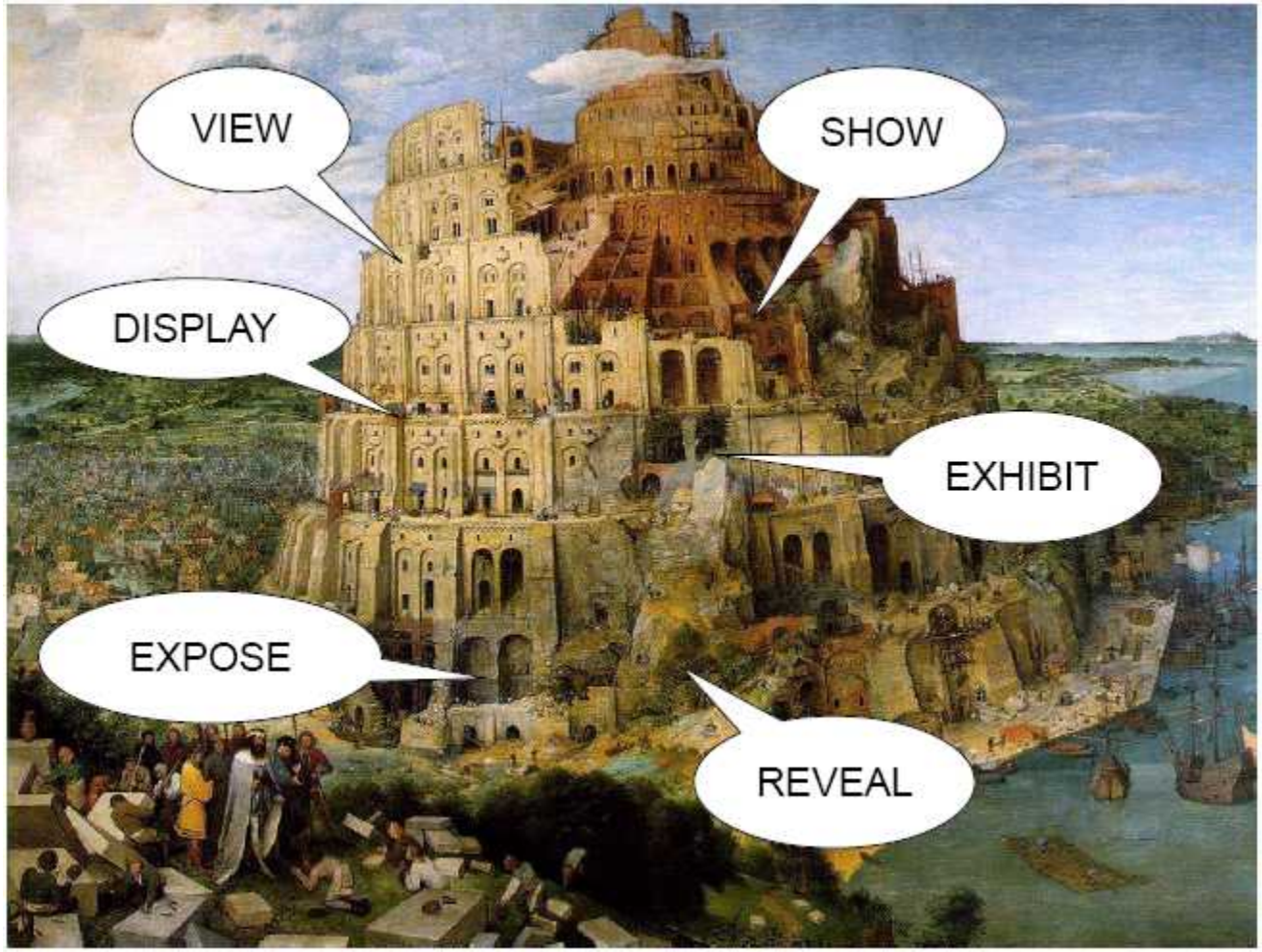
- 이해하라! 프로그래밍의 미래를
- 이해하라! 프로그래밍의 변화를

오픈이 적용된 어플리케이션, 서비스로 새로운 비즈니스 모델을!

- 어플리케이션과 어플리케이션의 소통방식은 Intents
- 누군가 내 어플리케이션을 Intents로 사용하게 하라!
- 많은 사람이 사용한다면, 비즈니스 기회는 만들수 있다.
- 미래에는 클라우드간에도 결국 융합될 것이다.

"안드로이드 어플리케이션 개발의 키워드"

"Openness"


VIEW
SHOW
DISPLAY
EXHIBIT
EXPOSE
REVEAL