

Présentation d'Android

98, avenue du Général Leclerc
92100 Boulogne-Billancourt
Tél : (+33)1 55 60 76 72
Fax : (+33)1 55 60 76 73



- **Introduction**
 - Android et les smartphones
- **Architecture technique**
 - Environnements et outils
- **Modèle de développement**
 - Cycle de vie
 - Le modèle de programmation par Intent
- **Exemple concret**
- **Quelques zooms sur ...**
 - La construction d'interface graphique
 - Invocation de services distants
- **Android, à quoi ça sert ?**
 - Grand public
 - S.I. d'entreprise
 - Contraintes et limitations
- **Questions / Réponses**



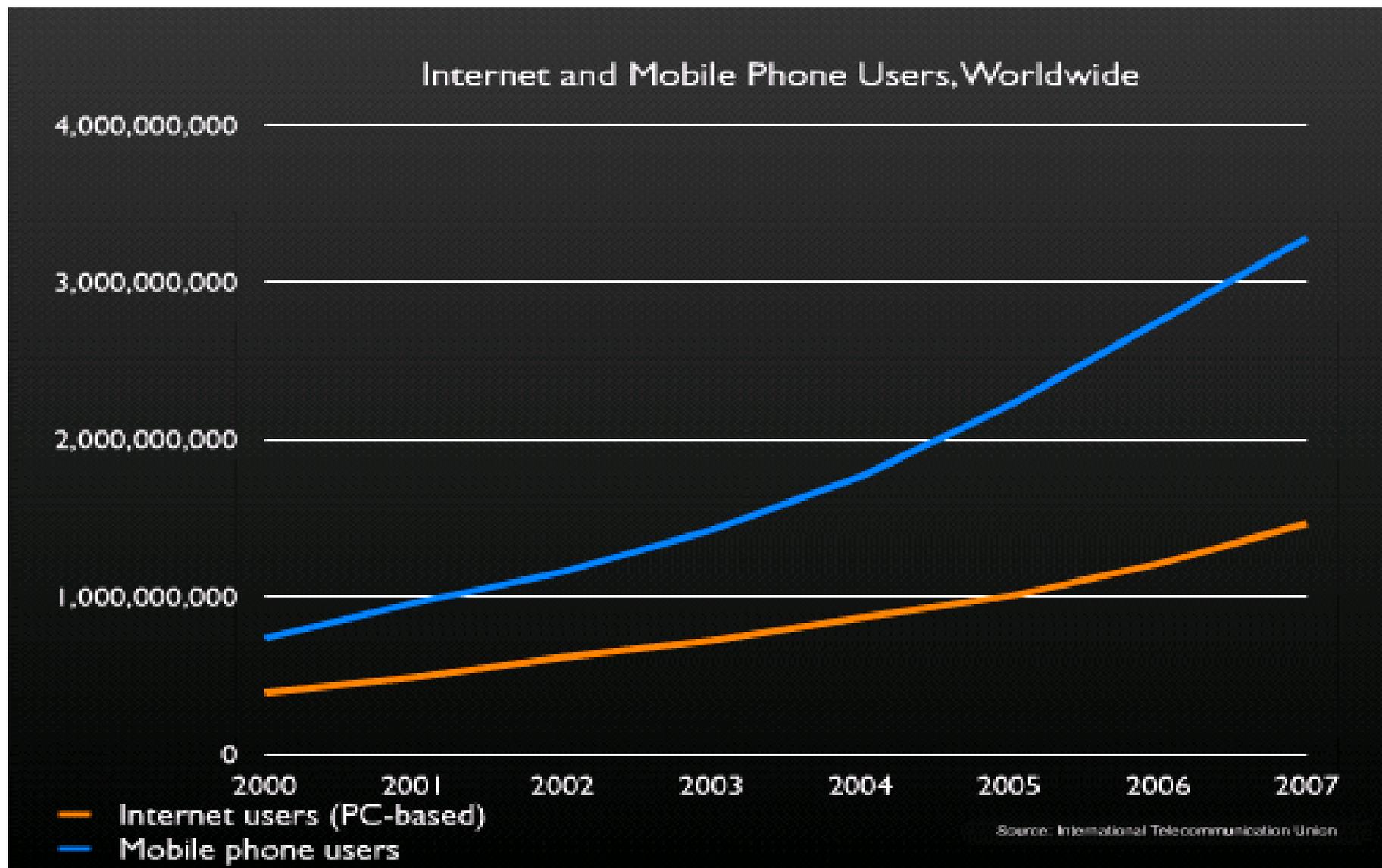


Android
Qu'est-ce
que c'est ?

- La génération actuelle des 'SmartPhones' est le fruit de la fusion de deux innovations majeures des années 90
 - Le téléphone portable
 - Les assistants personnels PDA



- **Les principaux Smartphones**
 - 2002, le BlackBerry
 - Concrétisation de l'accès à internet en mobilité, surtout aux mails
 - Positionnement entreprise et élitiste
 - 2007, l'iPhone : le pari réussi d'Apple : « Internet, le vrai! »
 - Saut technologique et d'ergonomie important pour le marché grand public
 - Emergence d'offre d'accès à internet illimité chez les opérateurs
 - Proposer des applications par l'Apple Market
- **139 millions de smartphones vendus en 2008**
 - Sur un milliard de téléphones



- **2005 : Google rachète Android (Inc)**
 - Une startup de Palo Alto
- **Google a annoncé Android fin 2007**
 - Système complet et ouvert pour les Smartphones
- **Le point clé est l'ouverture**
 - Open-source, sous licence Apache
 - Très loin du modèle fermé d'Apple
- **De nombreux acteurs ont rejoint Google**
 - Création de l'Open Handset Alliance : <http://www.openhandsetalliance.com/>
 - Quelques membres : HTC, Motorola, Samsung, Toshiba, Acer, Asus, Vodafone, T-Mobil, Texas-Instrument, Intel, Sony-Ericsson, Ebay, Google...



**G1 – HTC Dream
(Orange)**



**G2 – HTC Magic
(SFR)**



**Samsung i7500
Galaxy
(Bouygues)**

**Autres Modèles à
venir :**

HTC Hero

Motorola Morisson

Haier H7

Huawei U8230

SciPhone N12

...

