Android

COURS ANDROID

9-13 janvier 2017

CESI

Formateur: Philippe Dutron

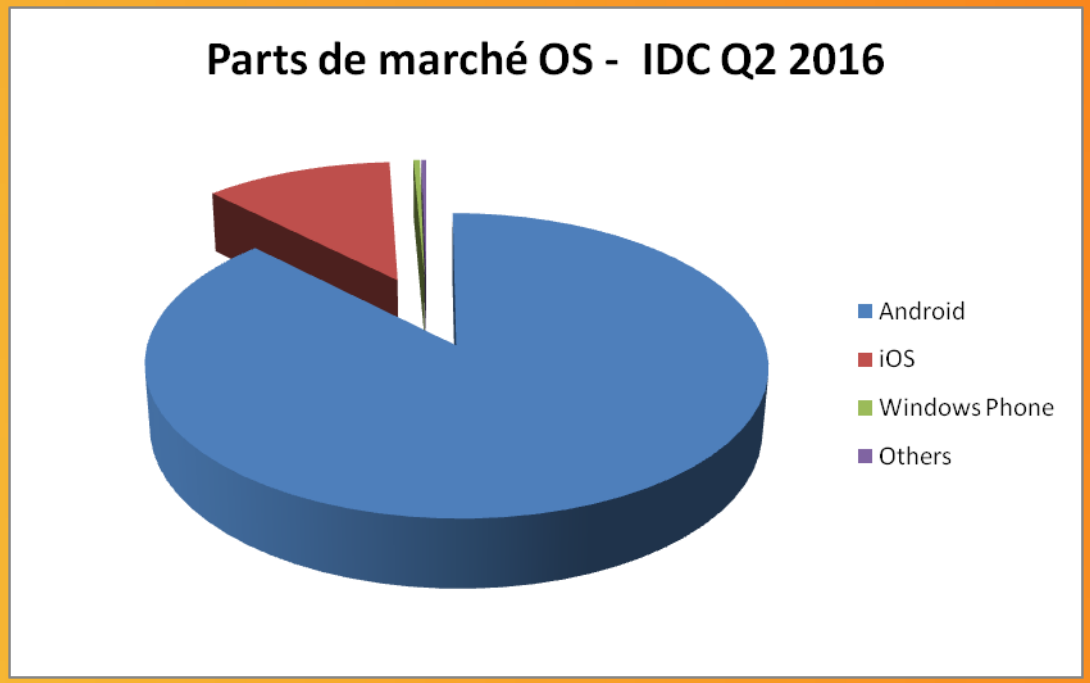
Android

L'une des piles logicielles permettant le développement de solutions mobiles et d'objets connectés

La pile Android

- Système d'exploitation : version modifiée de linux
- middleware, bibliothèques natives, la plupart écrites en Java: WebKit, OpenGL, FreeType, SQLite, Media, C runtime library (libc) etc.
- Une machine virtuelle d'éxécution (Dalvik VM)
- Un framework d'application
- Des applications natives : browser web, contact manager …

L'éco-système des OS pour Smart-Phones



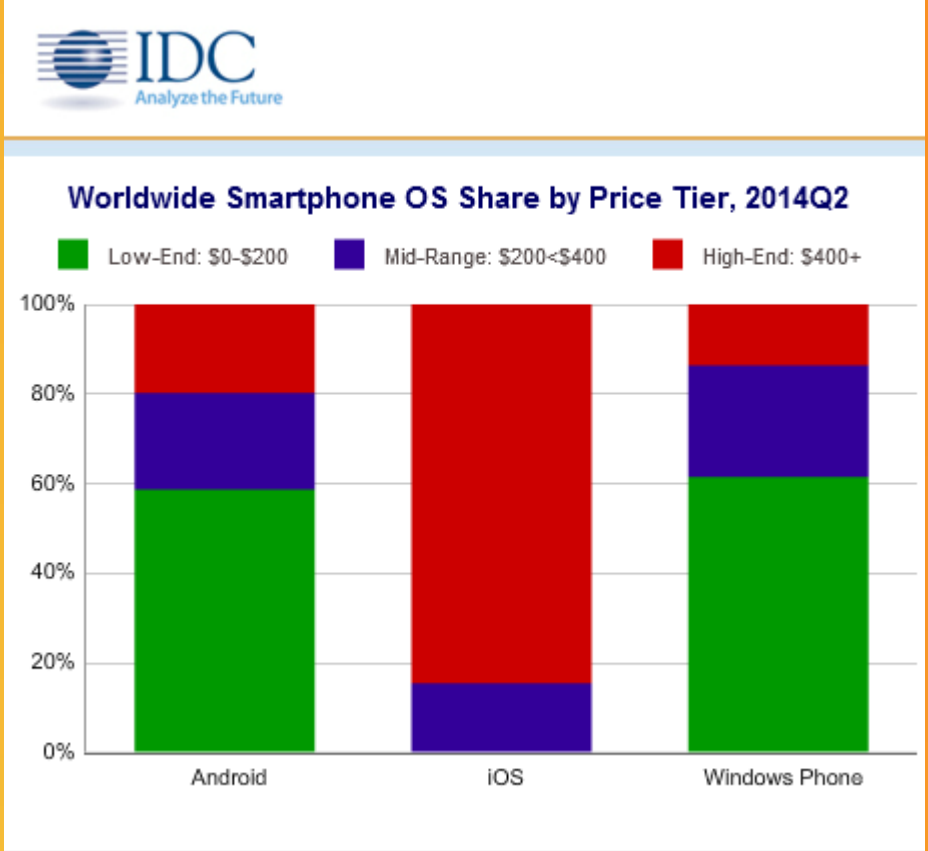
Android

OS pour Smart-Phones - tendances

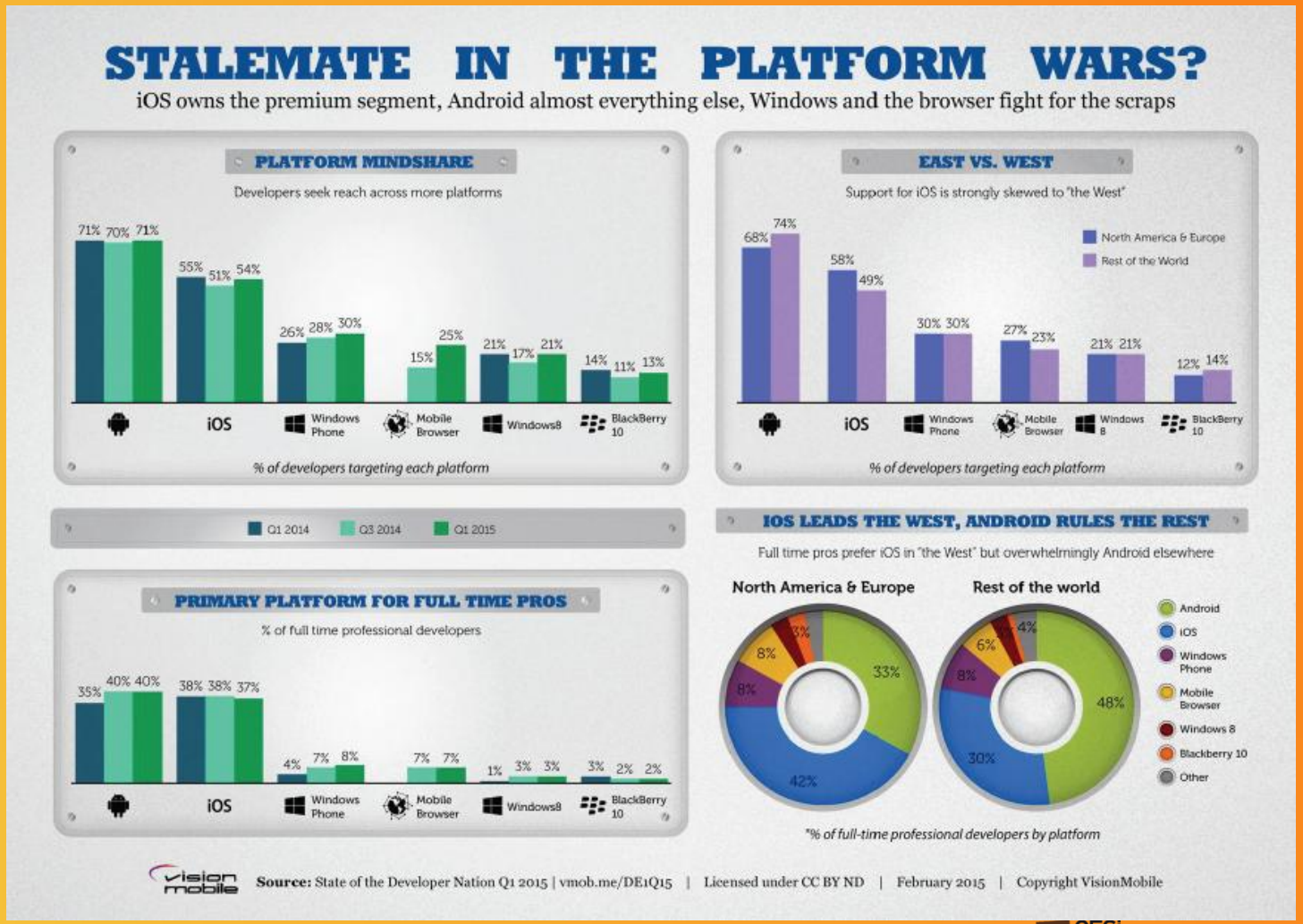
Period	Android	iOS	Windows Phone	Others
2015Q3	84.3%	13.4%	1.8%	0.5%
2015Q4	79.6%	18.6%	1.2%	0.5%
2016Q1	83.4%	15.4%	0.8%	0.4%
2016Q2	87.6%	11.7%	0.4%	0.3%

Source IDC 2016, Q2

OS pour Smart-Phones- Tendances



Positionnement des développeurs



Un peu d'histoire …

- Android
  - Android.Inc, Palo Alto Startup acquise par Google en 2005
  - Première beta release nov 2007
  - Première release de Google - Android 1.0 sept 2008
  - Premier téléphone basé sur Google Android OS phone et réalisé par HTC oct 2008
  - Fondation de l'Open Handset Alliance en 2007, crée par des OEM du monde du mobile, fabricants, Wireless carriers, fondeurs (Samsung, HTC, Mobile-T, TI, …)
  - Version d'Android la plus récente : Nougat

