TACT factory
mobile agency

Fournisseur **souple** et **réactif** d'applications mobiles innovantes avec une **méthodologie** projet **rigoureuse**.

Présentation

Mickael Gaillard (Architecte logiciel)
mickael.gaillard@tactfactory.com

Yoan Pintas (Lead Developper)
yoan.pintas@tactfactory.com

Planning session 1

Phase 1 :

Software Lexical
Software Concept
Android OS/Spec
Java (Level 2)
Mon Activity
Doc & Lien

Phase 2 :

L'interface Utilisateur
Les Intentions
Les Interfaces Avancées
Les Intentions Avancées

Phase 3 :

La Persistance
L’accès au réseau
Question ?

Software Lexical

Termes :

Source (code)
Binaire (code)
Compiler
Versioning (GIT)
Debug
Garbage Collector

Code Source

http://fr.wikipedia.org/wiki/Code_source

« Le code source est un texte qui représente les instructions qui doivent être exécutées par un microprocesseur. »

On évoque un élément par : (ordre)

- Espace de nom (Namespace)
- Nom du fichier (ou de la classe)
- Méthode ou ligne

Code Binaire

Peut-être exécutable ou de définition...

http://fr.wikipedia.org/wiki/Langage_machine

« Le langage machine, ou code machine, est la suite de bits qui est interprétée par le processeur d'un ordinateur exécutant un programme informatique. »

Compiler 1

COMPILER != INTERPRETER

Abstraction :

Code source => Exécution

Plus on est proche du langage machine plus l'application est :

- Rapide
- Peu consommateur en énergie
- Peu consommateur en espace de stockage

Compiler 2

Natif :

Code source >> Langage Machine == Exécution

Interpréter

Code source => Langage Machine (en mémoire) == Exécution

Hybride/Manager (Java, .net)

Code Source >> Langage bytecode (+ Machine Virtuel)
~= Langage Machine == Exécution

Versioning (GIT)

Pourquoi ? *L'erreur est humaine...*

http://fr.wikipedia.org/wiki/Gestion_de_versions

http://byte.kde.org/~zrusin/git/git-cheat-sheet-medium.png

Avantage !

- Sauvegarde multiple
- Retrouver du code fonctionnel
- Gestion des expérimentations (branch)
- Facilite le travaille collaboratif (et distant)

Debug

Pourquoi ? *L'erreur est humaine...*

http://fr.wikipedia.org/wiki/Débogueur

Avantage !

- Découvert dynamique
- Test dynamique
- Et les Threads ?

Les log (et console) ne sont pas des outils de Dev mais de Prod !!

Garbage Collector

Comment ?

Qui me référence ?!

http://fr.wikipedia.org/wiki/Ramasse-miettes_(informatique)

Software Concept

Design Patern

- Observateur
- Commande
- Transaction
- MVC
- Adapter / Bridge
- Comet
- ... (~ 300)

http://fr.wikipedia.org/wiki/Patron_de_conception

http://fr.wikipedia.org/wiki/Antipattern

Android OS/Spec 1

http://fr.wikipedia.org/wiki/Android

Le "Windows" de l'embarqué

- Multiple Architecture (ARM, ARM-V7, x86...)
- Multiple Affichage (Mobile, Tablet, TV, APN, Baladeur..)
- Internationalisation (i18n / i10n)
- Multiple Langage