- Eric Schmidt (CEO de Google)
- « *Android adoption is about to explode. You have all the necessary conditions.* » (mi octobre 2009)
- Douze modèles de téléphones, 26 pays, 32 opérateurs.
- L'opérateur Verizon mise sur le Motorola Droid
- Android 2.0 (26/10/09)
 - Bluetooth
 - Support d'Exchange
 - Navigateur HTML5
 - Multitouch
 - Nouvelle API Contact
 - Android Market
 - Text-to-speech engine



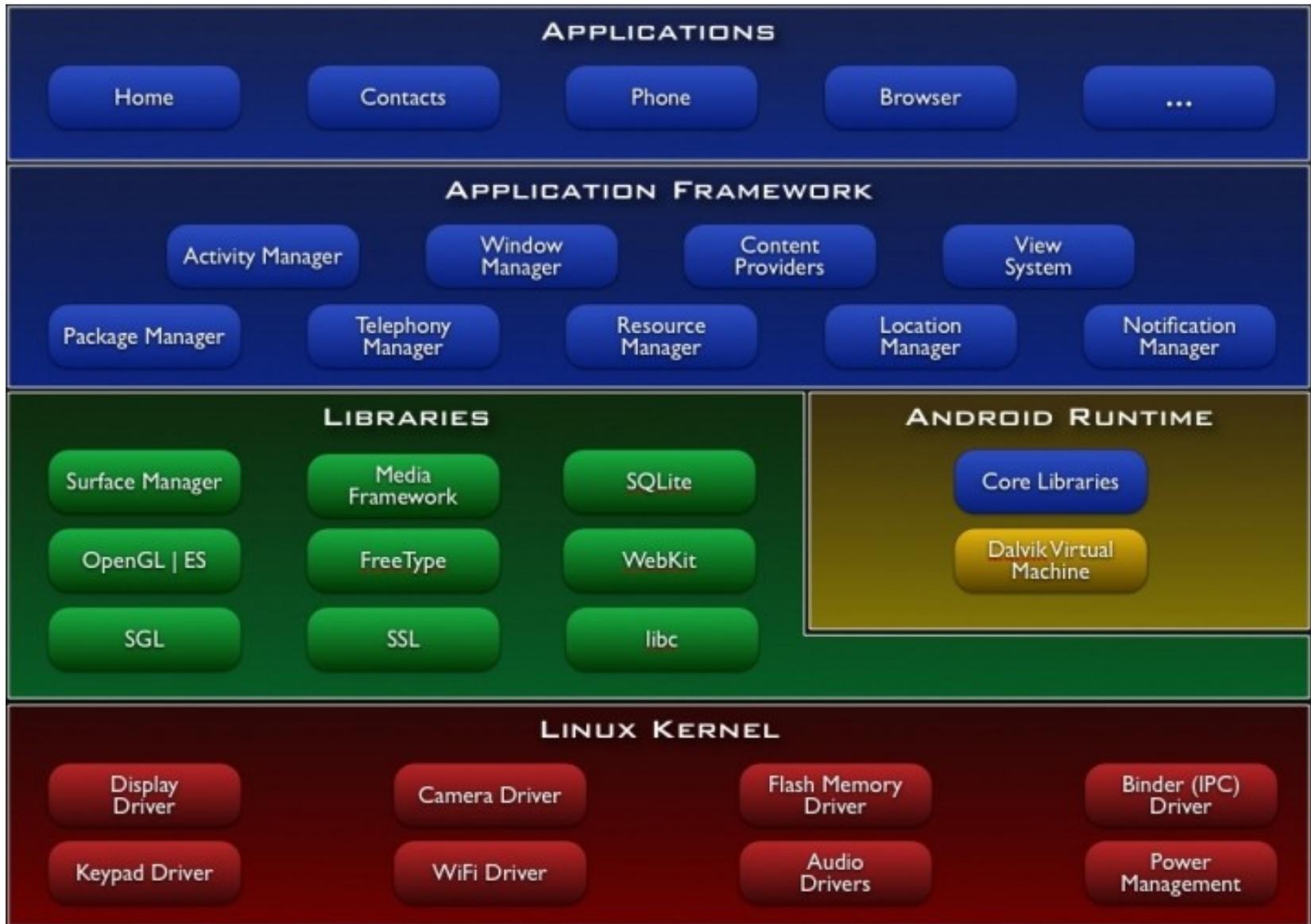
- **Nouveaux supports : notebook, tablets...**
 - Mais aussi dans des processeurs destinés au monde industriel
- **Etude : l'utilisation majoritaire du téléphone en 2012**
 - Les recherches sur Internet
- **Business : étude de Gartner**
 - Android 2e en 2012 (14,5%)
 - Derrière Symbian de Nokia (39%)
 - Devant Iphone (13,7%)
 - Windows Mobile (12,8%)
 - Blackberry (12,5%)





**Comment
ça fonctionne ?**

« Android is a software stack for mobiles devices that includes an operating system, middleware and key applications. »