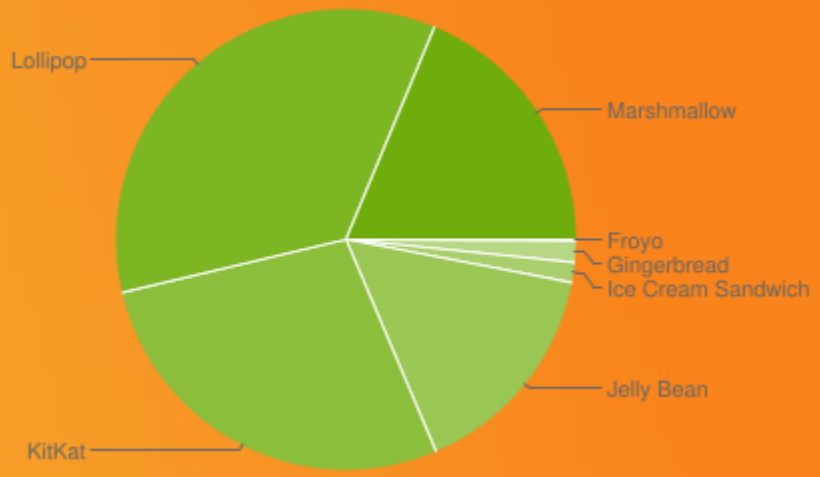
Design Challenges

- Gérer la compatibilité au regard des évolutions d'Android
  - Fonctionnalités offertes
  - Versions des plateformes
  - Tailles d'écrans

- Google Dashboard vous aide à connaitre l'environnement d'utilisation
  - https://developer.android.com/about/dashboards/index.html

Versions

Version	Code name	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	1.5%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.4%
4.1.x	Jelly Bean	16	5.6%
4.2.x		17	7.7%
4.3		18	2.3%
4.4	KitKat	19	27.7%
5.0	Lollipop	21	13.1%
5.1		22	21.9%
6.0	Marshmallow	23	18.7%



Data collected during a 7-day period ending on September 5, 2016.
Any versions with less than 0.1% distribution are not shown

Densités et taille d'écrans

- *Définitions utiles*
- *Resolution*
  - Nombre total de pixels constituant un écran.
- *Density-independent pixel (dp)*
  - Un pixel virtuel servant d'unité pour exprimer les dimensions d'un layout ou des positions sur un écran. Les dimensions dans cette unité s'expriment indépendamment de la densité des pixels.
    - Un pixel indépendant (dp) est l'équivalent d'un pixel réel sur un écran 160 dpi. ("medium" density screen). A l'éxécution, Android gère automatiquement la conversion des dimensions dp en dimensions de pixels réels. Il se fonde sur la connaissance de la densité de l'écran
    - Règle de conversion: dp vers pixels réels
    - 1 px = 1 dp * (dpi / 160).
    - Ex :écran de 240 dpi , 1 dp vaut 1.5 pixels.
    - Toujours utiliser des unités dp dans les application's UI- IHM

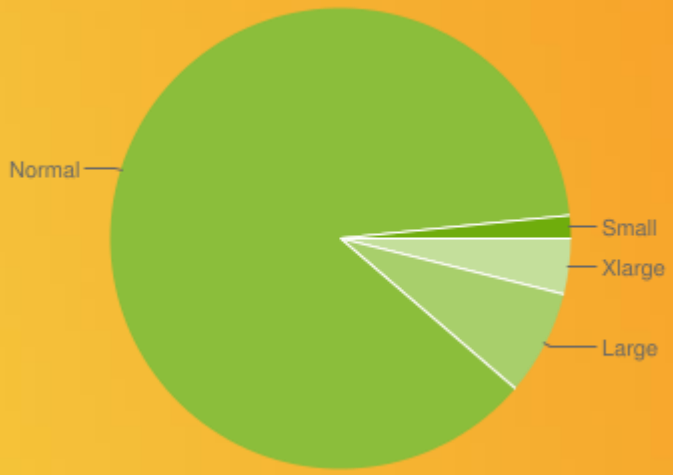
Segmentation selon les tailles d'écran

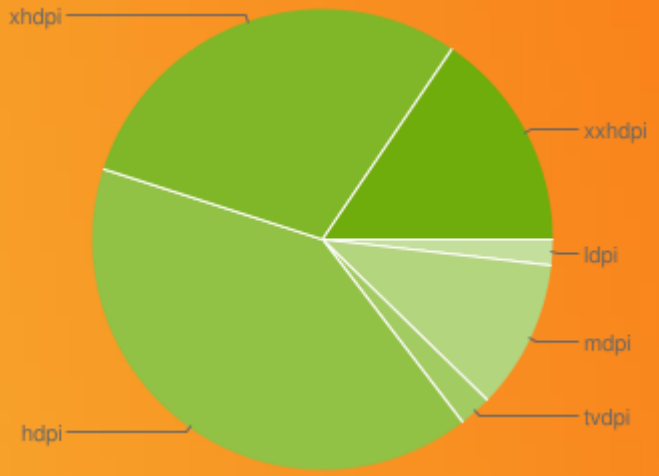
- *Référence de tailles utilisées par les Resources qualifiers*
  - res/layout/main_activity.xml //for handsets
  - res/layout-sw600dpi/main_activity.xml //for tablets
- *Tailles d'écrans, Sccreen Size*
  - Exprime la taille de diagonale de l'écran en dp.
    - 4 groupes : small, normal, large, and extra-large.
    - Small : 320x426 dp, normal: 320x470 dp, large : 480x640 dp, extra-large : 720x960 dp
    - Obsolète depuis API 13 . Remplacé par width (largeur d'écran)
    - Smallest width exemple : sw600dp
    - Available width exemple : w720dp
    - Available height exemple : h1024dp
- *Densité d'écran - Screen density*
  - S'exprime en dpi (dots per inch)
    - six densités de référence: low, medium, high, extra-high, extra-extra-high, and extra-extra-extra-high
    - ldpi: 0.75 (120dpi), mdpi:1.0 (160 dpi), hdpi :1.5 (240 dpi), xhdpi :2.0 '320 dpi), xxhdp:3.0 (480 dpi), xxxhdpi:4.0 (640 dpi)
    - mdpi : 160dpi (Baseline)
- *Orientation*
  - Orientation de l'écran vu de l'utilsateur
    - landscape or portrait (paysage ou portrait). L'orientation par défaut varie selon les modèles de smart-phones ou tablets. L'orientation peut changer au runtime, selon la rotation subie.
    - land, port

Distribution des tailles et densités d'écrans

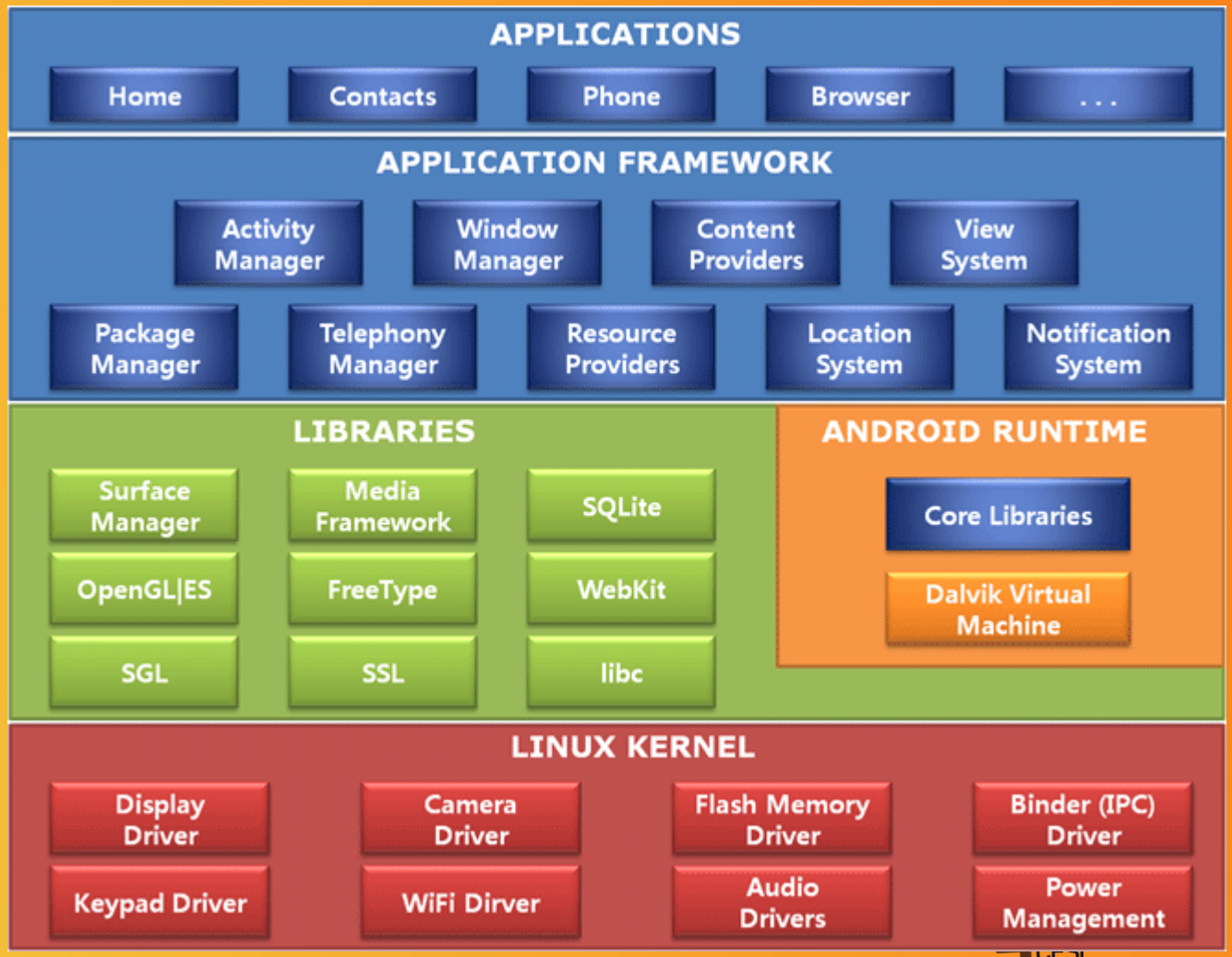
	ldpi	mdpi	tvdpi	hdpi	xhdpi	xxhdpi	Total
Small	1.6%						1.6%
Normal		3.5%	0.2%	39.5%	28.4%	15.5%	87.1%
Large	0.2%	4.1%	2.1%	0.5%	0.5%		7.4%
Xlarge		2.9%		0.3%	0.7%		3.9%
Total	1.8%	10.5%	2.3%	40.3%	29.6%	15.5%	

Data collected during a 7-day period ending on September 5, 2016. Any screen configurations with less than 0.1% distribution are not shown.