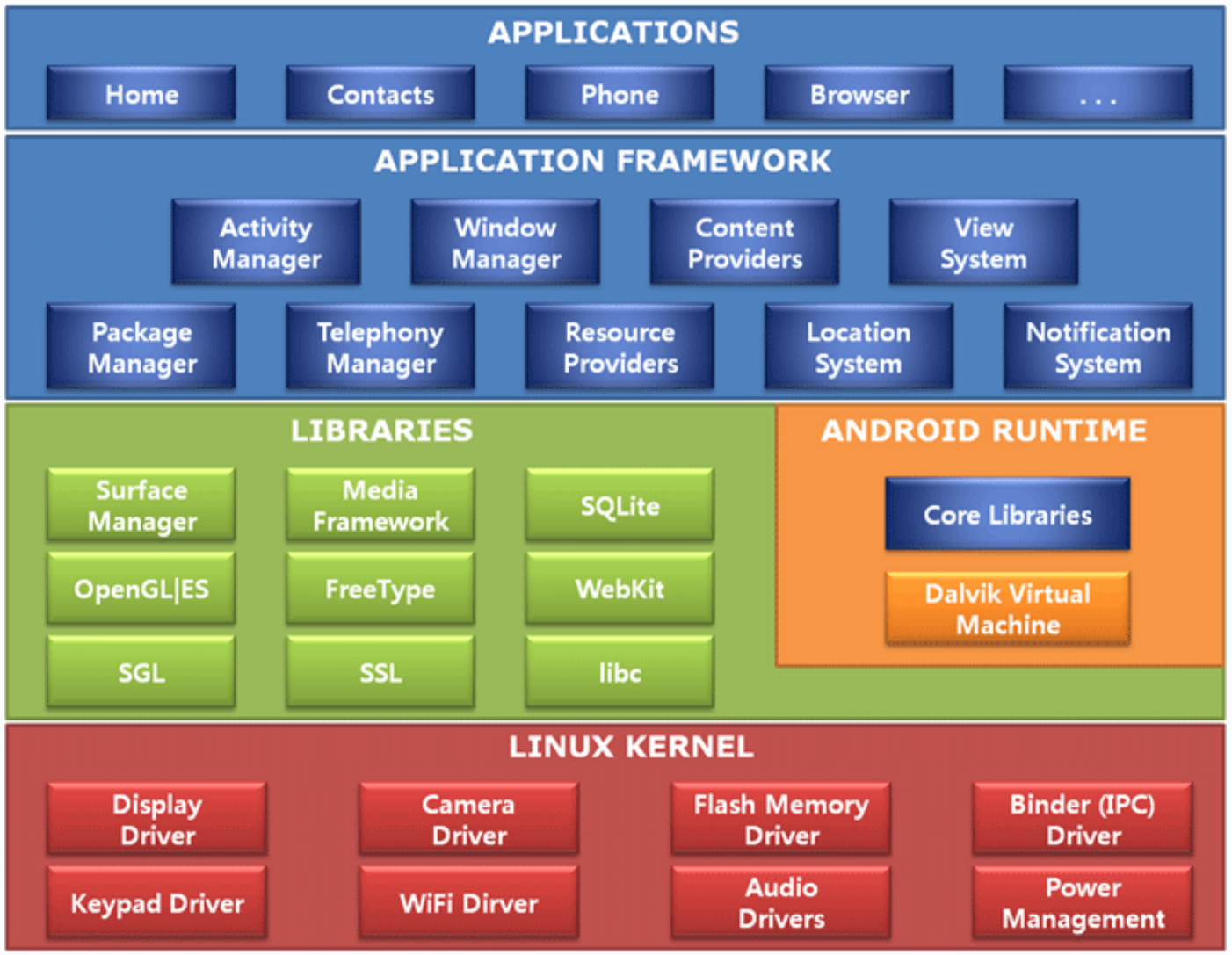
+

Sécurité / Sandbox

Android OS/Spec 2

Architecture

Application Intent - Provider
Framework Java-Dalvik (SDK)
Native binary (NDK)
Kernel Linux



Java 1

Du c/c++ sans pointeur

Primitive

- int
- char
- float
- ...

Objet

- String
- Array
- ...

Java 2

Passage de paramètre

- Primitive : by copy
- Object : by ref

```
public int cal(int i, Integer j) {                    // si i=2 et j=1
        int k = i + j ;                 int m = i – j ;
        i = k ;                         j = m ;
        return i ;
}                                                     //  i=? , j= ?, return ?
```

Java 3

Type class/interface/enum

- class : implémentation
- interface : contrat
- enum : valeur identifier

Comportement :

- Extend : Heritage direct
- Implement : Heritage indirect

Java 4

Porté : this ou base ou (local)

- this : porté de l'instance
- super: porté du parent
- local : porté d'une méthode

Visibilité : http://docs.oracle.com/javase/tutorial/java/java00/accesscontrol.html

- public
- protected (package ou class)
- private

Java 5

Package

Espace de nom (namespace) == Organisation

```
package tld.domain.projet.....

Import tld.domain.projet.MaClass ;
Import tld.domain.projet.* ;
```

Java 6

Pièges...

```
String t = new String('toto') ;
// (t == 'toto')  ?
// (t.equals('toto'))  ?
```

Java 7

Réflexion (IOC) : http://ricky81.developpez.com/tutoriel/java/api/reflection/

```
public class Exemple {
   public Exemple() { }
   public String getNom(Object o) {
      Class c = o.getClass();
      return c.getName();
   }
}
(Bar) Class.forName(Exemple).newInstance();
```