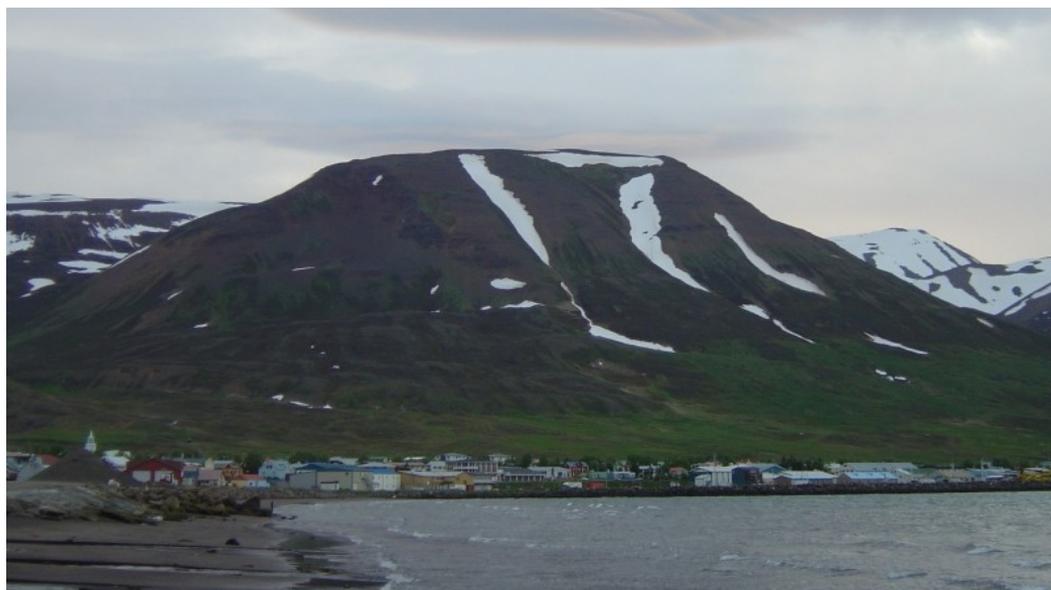


- **La plateforme Android se décompose en 5 couches**
 - Le noyau basé sur Linux 2.6 qui intègre notamment les drivers nécessaires (Wifi, Bluetooth, audio, ...)
 - Des bibliothèques C/C++ fournissant des fonctionnalités de plus haut niveau (moteur HTML WebKit, base de données SQLite, ...)
 - Un environnement d'exécution pour les applications dont le socle est une machine virtuelle (Dalvik)
 - Un framework Java exploitable par toutes les applications s'exécutant sur la machine virtuelle Dalvik. Propose, notamment, des notions permettant la structuration des applications et leur coopération
 - Des applications, écrites en Java, couvrant les besoins classiques (gestionnaire de contacts, navigateur Web, client messagerie, ...)

- **Non pas GNU libc mais Bionic sous licence BSD**
- **Critiques de Matt Porter (in « Mythbusters: Android »)**
 - *« The presentation shows how Google has simply thrown 5-10 years of Linux userspace evolution into the trashcan and re-implemented it partially for no reason. »*
- **Impossible de compiler plusieurs projets en C : VLC...**
 - *« In many cases, it seems like licensing considerations had a profound influence on technical decisions. The Android developers clearly went to great lengths to avoid using code that is distributed under GNU's General Public License (GPL). »*
 - *« The pervasive use of ASL throughout the Android middleware layer makes it possible for handset makers and mobile carriers to make proprietary customizations on top of the platform stack. »*

- **Dalvik, une machine virtuelle optimisée pour :**
 - Une machine avec un processeur à CPU lent
 - Une machine avec relativement peu de RAM (64Mo)
 - Une machine sans espace de Swap
 - Et qui marche avec une batterie

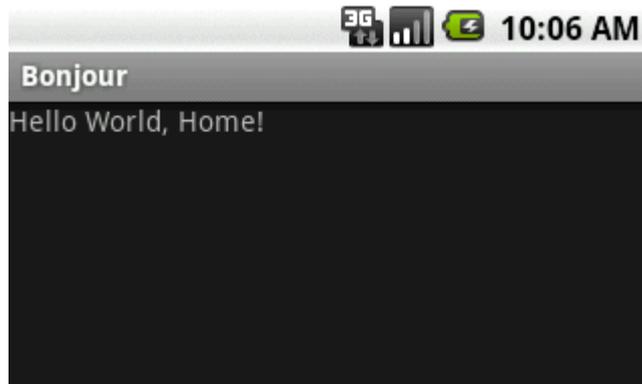
**Dalvik, petit port
de pêche en Islande**