Architecture Android



Environnement de développement IDE- Eclipse

- Java : JSE 1.7 min , JSE 1.8 recommandé
- Eclipse : version Luna ou Mars
- Installation de Java
  - Télécharger l'exe à: http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html
  - Définir le path windows
    - PATH : %JAVA_HOME%
    - JAVA_HOME : C:\Programs\Java\jdk1.7.0_71
- Eclipse
  - Télécharger Luna at:
    - https://eclipse.org/downloads/
    - Eclipse->project->properties to set up JRE
- Ajouter les Android development tools pour Eclipse (ADT)
  - Dans menu Eclipse help->Eclipse install new software
  - Work with : https://dl-ssl.google.com/android/eclipse/
  - Install
- Ajouter Android SDK à Eclipse
  - Dans menu Eclipse window-> android sdk manager
  - Install android 5.01, tools, sdk, samples

Environnement de développement
IDE– Android Studio

- IDE basé sur IntelliJ
- Java : JSE 1.7 min , JSE 1.8 recommandé
- Installer Android Studio
  - Télécharger le pack AS 2.2 depuis site Googgle
- Installer Java
  - Télécharger jdk 8 depuis le site Oracle
    - PATH : %JAVA_HOME%
    - JAVA_HOME : C:\Programs\Java\jdk1.7.0_71
- Ajout d'Android Studio SDK tools
  - Dans Android Studio, click **SDK Manager** dans le menu
  - Selectionner la version Android la + récente ie 25
  - Rajouter Googgle USB driver dans la section extra
  - Installer Google Api image x86 64 System Image
  - Installer x86 emulator HAXM
- Ajouter la support library
  - ouvrir extras directory et ajouter
    - Android Support Repository
    - Android Support Library
- Pb connu d'Android Studio
  - Sous windows pour utiliser un device USB, installer ADBdriver,
  - Installer adbdriver pour le device concerné
  - ordinateurr, bouton droit, gerer, gestionnaire de peripherique, android device, bouton droit, mettre à jour le drver , choisr dans la liste sur ordinateur

IDE – Android Studio

- Principales actions clavier
- ActionAndroid Studio Key Command
  - Command look-up (autocomplete command name) CTRL + SHIFT + A
  - Project quick fixALT + ENTER
  - Reformat code CTRL + ALT + L (Win)
  - Show docs for selected APICTRL + Q (Win)
  - Show parameters for selected method CTRL + P
  - Generate methodALT + Insert (Win)
  - Jump to sourceF4 (Win)
  - Delete line CTRL + Y (Win)
  - Search by symbol name CTRL + ALT + SHIFT + N (Win)

Demonstration

- Bonjour à tous
  - Créer un répertoire workspace dans le répertoire courant
  - Démarrer Eclipse / Android Studio
  - Browse pour sélectionner workspace
  - Setup run configuration
  - Run application
    - À partir d'un émulateur AVD
    - Ou sur votre Smartphone connecté en USB

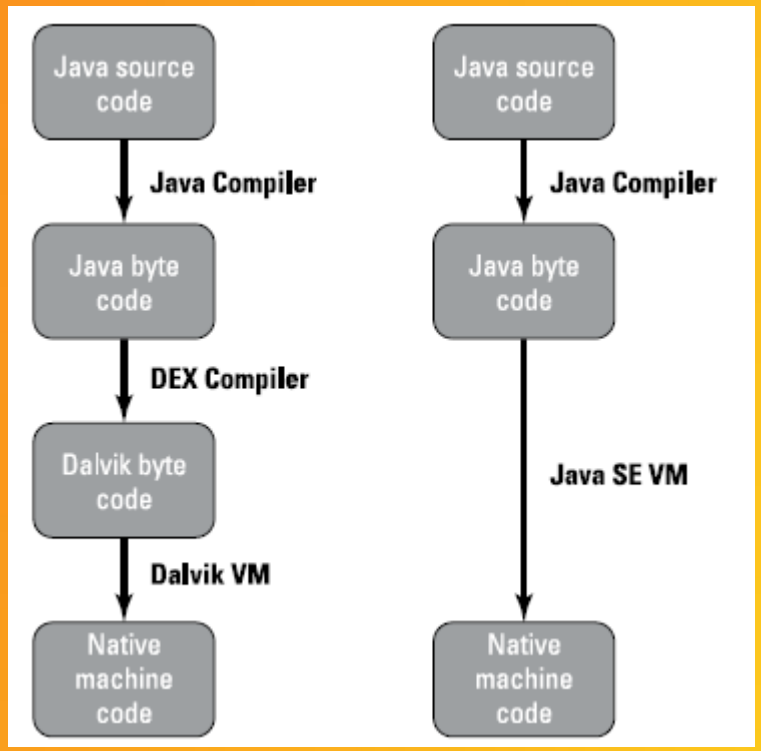
Set-up de l'avd

- Android Virtual Device
  - Pour tester votre application vous devez disposer
    - D'un device virtuel
    - D'un device physique
  - Utiliser ADV manager pour configurer un device virtuel
  - Déclarer le device à mettre en œuvre dans run / debug configuration menu

Utilisation du smartphone dans l'IDE

- Settings for USB connection and debug
- open the *Developer options* in the system Settings. On Android 4.2 and higher, the Developer options screen is hidden by default. To make it visible, go to**Settings > About phone** and tap **Build number** seven times. Return to the previous screen to find Developer options at the bottom.
- In the AndroidManifest.xml file, add android:debuggable="true" to the <application> element.

Flow d'éxecution Java vs Android



Organisation des fichiers– Android Studio

- Structure des dossiers
  - Projet
    - /java
      - Sources de l'application
    - /manifests
      - Meta data de l'application
    - /res
      - Ressources, string, layout, images, … utilisables dans l'application
    - /gradle scripts
      - Fichiers de dépendance permettant la compilation de l'application et la génération du binaire

Gestion de la compatibilité descendante

- Google offre des librairies qui assurent la compatilité descendante
- Déterminer la version minSdk à supporter
- Déterminer la version utilisée pour la génération de l'apk
- Déterminer la version cible

Les librairies de compatibilité sont téléchargeables depuis le SK

- ex
  - Appcomp - v7
  - Support - v4

  - Dans sdk manager
  - selectionner extras
    - Sdk repository
    - Sdk support lib

Applications Android

- Fondements
  - Application = ensemble de composants coordonnés pour atteindre un objectif
  - En tant que programmeur votre tache est de spécifier et developper les composants constitutifs de l'application
    - Model - View- Controller
    - Android fournit des API qui facilitent/accélèrent vos développements

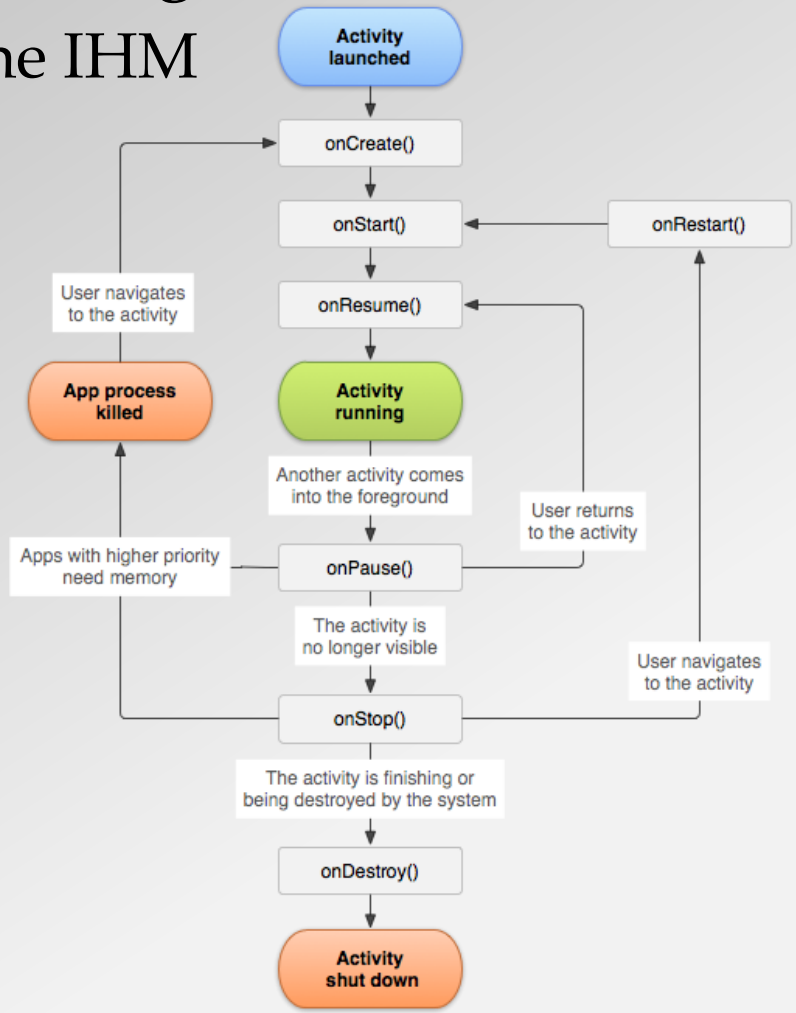
Les composants

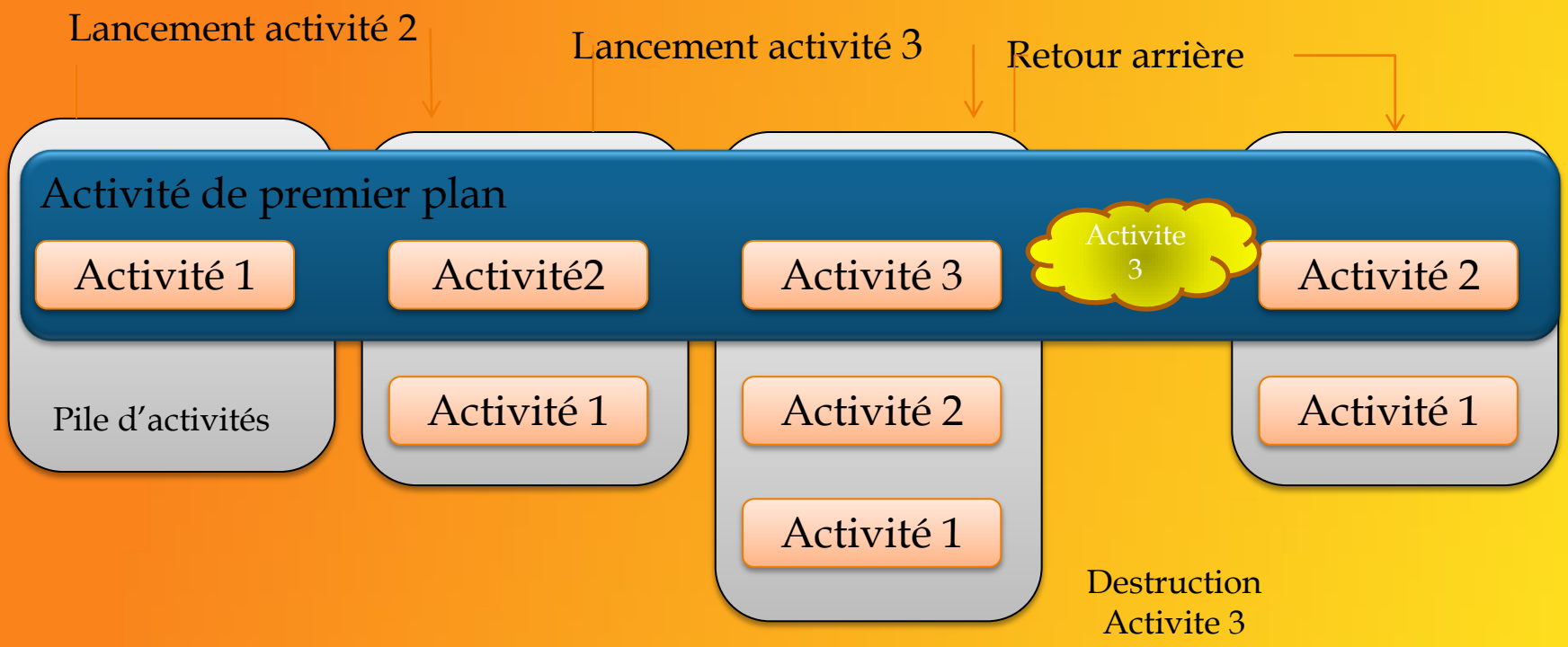
- 4 types
  - Activité
  - Service
  - Broadcast receiver
  - Content Provider