Java 8

Généricité : http://www.siteduzero.com/tutoriel-3-10421-la-genericite-en-java.htm

Template de C/C++

```
Vector<String> v = new Vector<String>() ;
public class Complex<T> {
        private T var;
        public Complex(T var) { this.var = var ; }
        public T getValue()  { return this.var ; }
}
```

Java 9

@annotation : (POA) http://adiguba.developpez.com/tutoriels/java/tiger/annotations/

```
public @interface MonAnnotation { }

@MonAnnotation
public class MaClasse { }
```

Mon Activité 1

Projet

Eclipse , shell... http://developer.android.com/tools/projects/projects-cmdline.html

Manifest

http://developer.android.com/guide/topics/manifest/manifest-intro.html

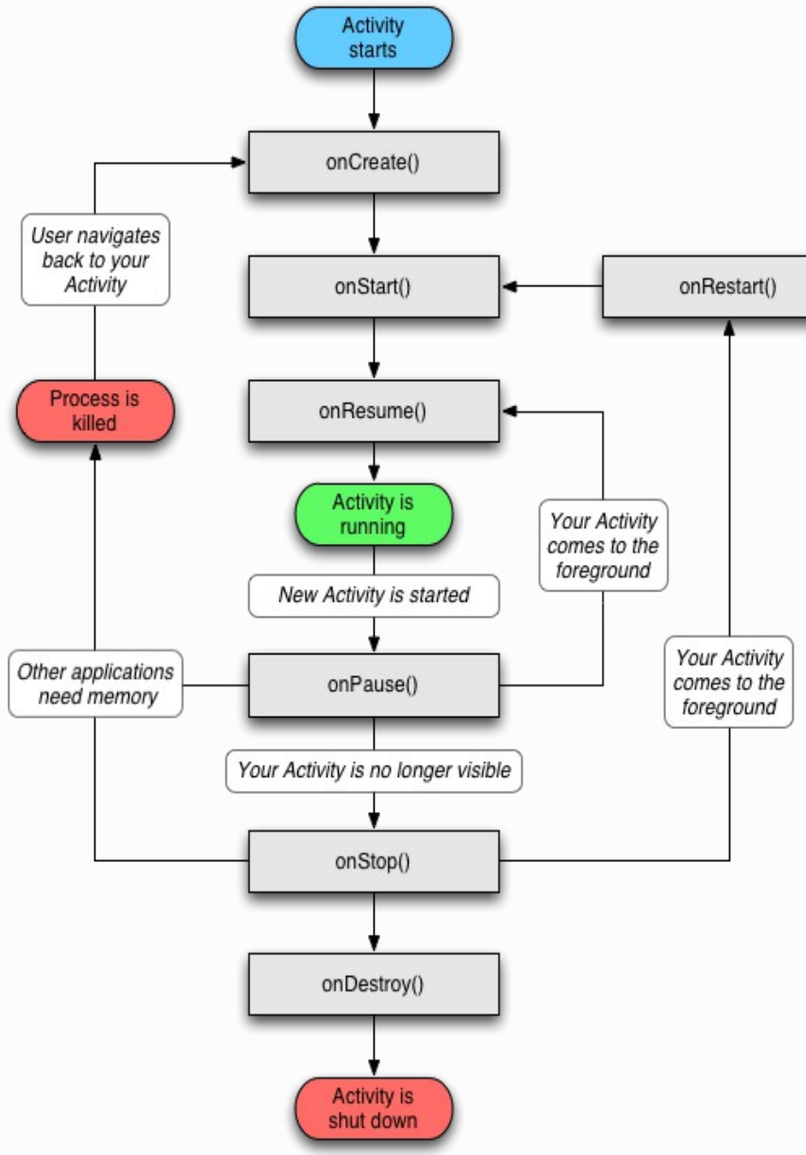
Composants :

- MyActivity.class
- **layout.xml**

Mon Activité 2

Activity, cycle de vie

http://developer.android.com/reference/android/app/Activity.html
http://developer.android.com/guide/components/activities.html



Et en background..

- Service
- AsyncTask
- Thread(Runnable)

Doc & Liens

Documentations

ISBN 2-212-09111-7, Conception et Programmation Orientées objet

ISBN 978-2-212-12587-0, Programmation Android

ISBN 2-212-11406-0, Eclipse et JBoss

Liens

http://developer.android.com/guide/components/index.html

Change log

- Mickael Gaillard 2012 : Initial document
- Yoan Pintas 2012 : Update
- Micakel Gaillard 2013 : Update visibility