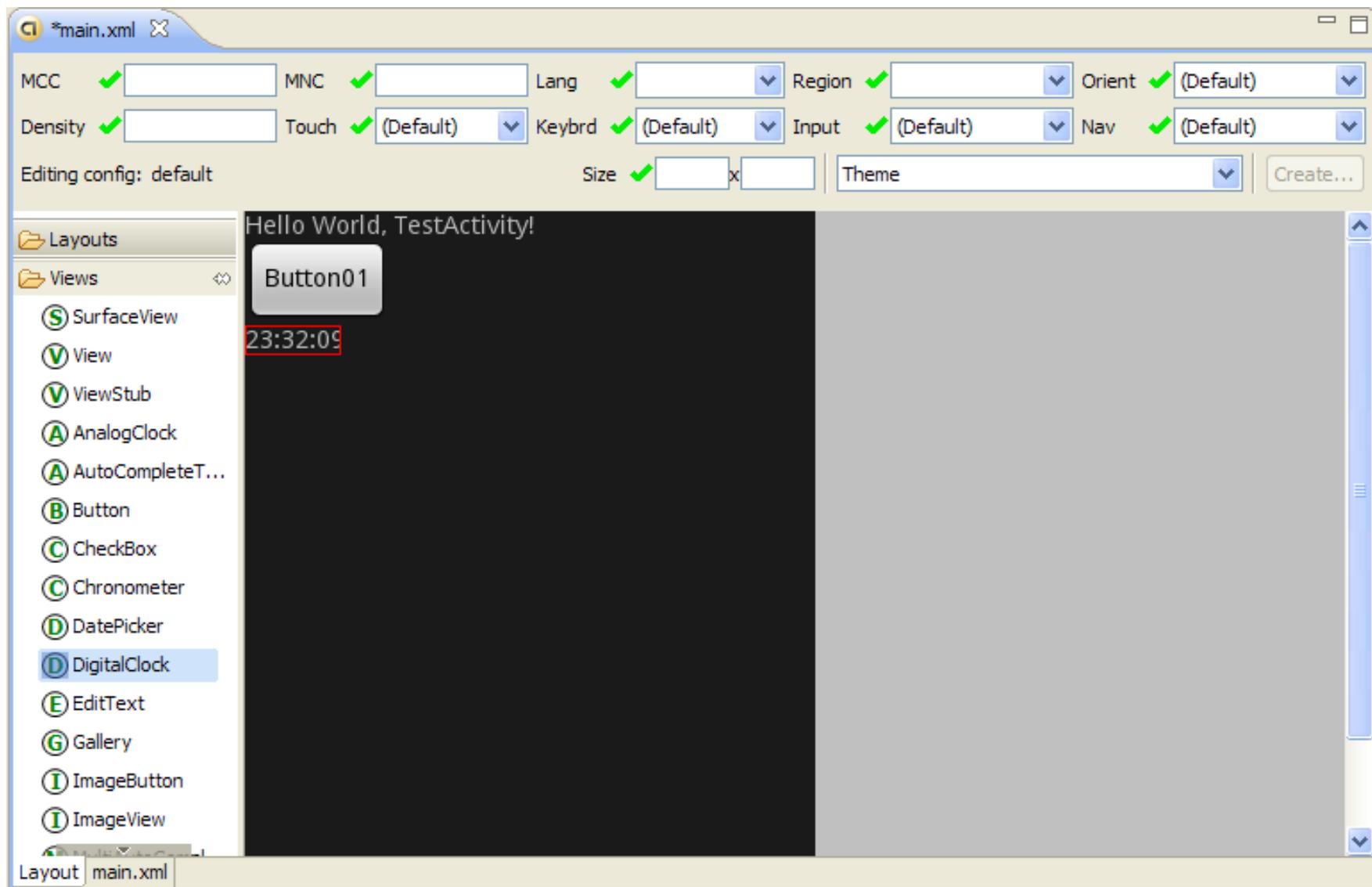


- **On programme avec un java quasi équivalent à j2se**
- **Les fichiers .class sont convertis en .dex**
 - Eviter les répétitions
 - Faire différentes optimisations : polls de types ...
 - Le format du bytecode sous Dalvik est spécifique
- **Tout cela permet de gagner de la place**
 - Un .dex décompressé est plus petit que le fichier jar correspondant
- **Pas de JIT, pas de génération de code à la volée**
 - Pas d'AOP.
 - Pas de Groovy, pas de Clojure... mais du Scala
- **Un Garbage Collector encore peu évolué : mark&sweep**
 - Pas de garbage collector générationnel

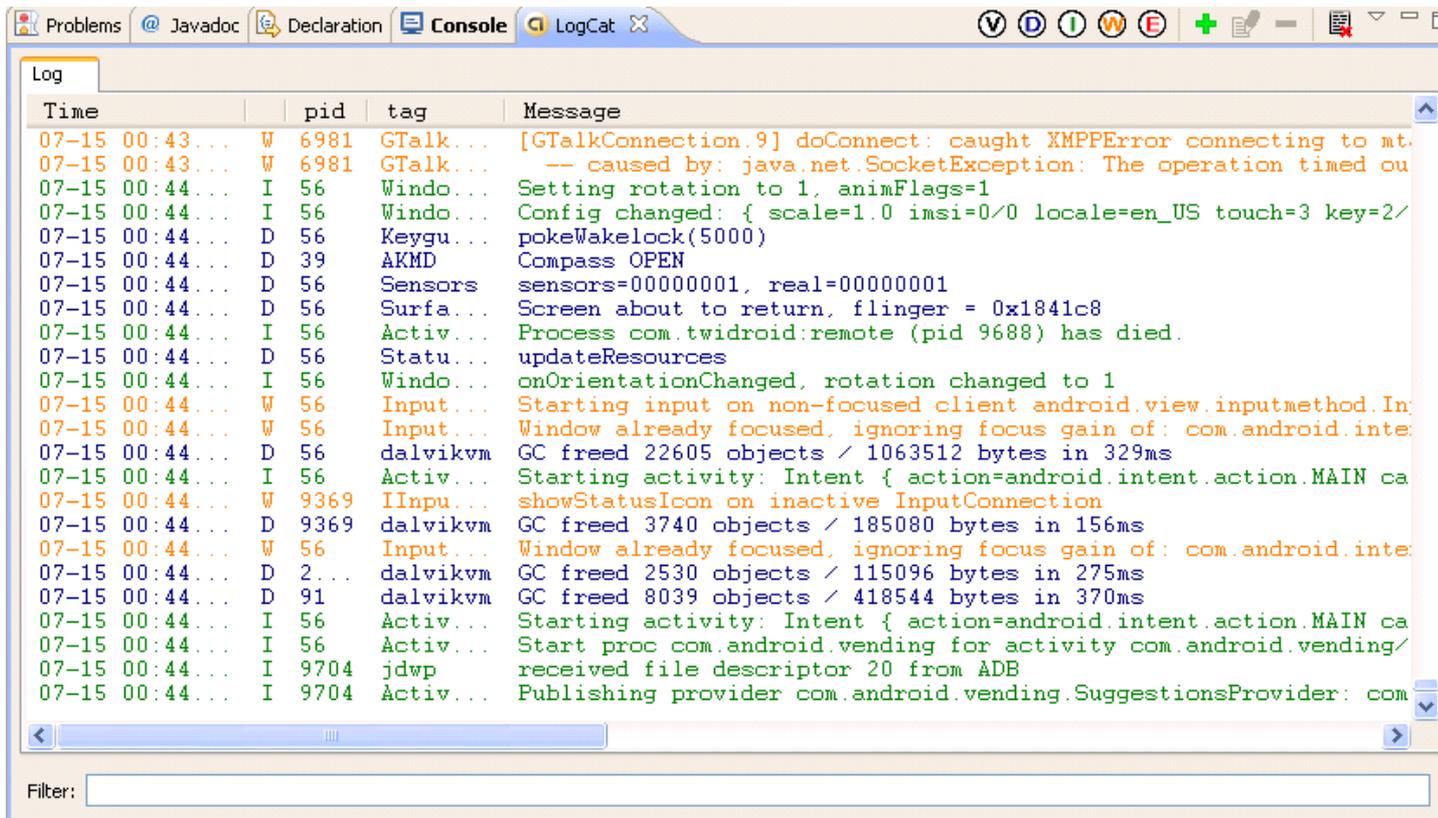
- **On peut développer des applications principalement**
 - En Java (utilisation du SDK d'Android)
 - En client léger (Webkit sous Safari 4)
- **Dépend de ses contraintes**
 - Parc homogène d'Androids/parc hétérogène de smartphones
 - Accès au réseau constant
 - Facilité à déployer
 - Sécurité
 - ...
- **Application qui nécessite des optimisations**
 - Possibilité d'intégrer du code compilé en natif (avec le NDK)
 - Appelé à partir de Java par des appels JNI (Java Native Interface)