Used for	Component	Example
Focused thing a user can do	Activity	Edit a note, play a game
Background process	Service	Play music, update weather icon
Receive messages	Broadcast Receiver	Trigger alarm upon event
Store and retrieve data	Content Provider	Open a phone contact

Activités

- Quelque chose qui inter-agit avec l'utilisateur
  - Offre généralement une IHM
- Etats d'une activié
  - Running
  - Paused
  - Stopped





Activités

- Permissions
- Cycle de vie
  - Activité en premier plan
  - Activité visible
  - Activité en tache de fond
  - Activité vide

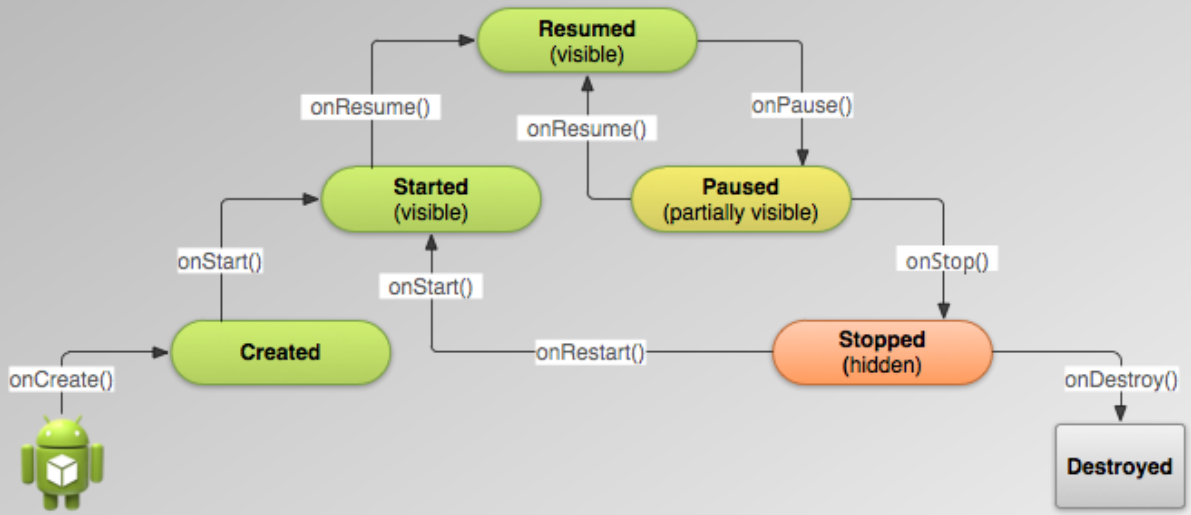
Ordre des priorités de destruction des activités

Activités

- Changement de configurations
  - L'activité est détruite et redémarrée
    - Ce comportement est configurable. android:configChanges du manifeste, onConfigurationChanged (Configuration) callback
- Lancement d'une activité
  - startActivity(intent);
  - startActivityForResult(intent, int);
    - Résultat lors de l'événement onActivityResult(int, int, Intent) callback. Un seul callback. Nécessite de tester l'origine de l'intent.
- Fin d'une activité On exit
  - L'activité parente peut être prévenue de la fin de l'activité et recevoir un statut
    - setResult(int)
    - L'activité parente gère l'événement onActivityResult() callback

Activités

- Créer une activité



  - Subclass android.app.activity
  - Implementer les callbacks de l'activité
    - onCreate()
      - Start/initialize UI
        - Inflate layout view
    - onPause()
      - Save activity state

Activite

- Ex de code source
  - **public class** MainActivity **extends** Activity {

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    Intent it = new Intent();
    it.setAction(Intent.ACTION_DIAL);
    startActivity(it);
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present.
    getMenuInflater().inflate(R.menu.menu_main, menu);
    return true;
}
```

Warning

- Lors du développement pensez toujours à configurer le fichier manifest

Intents

- Objet utilisé pour demander le lancement de l'exécution d'un composant de l'application. L'objet est aussi un conteneur de messages. Permet le passage de données entre composants.
  - Pour lancer une activity / application
    - startActivity(Intent). //Par défaut pas de résultat en retour
  - Pour lancer un service
    - startService(Intent)
  - Pour se lier à un service
    - onBind(Intent)
  - Pour diffuser un broadcast
    - sendBroadcast(Intent);
  - Pour vehiculer des données key/value
    - putExtra, get Extra
- Lorsqu'elle se termine, une activité peut retourner un résultat
  - startActivityForResult(Intent)
    - Un intent est renvoyé sur l'événement onActivityResult() callback

Intent

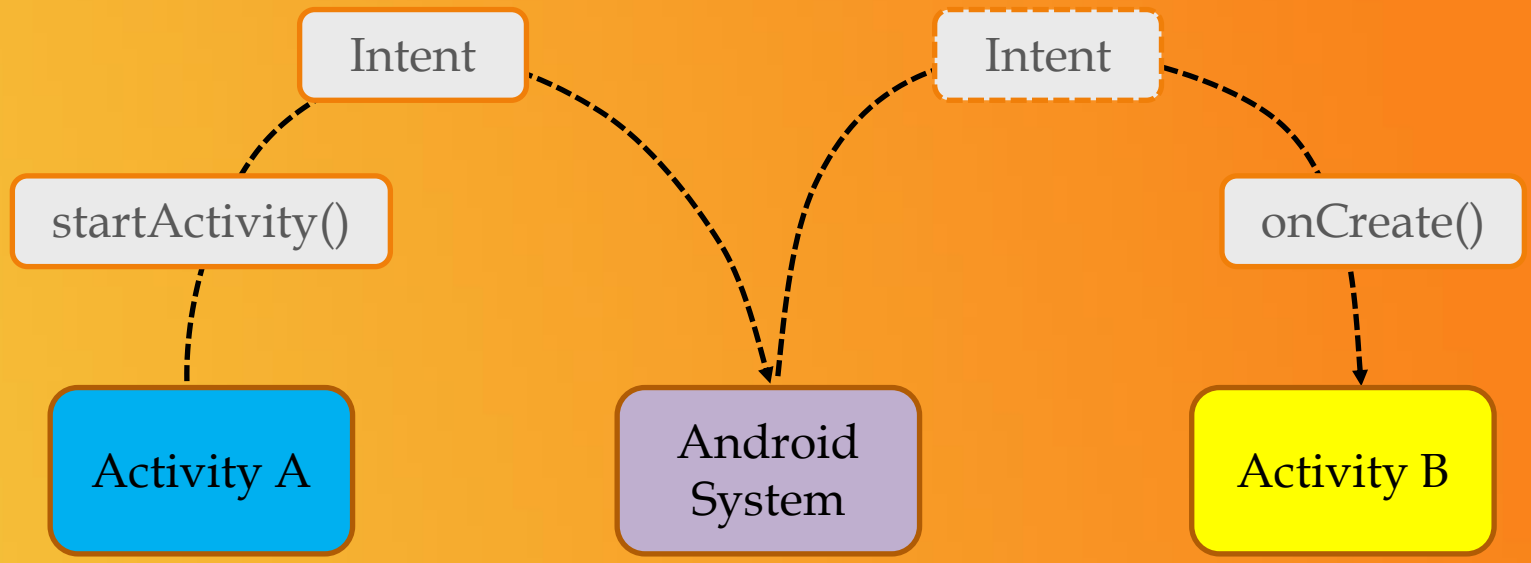
- Création d'un intent
  - 2 types : Explicit et Implicit
    - Permet Couplage fort ou faible entre composants
    - Avantages / inconvénients
  - Intent explicit
    - Component name
      - className du composant à exécuter / lancer
  - Intent implicit : on associe à l'intent des critères qui permettront de sélectionner une activité/service
    - Action
      - String précisant l'action demadée
        - ACTION_VIEW, ACTION_SEND, ACTION_DIAL
    - Category
      - String précisant la catégorie de l'activité permettant de traiter l'intent
        - CATEGORY_LAUNCHER
  - L'intent peut contenir des données
    - Data
      - URI des données à traiter
    - Extras
      - Key/Value paires contenant l'information à traiter
      - L'information est enregistrée par putExtra(), putExtras()
    - Flags
      - MetaData sur l'intent . setFlags();

Intent

- Exemples
- Explicit intent
  - // The fileUrl is a string URL, such as "http://www.example.com/image.png"
    Intent downloadIntent = new Intent(this, DownloadService.class);
    downloadIntent.setData(Uri.parse(fileUrl));
    startService(downloadIntent);
- Implicit Intent
  - // Create the text message with a string
    Intent sendIntent = new Intent();
    sendIntent.setAction(Intent.ACTION_SEND);
    sendIntent.putExtra(Intent.EXTRA_TEXT, textMessage);
    sendIntent.setType(HTTP.PLAIN_TEXT_TYPE); // "text/plain" MIME type

    // Verify that the intent will resolve to an activity
    if (sendIntent.resolveActivity(getPackageManager()) != null) {
    startActivity(sendIntent);
    }

Intent



Intent

- Chooser
  - Dans le cas où plusieurs activités répondent aux critères de l'intent, le choix de l'activité est proposé à l'utilisateur

```
String title = getResources().getString(R.string.chooser_title);
Intent sendIntent = new Intent(Intent.ACTION_SEND);
// Cree un  intent qui affiche un chooser dialog
Intent chooser = Intent.createChooser(sendIntent, title);
// Verifie que l'intent sera résolu
if (sendIntent.resolveActivity(getPackageManager()) != null) {
startActivity(chooser);
}
```

  - Il est conseillé de valider la résolution d'un intent avant de lancer un intent

Intent

- Filtre d'intentions
  - Configuration du manifeste pour préciser les critères de reception d'intent implicit par l'application
  - Le Manifest declare quels « intent » l'application peut gérer
    - <intent-filter>
    - <action>
    - <category>
    - <data>
    - </intent-filter>

Intent

- Action
  - Le type d'action sollicitée
    - i.e Location, send, record
- Data
  - L'URI des données, et /ou le type Mime
- Category
  - Le type de composant pouvant supporter la demande
    - i.e CATEGORY_BROWSABLE, CATEGORY_LAUNCHER

Intent Filter

exemple

```
<activity android:name="MainActivity">
    <!-- This activity is the main entry, should appear in app launcher -->
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>