- **Google propose un outillage dédié au développement d'applications pour Android**
 - ADT : Android Development Toolkit
 - Fonctionne sur Windows (x86), Mac et Linux
- **Cet outillage s'intègre à Eclipse**
 - Emulateur basé sur Qemu (Fabrice Quellard)
 - Débugage, logs
 - Editeurs d'interface graphiques





- Outil pour suivre les logs de son application
- Mais aussi des autres applications et du système
 - Le « News feed » du système



```
Log
Time      pid      tag      Message
07-15 00:43... W 6981 GTalk... [GTalkConnection.9] doConnect: caught XMPPError connecting to mt.
07-15 00:43... W 6981 GTalk... -- caused by: java.net.SocketException: The operation timed ou
07-15 00:44... I 56 Windo... Setting rotation to 1, animFlags=1
07-15 00:44... I 56 Windo... Config changed: { scale=1.0 imsi=0/0 locale=en_US touch=3 key=2/
07-15 00:44... D 56 Keygu... pokeWakelock(5000)
07-15 00:44... D 39 AKMD    Compass OPEN
07-15 00:44... D 56 Sensors sensors=000000001, real=000000001
07-15 00:44... D 56 Surfa... Screen about to return, flinger = 0x1841c8
07-15 00:44... I 56 Activ... Process com.twidroid:remote (pid 9688) has died.
07-15 00:44... D 56 Statu... updateResources
07-15 00:44... I 56 Windo... onOrientationChanged, rotation changed to 1
07-15 00:44... W 56 Input... Starting input on non-focused client android.view.inputmethod.In
07-15 00:44... W 56 Input... Window already focused, ignoring focus gain of: com.android.inte
07-15 00:44... D 56 dalvikvm GC freed 22605 objects / 1063512 bytes in 329ms
07-15 00:44... I 56 Activ... Starting activity: Intent { action=android.intent.action.MAIN ca
07-15 00:44... W 9369 IInput... showStatusIcon on inactive InputConnection
07-15 00:44... D 9369 dalvikvm GC freed 3740 objects / 185080 bytes in 156ms
07-15 00:44... W 56 Input... Window already focused, ignoring focus gain of: com.android.inte
07-15 00:44... D 2... dalvikvm GC freed 2530 objects / 115096 bytes in 275ms
07-15 00:44... D 91 dalvikvm GC freed 8039 objects / 418544 bytes in 370ms
07-15 00:44... I 56 Activ... Starting activity: Intent { action=android.intent.action.MAIN ca
07-15 00:44... I 56 Activ... Start proc com.android.vending for activity com.android.vending/
07-15 00:44... I 9704 jdwp    received file descriptor 20 from ADB
07-15 00:44... I 9704 Activ... Publishing provider com.android.vending.SuggestionsProvider: com
```

Filter:

- **Les applications qui s'exécuteront sur la plate-forme**
 - Utilisent le middleware java
 - Ont toutes accès aux mêmes fonctionnalités des appareils
- **Les applications sont toutes égales sur la plate-forme**
 - Vous pouvez remplacer les applications déjà installées
- **Les applications ont toujours une IHM en java**
 - Mais souvent des appels JNI aux fonctions système

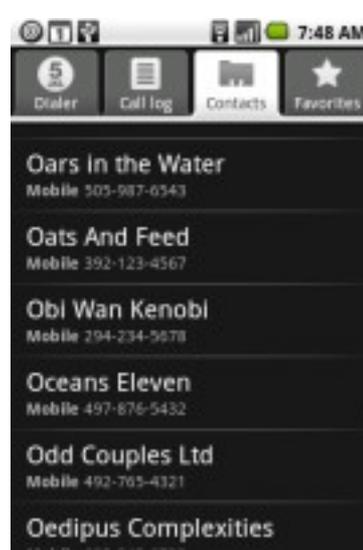
- **Classes principales de l'application**
 - Un composant n'est pas accessible par défaut par d'autres applications
- **Le système démarre le processus d'une application**
 - Quand un de ses composants est requis
 - Le système est responsable du cycle de vie des composants
- **Plusieurs points d'entrée pour une application Android**
 - Pas de méthode main
- **Configuration et composition de l'application**
 - AndroidManifest.xml

- **Principaux composants d'une application Android:**
 - Les activités (`activities`)
 - Les services (`services`)
 - Les `broadcast receivers`
 - Les `content providers`

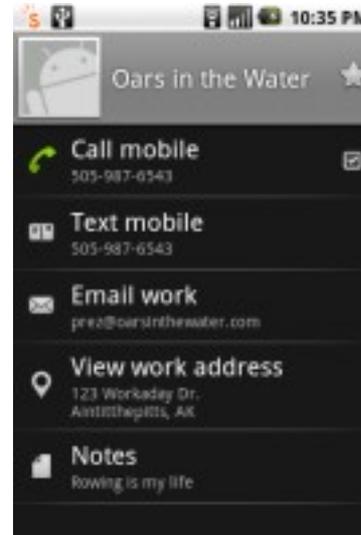
- **Une interface visuelle correspond à une activité**
 - Sorte de « page » ou d' « écran » dans une application
- **Chaque activité est indépendante des autres**
- **Exemple : l'application Dialer**



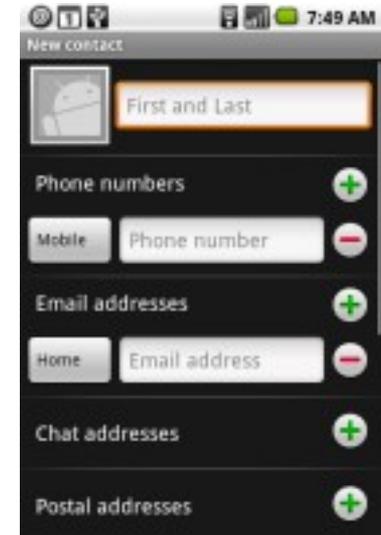
Dialer



Contacts



View Contact



New Contact

- **Pas d'interface visuelle pour les services**
 - Tournent en arrière plan
- **L'utilisateur garde la main**
 - Jouer de la musique, en saisissant son message
- **Expose une interface de communication**
 - Par exemple pour stopper ou reprendre la musique