<activity android:name="ShareActivity">
    <!-- This activity handles "SEND" actions with text data -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="text/plain"/>
 <!-- This activity also handles "SEND" and "SEND_MULTIPLE" with media data -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <action android:name="android.intent.action.SEND_MULTIPLE"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="application/vnd.google.panorama360+jpg"/>
        <data android:mimeType="image/*"/>
        <data android:mimeType="video/*"/>
    </intent-filter>
</activity>   </intent-filter>
```

Android

- Liste des actions natives
- ACTION_MAIN
- ACTION_VIEW
- ACTION_ATTACH_DATA
- ACTION_EDIT
- ACTION_PICK
- ACTION_CHOOSER
- ACTION_GET_CONTENT
- ACTION_DIAL
- ACTION_CALL
- ACTION_SEND
- ACTION_SENDTO
- ACTION_ANSWER
- ACTION_INSERT
- ACTION_DELETE
- ACTION_RUN
- ACTION_SYNC
- ACTION_PICK_ACTIVITY
- ACTION_SEARCH
- ACTION_WEB_SEARCH
- ACTION_FACTORY_TEST

Intents

- Résolution d'intention
  - Test action
    - Un filtre peut contenir 0, 1 ou plusieurs actions
      - <intent-filter>
        <action android:name="android.intent.action.EDIT" />
        <action android:name="android.intent.action.VIEW" />
        ...
        </intent-filter>
    - Test
      - passé si 0 ou une action est résolue
      - Echec si le filtre ne contient aucune action
  - Test category //attention par défaut un intent implicite est de category_defaut
    - Un filtre peut contenir 0,1 ou plusieurs category
    - Test
      - Passé si toutes les category de l'intent sont résolues par le filtre ('inverse est faux)
      - Passé si l'intent ne contient aucune category et que votre filtre d'activité contient category_default
      - Echec sinon

Intents

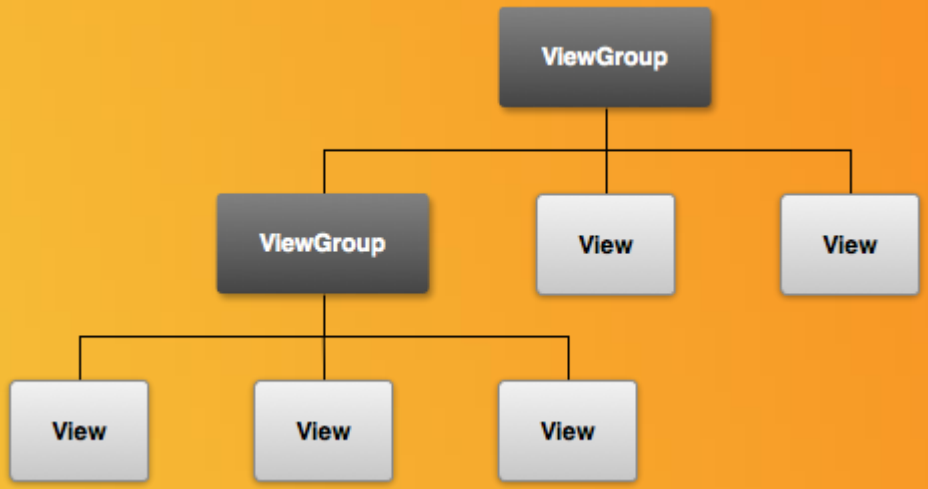
- Résolution d'intention
  - Test data
    - Un filtre peut contenir 0,1 ou plusiers data
    - <intent-filter>
      <data android:mimeType="video/mpeg" android:scheme="http" ... />
      <data android:mimeType="audio/mpeg" android:scheme="http" ... />
      ...
      </intent-filter>
      - <scheme>://<host>:<port>/<path>
- Sécurité
  - setExport(false)

Pending Intent

- Un wrapper qui encapsule un intent.
  - L'intent sera exécuté même si l'activité qui a crée le pending intent a été détruite, annulée.
  - Souvent associé à des activités de type broadcastReceiver
    - Ex
      - int seconds = 2;

```
// Create an intent that will be wrapped in PendingIntent
Intent intent = new Intent(this, MyReceiver.class)
// Create the pending intent and wrap our intent
PendingIntent pendingIntent = PendingIntent.getBroadcast(this, 1, intent, 0);
// Get the alarm manager service and schedule it to go off after 3s
AlarmManager alarmManager =
   (AlarmManager)getSystemService(ALARM_SERVICE);
alarmManager.set(AlarmManager.RTC_WAKEUP,
   System.currentTimeMillis() + (seconds * 1000), pendingIntent);
Toast.makeText(this, "Alarm set in " + seconds + " seconds",
   Toast.LENGTH_LONG).show();
```

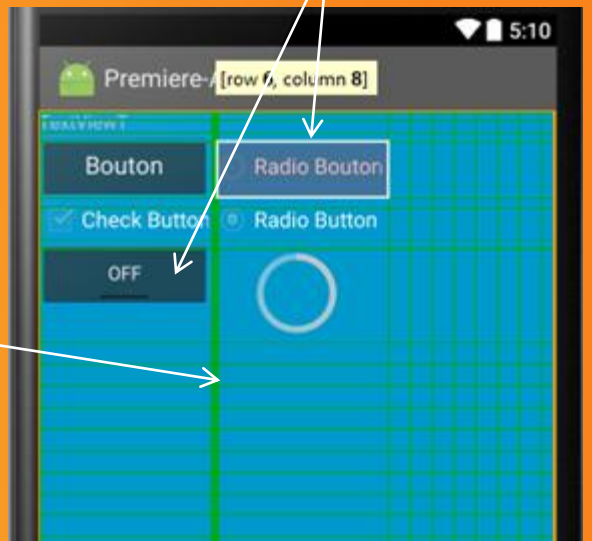
Les IHM



Les vues sont placées dans des conteneurs

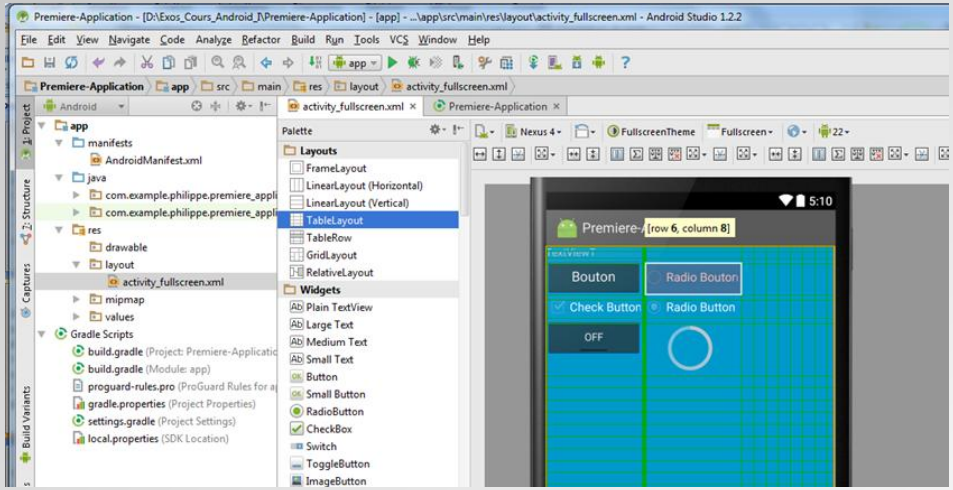
Views:

ViewGroup
Layouts :



Widget - Views

- Widgets / Views pour créer des IHM



- Les principaux
  - TextView
    - Size: lines, pixels, ems,
    - autoLink: web|email
  - EditText
    - inputFilter
      - setFilters
    - Autocompletion
  - Spinner
  - Button
  - Toggle Button
  - CheckBox
  - Switch
  - RadioButton
  - RadioGroup
  - DatePicker
  - TimePicker

Widgets - Views

- ProgressBar
- SeekBar
- RatingBar
- Chronometer
- DigitalClock
- TextClock
- AnalogClock

Les événements UI

- Capter les événements des Views
  - Les interfaces– event listeners
    - Declarentsles callback
    - Un seul callback
  - Les event listener s'enregistrent auprès des views
    - onClick()
      - From
      - View.OnClickListener
    - onLongClick()
      - View.OnLongClickListener.
    - onFocusChange()
      - View.OnFocusChangeListener.
    - onKey()
      - View.OnKeyListener.
    - onTouch()
      - View.OnTouchListener.
    - onCreateContextMenu()
      - View.OnCreateContextMenuListener.

UI Events

- Gérer les événements provenant des Views
  - Implementer une interface dans votre activité
  - Gérer localement les événements en implémentant une interface anonyme
  - public class ExampleActivity extends Activity implements OnClickListener {

```
protected void onCreate(Bundle savedValues) {
 ...
Button button = (Button)findViewById(R.id.corky);
button.setOnClickListener(this);
}.
public void onClick(View v) {
    // do something when the button is clicked
  }
// Create an anonymous implementation of OnClickListener
private OnClickListener mCorkyListener = new OnClickListener() {
  public void onClick(View v) {
…
  }
};
```

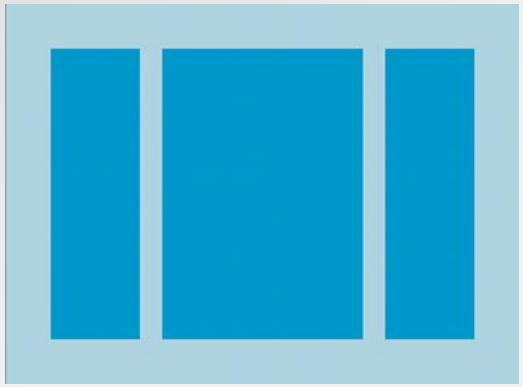
Conteneurs -UI Layouts

- Layout : topologie de l'affichage des vues à l'écran
- Défini comme ressource XML (le + simple) ou par programmation
- <?xml version="1.0" encoding="utf-8"?>