- **Reçoit les messages qui circulent dans le système**
 - Communication inter-composants
 - Ou messages venus du système même
 - Exemple : le signal « batterie faible »
- **Plusieurs broadcast receivers par application**
- **Pas d'interface utilisateur**
- **Peut démarrer une activité en réponse à un message**
- **Peut alerter l'utilisateur**
 - Notification par le `NotificationManager`

- **Rend disponible les données de l'application**
- **Moyen de partage de données entre applications**
- **Stockage des données**
 - Fichier, base de données SQLite, réseau, etc...
- **Implémenter les six méthodes du Content Resolver**
 - query(), insert(), update(), delete(), getType() et onCreate()

- **Contient les informations utiles au récepteur**
 - L'action à exécuter
 - Les données sur lesquelles travailler
- **Contient les informations utiles au système Android**
 - Catégorie des composants qui doivent prendre en charge l'intent
 - Les instructions permettant de démarrer la cible
- **Peut contenir principalement**
 - Ou bien : La classe (java) du composant cible
 - Ou bien : L'action à exécuter
 - Les données
 - La catégorie de la cible
 - Des objets Extras
 - Les Flags

- **Passage de plusieurs paramètres dans un Bundle**
 - Permet de passer une série de données au travers de l'Intent

```
Bundle bundle = new Bundle();
bundle.putString(NotesDbAdapter.KEY_TITLE, mTitleText.getText().toString());
bundle.putString(NotesDbAdapter.KEY_BODY, mBodyText.getText().toString());
bundle.putSerializable(NotesDbAdapter.KEY_ATTACHED, anObject);

if (mRowId != null) {
    bundle.putLong(NotesDbAdapter.KEY_ROWID, mRowId);
}

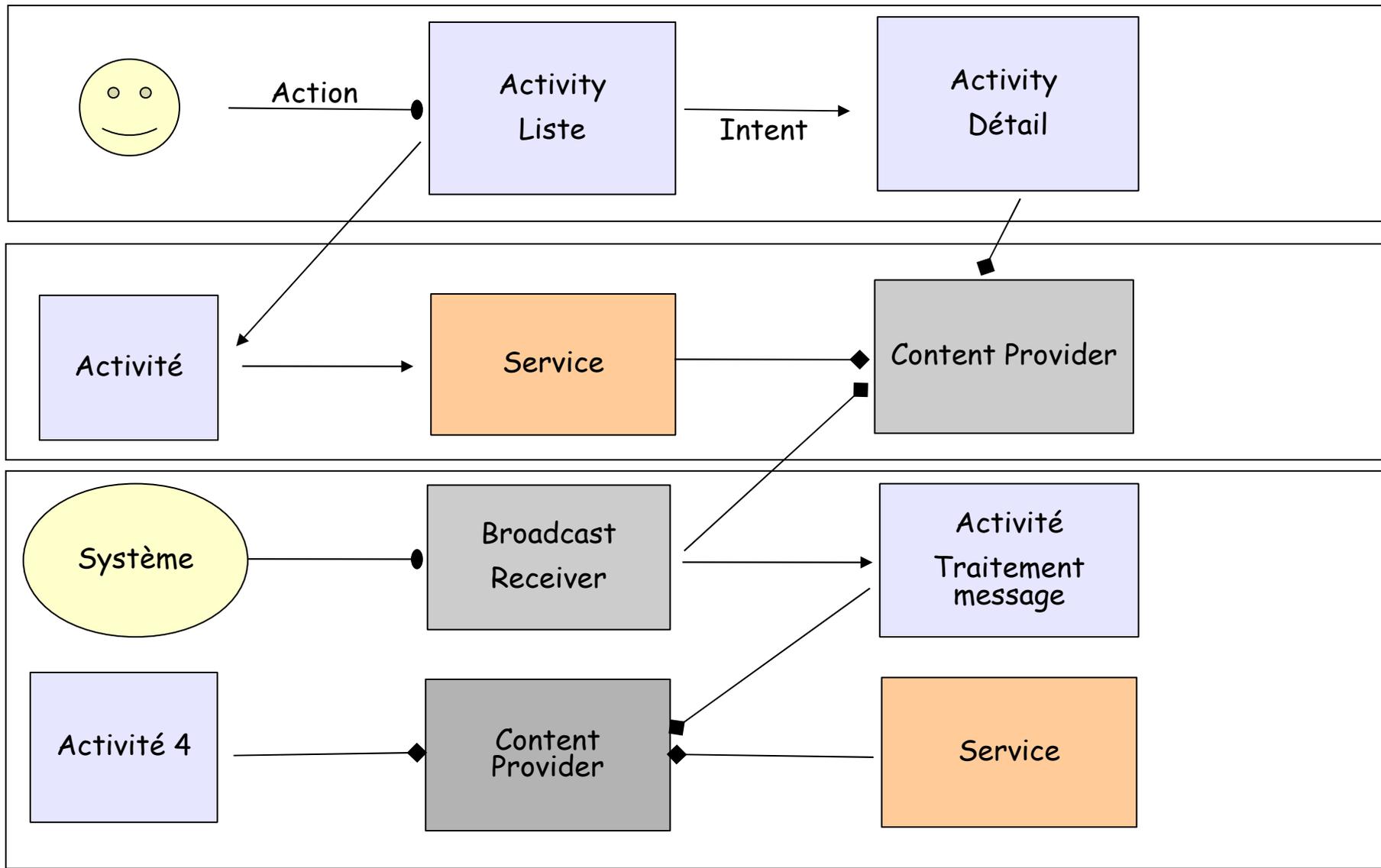
Intent mIntent = new Intent();
mIntent.putExtras(bundle);
setResult(RESULT_OK, mIntent);
finish(); // Appeler à la fin d'une Activité.
```

```
Bundle extras = intent.getExtras();
String title = extras.getString(NotesDbAdapter.KEY_TITLE);
String body = extras.getString(NotesDbAdapter.KEY_BODY);
Object anObj= extras.getSerializable(NotesDbAdapter.KEY_ATTACHED);
```