```
<LinearLayout
xmlns:android=http://schemas.android.com/apk/res/android
 android:layout_width="fill_parent«
android:layout_height="fill_parent«
android:orientation="vertical" >
<TextView android:id="@+id/text«
android:layout_width="wrap_content«
 android:layout_height="wrap_content«
android:text="I am a TextView" />
<Button android:id="@+id/button«
android:layout_width="wrap_content«
 android:layout_height="wrap_content«
android:text="I am a Button" />
</LinearLayout>
```

Conteneurs - UI Layouts

- Linear Layout
  - Les vues enfants s'inscrivent dans un même ordre V/H
  - Vertical – horizontal
    - Des propriétés pour affiner le psotionnement des vues enfants
    - Android:layout_weight
    - Android:layout_gravity



UI Layout

- Gravity

top	Push object to the top of its container, not changing its size.
Bottom	Push object to the bottom of its container, not changing its size.
Left	Push object to the left of its container, not changing its size.
Right	Push object to the right of its container, not changing its size.
center_vertical	Place object in the vertical center of its container, not changing its size.
fill_vertical	Grow the vertical size of the object if needed so it completely fills its container.
center_horizontal	Place object in the horizontal center of its container, not changing its size.
fill_horizontal	Grow the horizontal size of the object if needed so it completely fills its container.
Center	Place the object in the center of its container in both the vertical and horizontal axis, not changing its size.
fill	Grow the horizontal and vertical size of the object if needed so it completely fills its container.
clip_vertical	Additional option that can be set to have the top and/or bottom edges of the child clipped to its container's bounds. The clip will be based on the vertical gravity: a top gravity will clip the bottom edge, a bottom gravity will clip the top edge, and neither will clip both edges.
clip_horizontal	Additional option that can be set to have the left and/or right edges of the child clipped to its container's bounds. The clip will be based on the horizontal gravity: a left gravity will clip the right edge, a right gravity will clip the left edge, and neither will clip both edges.
start	Push object to the beginning of its container, not changing its size.
end	Push object to the end of its container, not changing its size.

UI Layouts

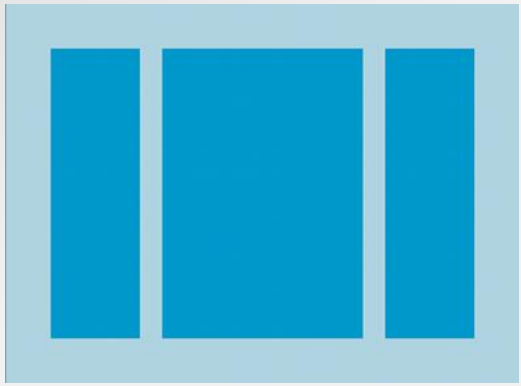
- Relative Layout
  - Les vues enfants se positionnent par rapport aux vues voisines ou parentes
    - Ex attributs relatifs aux vues parentes
      - android:layout_alignParentTop ="true", android:layout_centerVertical="true"
    - Ex attibuts relatifs aux vues voisines
      - android:layout_below=@id/view1
      - android:layout_toRightOf=@id/list1

Conteneurs- UI Layouts

- Grid Layout
  - Les vues enfants sont placées sur une grille rectangulaire (cells)
  - Possibilité de recouvrement de cells (span)
  - Attributs définissant le positionnement :
    - rowspec, columnspec
    - Espacement défini le margin et le space view
    - Etirement des views grâce aux attributs gravity et weight

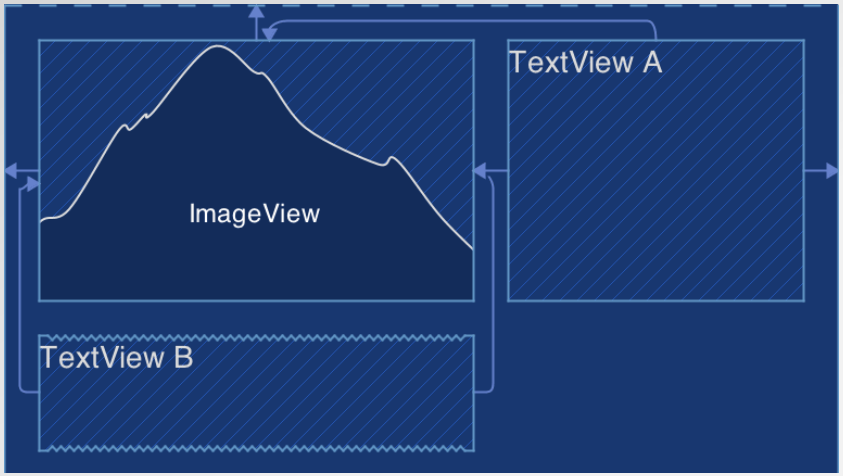
Conteneurs - UI Layouts

- CoordinatorLayout
  - FrameLayout évolué
  - Conteneur permettant des interactions avec les vues enfants
  - Les vues enfants définissent : app:layout_behavior
  - Redéfinir layoutDependsOn()



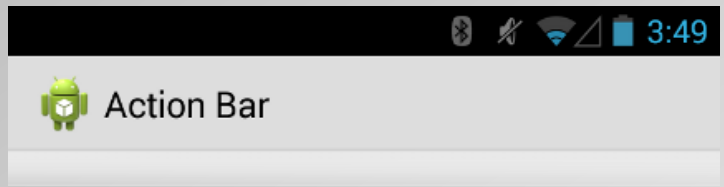
Conteneurs - UI Layouts

- Constraintayout
  - Positionnement relatif des vues enfants + contraintes
    - Axe Horizontal : Left, Right, Start and End sides
    - AxeVertical : top, bottom sides and text baseline

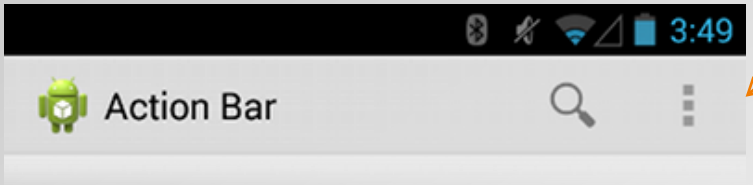


IHM- Barre d'actions

- Navigation
  - Barre d'actions



    - Apparait avec l'API 11 (Version 3.0) min
    - Thème Holo
  - Ajout de boutons



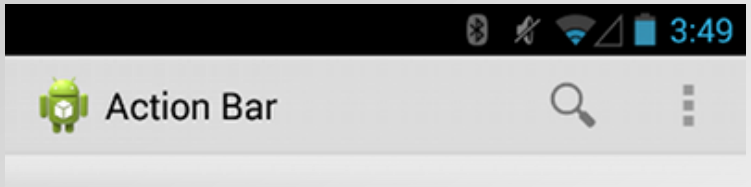
OverFlow menu

  - Menu item
    - Défini en xml dans /res
    - <menu xmlns:android="http://schemas.android.com/apk/res/android" >
      <!-- Search, should appear as action button -->
      <
    - "
      android:icon="@drawable/ic_action_search"
      android:title="@string/action_search"
      android:showAsAction="ifRoom" />
      <!-- Settings, should always be in the overflow -->
      <item android:id="@+id/action_settings"
      android:title="@string/action_settings"
      android:showAsAction="never" />
      </menu>

Propriété: showAsAction : ifRoom, Never

IHM- Barre d'actions

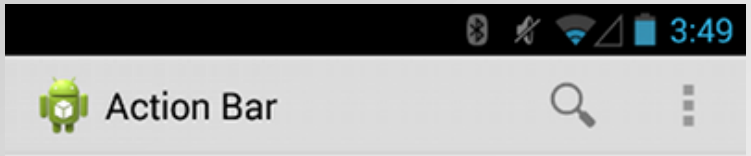
- Navigation
  - En mode compatibilité utiliser ToolBar



  - ActionBar split est déprécié en version 25
  - Toolbar dispose d'un bottom
  - UP action
    - Utilisé pour naviguer vers l'activité parente
    - Activation par code
    - Nécessite modification dans le manifeste

IHM- Barre d'actions

- ActionView
  - Prise en charge d'une action complexe ie search, fragment

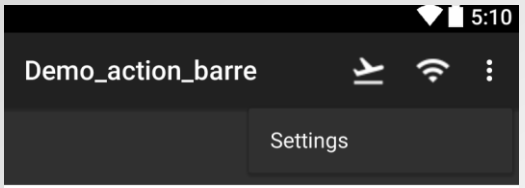


- Action provider
  - Prise en charge d'une action par appel d'un nouveau layout ie nouveau menu

IHM - Barre d'actions

- Gestion des événements issus des boutons
  - Utiliser le callback onOptionsItemSelected de l'activité

```
@Override
public booleanonOptionsItemSelected(MenuItem item) {
    // déclenché sur la sélection d'un menu item
    switch (item.getItemId()) {
        case R.id.action_flight_take_off:
            openListFlightsDeparture;
            return true;
             case R.id.action_Wifi:
            setWifiOn();
            return true;
        case R.id.action_settings:
            openSettings();
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}
```

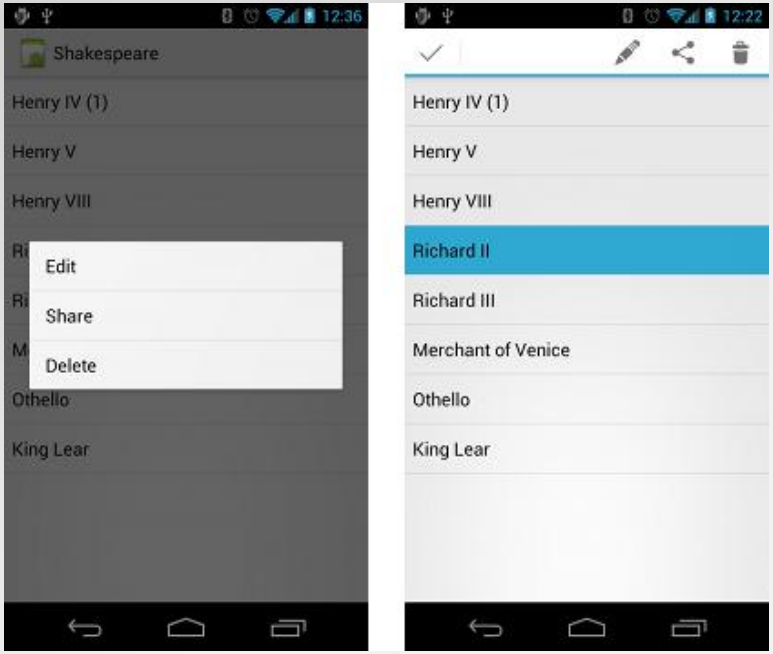


IHM - Barre d'actions

- Menus dynamiques
  - API >=3.0
    - utiliser le callbackc onPrepareOptionsMenu().
    - Le callback est déclenché par appel à invalidateOptionsMenu()
    - Utiliser ActionView
    - Utiliser ActionProvider

IHM – Menu Contextuel

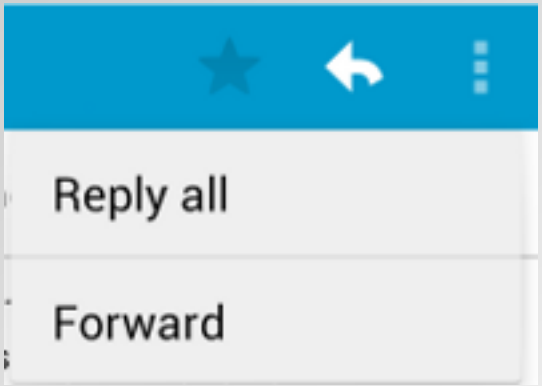
- Associé à des vues pour agir sur des propriétés spécifiques
  - Ex listView et modification d'items
- Menu contextuel flottant (API <11)
- Menu contextuel avec barre d'action (Recommandé API >11)



- Click long sur une vue
  - StartActionMode()
  - Implémenter l'interface
    - Gère les actions du menu conxtextuel

IHM – PopUpMenu

- Menu attaché à une vue
  - Api >11
  -



IHM – Intent Menuitem

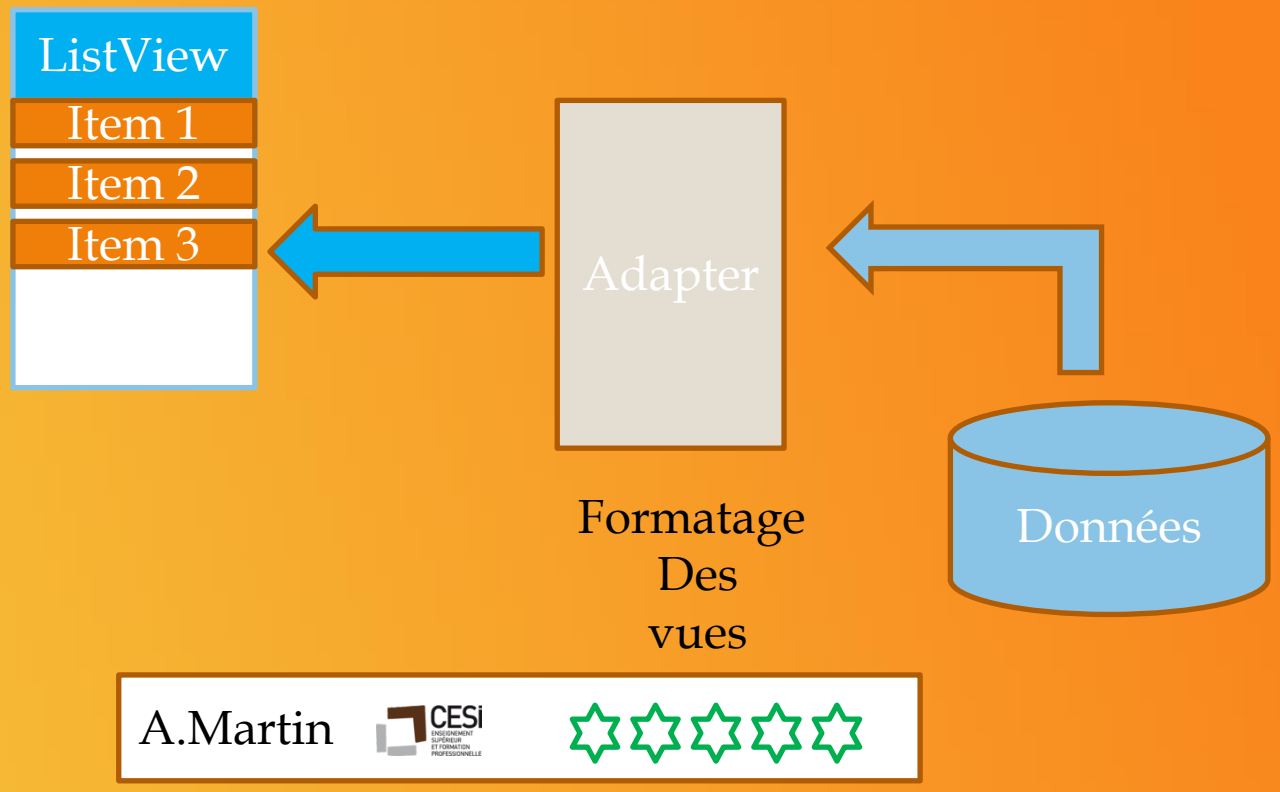
- Pop Up
  -

Dessin sur une vue

- Dessiner des formes aléatoires

Adapters

Permettent le contrôle de l'affichage des données depuis une source



Adapters

- Affichage de données avec un adapter
  - Base adapter, ListAdapter, arrayAdapter …
  - Implementent Adapter Interface
  - arrayAdapter
    - ArrayAdapter<String> adapter = new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1, myStringArray);
    - ListView listView = (ListView) findViewById(R.id.listview); listView.setAdapter(adapter);
    - For customization override
      - toString() and getView()
- SimpleCursorAdapter
  - Permet l'affichage de données de requêtes retournant un cursor
  - On spécifie l e layout pour chaque ligne retournée
  - on spécifie l'association colonne et vue du layout

Adapters

- SimpleCursorAdapter
  - Filling with data
  - String[] fromColumns = {ContactsContract.Data.DISPLAY_NAME,
    ContactsContract.CommonDataKinds.Phone.NUMBER};
    int[] toViews = {R.id.display_name, R.id.phone_number};
  - SimpleCursorAdapter adapter = new SimpleCursorAdapter(this,
    R.layout.person_name_and_number, cursor, fromColumns, toViews, 0);
    ListView listView = getListView();
    listView.setAdapter(adapter);
  - notifyDataSetChanged()
- Handling click events
  - You implement AdapterView.OnItemClickListener interface.