- **Deux groupes d'intents**
 - Les intents explicites
 - Les intents implicites
- **Intent explicite : au sein d'une application**
 - Le nom du composant cible est indiqué
 - Sert à communiquer dans la même application
- **Intent implicite : activer un composant qu'on ne connaît pas**
 - Le système doit chercher le composant le plus approprié
 - Couplage lâche entre composants et donc entre applications

- **L'intent explicite est délivré à la cible désignée**
 - Seul le nom du composant est nécessaire au système
- **Intent implicite**
 - Le système doit chercher le composant le plus approprié
- **Des filtres d'intent sont associés aux composants**
 - Peuvent aussi recevoir des intents
 - Publient les fonctionnalités du composant
 - Délimitent les intents que le composant peut traiter

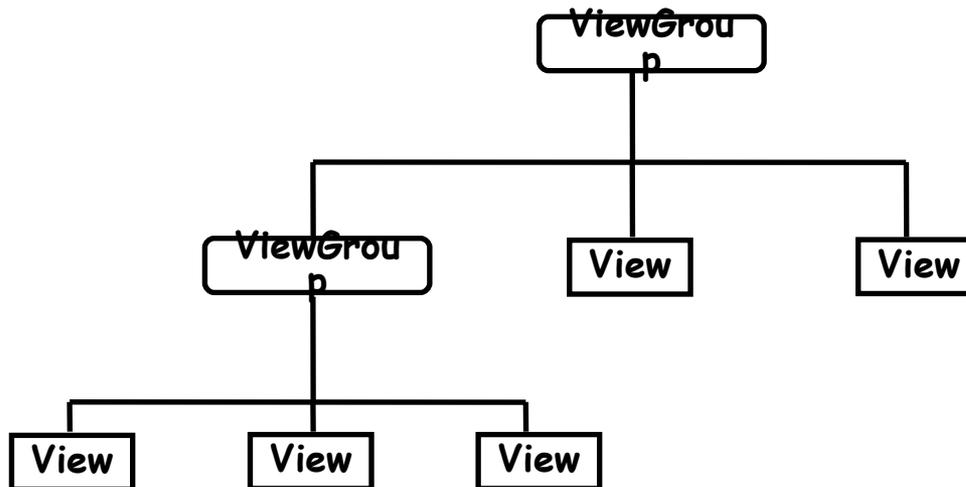
```
<intent-filter . . . >  
    <action android:name="com.example.project.SHOW_CURRENT" />  
    <action android:name="com.example.project.SHOW_PENDING" /> . . .  
</intent-filter>
```



- **Une activité a essentiellement 3 états**
 - Active
 - Elle est au sommet de la pile
 - En pause
 - Arrêt
- **La méthode procédurale**
 - En Java
 - Implique la construction de l'UI depuis le code
 - Exemple: Swing
- **La conception déclarative**
 - En XML
 - N'implique pas de code pour l'UI
 - Exemple: page web HTML
- **Techniques équivalentes**
 - Correspondances entre attributs XML et méthodes java

- **Sur Android les deux méthodes sont possibles**
- **La méthode procédurale**
 - La méthode procédurale en Java
 - La méthode déclarative en XML
- **Techniques équivalentes**
 - Correspondances entre attributs XML et méthodes java
- **Privilégier autant que possible la méthode déclarative**
 - Séparation entre la présentation et le code
 - Et pas de perte de performance : le xml est compilé.
- **Mais continuer à utiliser la méthode procédurale**
 - Pour des modifications dynamiques de l'interface graphique

- **Définition d'une interface Android**
 - Hiérarchie de vues et de groupes de vues
 - La racine est un ViewGroup
 - Fait de widgets (View) et de layouts (ViewGroup)
 - Hiérarchie définie en XML, par programme, ou un mixte des deux



- **Un fichier Layout est un fichier XML qui décrit l'écran**
 - Contient la hiérarchie des objets pour une page
 - Est stocké sous res/layout
 - Prévisualisation de l'interface
 - Onglet layout sous Eclipse

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    />
</LinearLayout>
```

- **Pour afficher l'écran à partir d'une activité**
 - Appeler la méthode setContentView()
 - Prend en paramètre la référence à la racine de l'arbre

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main_layout);
    ...
}
```

Paramètre indiquant quel fichier layout est utilisé.

A la compilation, chaque fichier layout est compilé en une ressource View

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    LinearLayout layout=new LinearLayout(this);
    TextView textView=new TextView(this);
    textView.setText("Hello, Android world!");

    layout.addView(textView);
    setContentView(layout);
}
```

On peut définir l'interface de manière programmatique

- **Petite base de données créée en 2000**
 - Ecrite en C
 - Suit en grande partie le standard SQL92
 - De petite taille et ne requérant pas d'administration
 - Stockage d'une base sous forme d'un seul fichier
- **Moteur de base de données le plus déployé au monde**
 - Base embarquée : Firefox, Skype, Google Gears... Et l'Iphone
- **Open source**
- **Pas de licence mais une bénédiction**

*May you do good and not evil
May you find forgiveness for yourself and forgive others
May you share freely, never taking more than you give.*

- **Possibilité d'utiliser le réseau pour l'accès aux données**
 - Utilisation des sockets
 - Pour le transfert de données depuis ou vers un serveur
 - Détails d'implémentation du serveur à la charge du développeur
 - Utilisation de HTTP
 - Le serveur HTTP prend à sa charge tous les détails liés aux sockets
 - Vous vous concentrez sur l'application cliente android
 - Simple requête HTTP avec l'API java.net: un seul thread
 - Requête HTTP avec HttpClient: thread multiple
 - Utilisation des web services
 - REST – Representational State Transfer
 - POX - Plain Old XML
 - JSON – JavaScript Object Notation
 - SOAP – Simple Object Access Protocol

- **Via des intents**
 - Avec l'action Intent.ACTION_CALL
- **Exemple passer des appels téléphoniques**
- **Via des intents**
 - Avec l'action Intent.ACTION_CALL
 - Pour lancer un appel sans l'intervention de l'utilisateur
 - l'uri tel: indique numéro à appeler
 - Avec l'action Intent.ACTION_DIAL
 - Pour lancer le numéroteur (dialer)
 - L'uri tel: indique le numéro qu'on souhaite appeler

Passer des appels (2)