```
// Create a message handling object as an anonymous class.

private OnItemClickListener mMessageClickedHandler = new
OnItemClickListener() {
    public void onItemClick(AdapterView parent, View v, int position, long id) {
        // Do something in response to the click
    }
};
listView.setOnItemClickListener(mMessageClickedHandler);
```

Animation

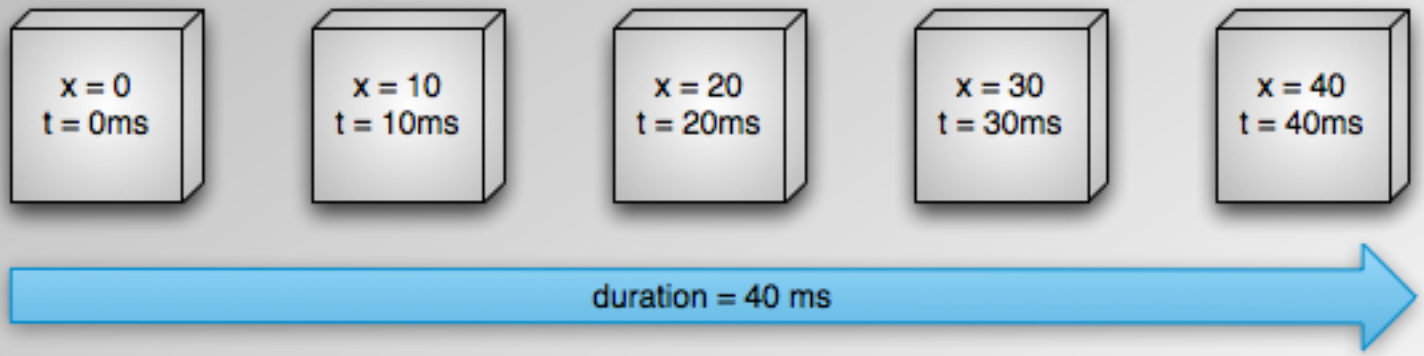
- Que peut-on animer ?
  - Properties *recommandé
  - Views
  - Drawables
- Animation de propriétés
  - Durée
  - Temps d' Interpolation
  - Répetition
  - Ensembles d'animators
  - Fréquence de refresh 10 ms par défaut

Animation de vue

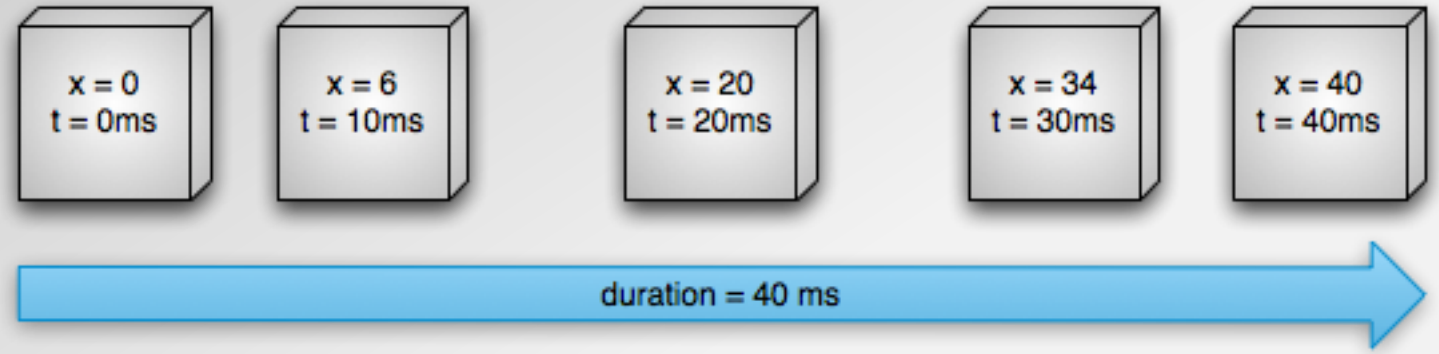
- Agit sur
  - Position
  - Taille/Echelle
  - Rotation
  - Transparence
- Les actions se définissent par programme ou dans une ressources re/anim

Animation

- Principe
  - Animation de propriété
  - Linéaire

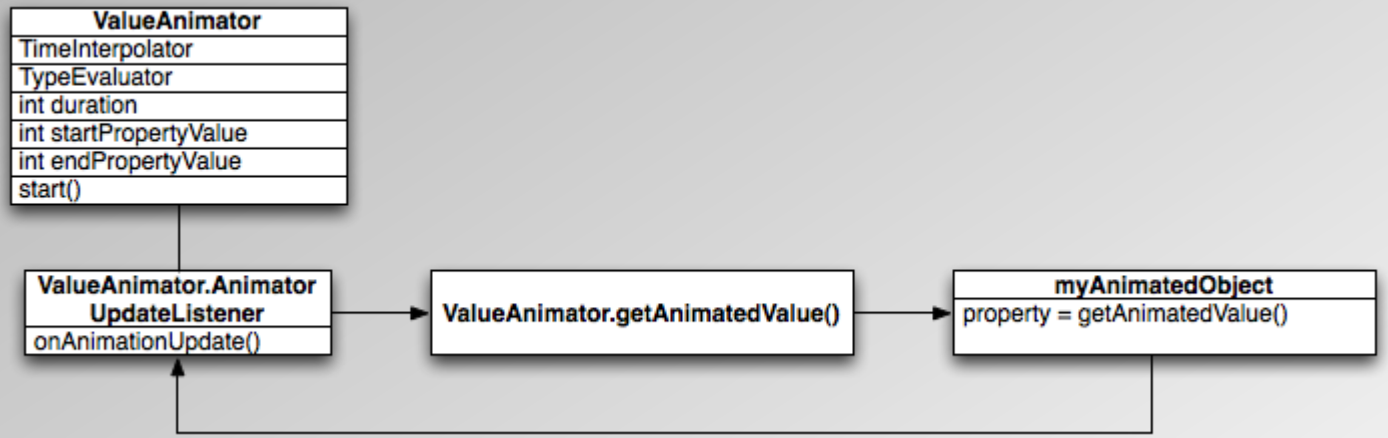


  - Non Linéaire



Animation

- Animation de propriété



- Mise en oeuvre
  - Instancier un ValueAnimator ou un ObjectAnimator
  - Fournir les valeurs début-fin
  - Start()
    - Le temps écoulé s'exprime en fraction %
    - Définir le type Evaluator
      - IntEvaluator, FloatEvaluator, ArgbEvaluator, TypeEvaluator
    - Choisir un type d'interpolator
    - AccelerateDecelerateInterpolator, BounceInterpolator…

Animation

- Ex
- Animation de propriétés
  - ValueAnimator animation = ValueAnimator.ofFloat(0f, 1f);

    animation.setDuration(1000);

    animation.start();

    //custom animation avec MyTypeEvaluator
  - ValueAnimator animation = ValueAnimator.ofObject(new MyTypeEvaluator(), startPropertyValue, endPropertyValue);

    animation.setDuration(1000);

    animation.start();
  - Ajouter un AnimatorListener
    - Utiliser onAnimationValue() call back pour animer la View
    - Pour rafraichir l'affichage du conteneur de la View : appeler invalidate()

Animation

- Animation de propriétés
  - Utiliser un objectAnimator
  - Spécifier la classe et les propriétés à animer
    - ObjectAnimator anim = ObjectAnimator.ofFloat(la_propriete, "alpha", 0f, 1f);
      anim.setDuration(1000);
      anim.start();
    - La proprieté doit avoir un setter
    - Necessite un getter si l'appel au constructeur ne précise pas la valeur de début
  - Jouer des séquences d'animations
    - Subclass AnimatorSet

Animation

- Animation de propriétés
  - Callbacks
    - Animator.AnimatorListener
    - onAnimationStart() - Called when the animation starts.
    - onAnimationEnd() - Called when the animation ends.
    - onAnimationRepeat() - Called when the animation repeats itself.
    - onAnimationCancel() - Called when the animation is canceled. A cancelled animation also calls onAnimationEnd(), regardless of how they were ended.
    - ValueAnimator.AnimatorUpdateListener

Animation

- Animation de propriétés
  - Animate layouts change to ViewGroups
    - APPEARING
    - CHANGE_APPEARING
    - DISAPPEARING
    - CHANGE_DISAPPEARING
  - Create LayoutTransition instance
  - ViewGroup.setLayoutTransition(LayoutTransition)
- Custom evaluator
  - Implement the TypeEvaluator Interface
- Custom interpolator
  - Implement TimeInterpolator interface
- KeyFrame
  - Defines a (fraction key, value) pair
  - Keyframe kf0 = Keyframe.ofFloat(0f, 0f);
    Keyframe kf1 = Keyframe.ofFloat(.5f, 360f);
    Keyframe kf2 = Keyframe.ofFloat(1f, 0f);
    PropertyValuesHolder pvhRotation =
    PropertyValuesHolder.ofKeyframe("rotation", kf0, kf1, kf2);
    ObjectAnimator rotationAnim =
    ObjectAnimator.ofPropertyValuesHolder(target, pvhRotation)
    rotationAnim.setDuration(5000ms);

Animation

- Animation de propriétés
  - S'applique aussi à des vues
    - translationX , translationY:
    - rotation, rotationX, rotationY
    - scaleX , scaleY
    - pivotX , pivotY
    - x and y:
    - alpha
  - ObjectAnimator.ofFloat(myView, "rotation", 0f, 360f);

Animation

- View animation
- Drawable animation
- Draw on Canvas

Animation

- Exemple views fading

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
            android:id="@+id/content"
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView style="?android:textAppearanceMedium"
                android:lineSpacingMultiplier="1.2"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="@string/lorem_ipsum"
                android:padding="16dp" />

        </ScrollView>

        <ProgressBar android:id="@+id/loading_spinner"
            style="?android:progressBarStyleLarge"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center" />

    </FrameLayout>
```