```
dialintent = (Button) findViewById(R.id.dialintent_button);  
dialintent.setOnClickListener(new OnClickListener() {  
    public void onClick(View v) {  
        Intent intent = new Intent(Intent.DIAL_ACTION,  
        Uri.parse("tel:" + NUMBER));  
        startActivity(intent);  
    }  
});
```

Utilisation de
DIAL_ACTION

On y inclut le numéro

```
callintent = (Button) findViewById(R.id.callintent_button);  
callintent.setOnClickListener(new OnClickListener() {  
    public void onClick(View v) {  
        Intent intent = new Intent(Intent.CALL_ACTION,  
        Uri.parse("tel:" + NUMBER));  
        startActivity(intent);  
    }  
});
```

Utilisation de CALL_ACTION

- Réduire le nombre d'instanciation
- Car cela provoque le Garbage Collector
 - Et celui-ci arrête tout pour faire son travail. Et il est lent.
 - Exemple lorsqu'on scroll vite une ListView. L'interface se gèle
 - Garbage Collector
- Prudence sur certaines méthodes

onTouchEvent peut être appelé
80 à 90 fois en une seconde.
Donc dans un onTouchEvent
il faut instancier
le moins de variables possibles

Performance sensitive paths

Measurement	onMeasure()
Layout	onLayout()
Drawing	draw() dispatchDraw() onDraw()
Events handling	dispatchTouchEvent() onTouchEvent()
Adapters	getView() convertView()





Conclusion

- **Android est un OS ouvert pour appareils mobiles**
 - Optimisations, architecture guidée par cette contrainte
- **C'est une solution qui a dépassé le stade de la beta**
 - Les mobiles et les applications Android existent d'ors et déjà
- **Il propose de quoi développer des applications en java**
 - Richesse des API : géolocalisation, services HTTP, sqlite, 2D, 3D...
 - Outillage et modèle de développement
- **Le système est constitué de composants qui communiquent par messages**
 - Architecturer son application pour répondre aux besoins synchrones, asynchrones, à la gestion de l'environnement...
 - On peut créer un écosystème d'applications sur son téléphone

- **C'est un système encore jeune – des défauts à corriger**
 - Garbage Collector assez simple
 - Défauts dans l'API. L'api Json ne gère pas les flux, il parse tout
 - Défauts dans l'Android Market – corrigé avec les nouvelles versions
- **Il s'est coupé d'une partie des mondes Linux et Java**
- **Google organise ce développement Open Source**
 - Est-ce mal ou est-ce pragmatique ?

- **Pour le grand public**

- De la concurrence pour l'iPhone
- Plus d'applications et plus de développeurs
- Des smartphones pas trop chers

- **Pour les développeurs**

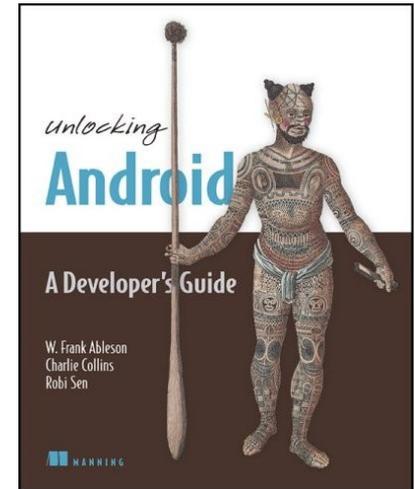
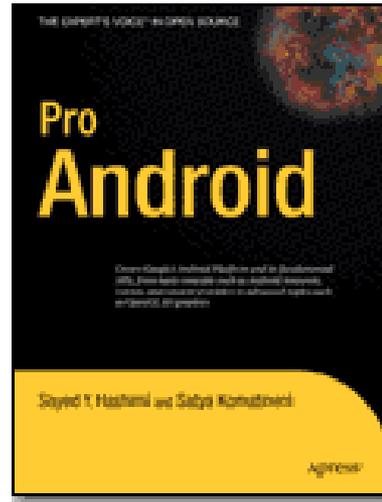
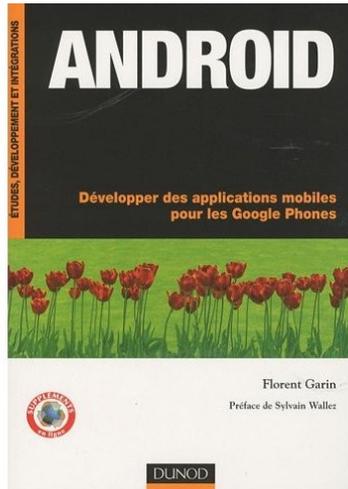
- Une solution complète pour des applications sur téléphone
- Un OS entièrement paramétrable, modifiable
- Facilité de prise en main et puissance des api et de l'architecture

- **Pour les entreprises**

- Un OS complet et gratuit
- La possibilité de faire des développements sur mobile
- La possibilité d'ouvrir le SI

- **A developer's introduction to Google Android**
 - <http://arstechnica.com/open-source/reviews/2009/02/an-introduction-to-google-android-for-developers.ars/2>
- **Android, Mythbusters and openness**
 - <http://blog.peijnik.at/2009/11/08/android-mythbusters-and-openness/>
- **Les castcodeurs : interview de Romain Guy**
 - <http://lescastcodeurs.com/2009/08/les-cast-codeurs-podcast-episode-8-interview-de-romain-guy-sur-google-android/>
- **Vidéos des conférences Google**
 - <http://www.youtube.com/user/GoogleDevelopers>

- **Nombreux livres en anglais et en français**
- **Mailing-lists actives**
 - **android-developers (>4500 messages/mois)**
 - **android-beginners (>1400 messages/mois)**



- **Android Developers :**
 - <http://d.android.com>
- **Blog officiel :**
 - <http://android-developers.blogspot.com/>
- **Blog Oxiane**
 - <http://blog.oxiane.com/tag/android/>