Animation

- Exemple views fading

```
public class CrossfadeActivity extends Activity {

        private View mContentView;
        private View mLoadingView;
        private int mShortAnimationDuration;

        ...

        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_crossfade);

            mContentView = findViewById(R.id.content);
            mLoadingView = findViewById(R.id.loading_spinner);

            // Initially hide the content view.
            mContentView.setVisibility(View.GONE);

            // Retrieve and cache the system's default "short" animation time.
            mShortAnimationDuration = getResources().getInteger(
                    android.R.integer.config_shortAnimTime);
        }
```

Animation

- Exemple views fading

```
private View mContentView;
        private View mLoadingView;
        private int mShortAnimationDuration;

        ...

        private void crossfade() {

            // Set the content view to 0% opacity but visible, so that it is visible
            // (but fully transparent) during the animation.
            mContentView.setAlpha(0f);
            mContentView.setVisibility(View.VISIBLE);

            // Animate the content view to 100% opacity, and clear any animation
            // listener set on the view.
            mContentView.animate()
                    .alpha(1f)
                    .setDuration(mShortAnimationDuration)
                    .setListener(null);

            // Animate the loading view to 0% opacity. After the animation ends,
            // set its visibility to GONE as an optimization step (it won't
            // participate in layout passes, etc.)
            mLoadingView.animate()
                    .alpha(0f)
                    .setDuration(mShortAnimationDuration)
                    .setListener(new AnimatorListenerAdapter() {
                        @Override
                        public void onAnimationEnd(Animator animation) {
                            mLoadingView.setVisibility(View.GONE);
                        }
                    });
        }
```

Boites de dialogues

- **Dialog**: the basic class for all Dialog types. A basic Dialog (android.app.Dialog)
- **AlertDialog**: a Dialog with one, two, or three Button controls. An AlertDialog (android.app.AlertDialog)
- **CharacterPickerDialog**: a Dialog for choosing an accented character associated with a base character.
- **DatePickerDialog**: a Dialog with a DatePicker control. (android.app.DatePickerDialog)
- **ProgressDialog**: a Dialog with a determinate or indeterminate ProgressBar (android.app.ProgressDialog)
- **TimePickerDialog**: a Dialog with a TimePicker control. A TimePickerDialog (android.app.TimePickerDialog)
- **Presentation**: Presentation dialog is a type of Dialog used for presenting content to a secondary display. (android.app.Presentation)

Dialogs

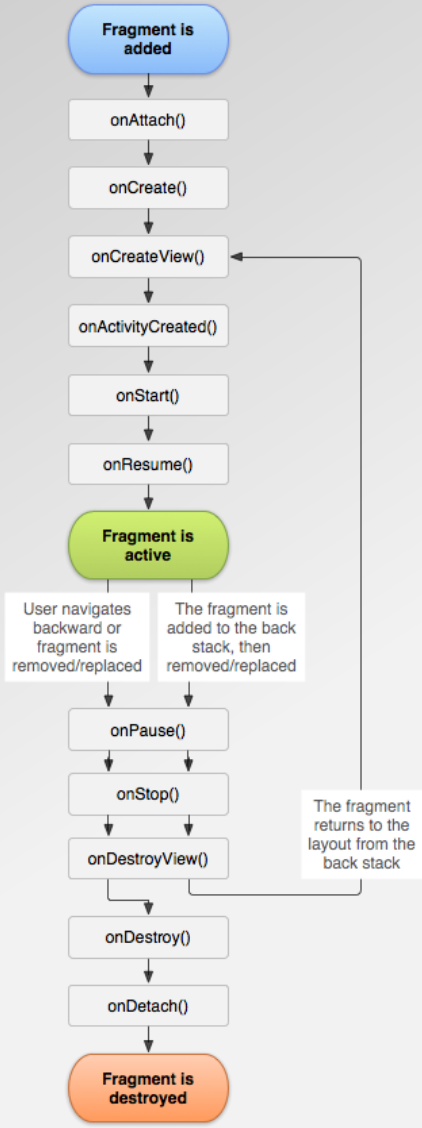
- **Use DialogFragment**:
  - The show() method is used to display the Dialog.
  - The dismiss() method is used to stop showing the Dialog.

Fragments

- Fragments
  - Un fragment encapsule le comportement d'une partie d'un UI
  - Re-utilisable
  - Constitue un sous-ensemble permettant de réaliser des UI multi-vues
  - Répond au besoin de gérer des UI dynamiques sur des écrans de tailles variées

Fragments

- Un fragment n'existe qu'au sein d'une activité, son cycle de vie est intrinsèquement lié à celui de l'activité parente
- Cycle de vie
  - Callbacks
    - onCreate()
      - Place d'initialisation des composants
      - du fragment
    - onCreateView()
      - Appelé au moment de la création de l'UI du fragment
    - onPause()
      - Appelé lorsque l'utilisateur quitte
      - le fragment



Fragments

- Créer un fragment
  - Dériver une classe de android.app.Fragment
  - Créer un layout
  - Initialiser l' UI
    - Return a View as part of onCreateView() callback
      - Inflate a layout resource
  - Ajouter le fragment à une activity
  - Gérer le fragment dans une activité

Fragment

- Autres classes dérivées
- DialogFragment
  - Affiche une boîte de dialogue
- ListFragment
  - Affiche une liste d'items qui sont gérés par un adapter (SimpleCursorAdapter), équivalent à ListActivity.
- PreferenceFragment
  - Affiche une hiérarchie de Preferences sous forme de liste, similaire à PreferenceActivity.

Fragment

- Details
  - Création de classe dérivée
    - Public class MyFragment extends Fragment{…}
  - Initialisation de la vue
    - public static class ExampleFragment extends Fragment {

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup
container,Bundle savedInstanceState) {
// Inflate the layout for this fragment
return inflater.inflate(R.layout.example_fragment, container, false);
}
}
```

      - Params:
        - Resource id du fragment
        - container parent du fragment , de type viewGroup
        - False : ne pas attacher le fragment au view group parent

Fragment

- Ajout à une activity
  - Declaration du fragment dans le layout de l'activité

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android=http://schemas.android.com/apk/res/android
android:orientation="horizontal«
 android:layout_width="match_parent«
 android:layout_height="match_parent">
 <fragment android:name="com.example.news.ArticleListFragment«
android:id="@+id/list«
android:layout_weight="1«
android:layout_width="0dp«
 android:layout_height="match_parent" />
<fragment android:name="com.example.news.ArticleReaderFragment«
android:id="@+id/viewer«
android:layout_weight="2«
 android:layout_width="0dp«
android:layout_height="match_parent" />
</LinearLayout>
```

android:name spécifie la class fragment à instancier
android:tag optionnel

Fragment

- Fragment : reservation d'un espace du layout qui recevra un fragment
- Ex de layout

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingBottom="@dimen/activity_vertical_margin"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
   android:id="@+id/layoutparent"
   tools:context=".MainActivity" >

   <Button
      android:id="@+id/button1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_alignParentLeft="true"
      android:layout_alignParentTop="true"
      android:layout_marginLeft="24dp"
      android:layout_marginTop="74dp"
      android:text="Bouton OK" />

   <FrameLayout
     android:layout_width="wrap_content"
      android:layout_height="match_parent"
      android:id="@+id/ui_container"
      android:layout_weight="3"
      />

</RelativeLayout>
```

Fragment

- Ajout à une activity
  - Par programme ajout à un viewGroup
    - Obtenir un FragmentTransaction
      - FragmentManager fragmentManager=getFragmentManager()
      - FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();
    - Puis add et commit
      - ExampleFragment fragment = new ExampleFragment();
      - fragmentTransaction.add(R.id.fragment_container, fragment);
      - fragmentTransaction.commit();

Fragments

- Gestion des fragments dans une activity
  - Utiliser le FragmentManager
    - findFragmentById()
    - findFragmentByTag()
    - Pop fragments off back stack : popBackStack()
    - Register a listener for changes to the back stack: addOnBackStackChangedListener().
  - Add, remove, replace fragments, running fragment transactions
    - // Crée un nouveau fragment et transaction
      Fragment newFragment = new ExampleFragment();
      FragmentTransaction transaction = getFragmentManager().beginTransaction();
      // Remplace la view d'un container de fragment par un nouveau fragment,
      // anajout de la transaction au back stack
      transaction.replace(R.id.fragment_container, newFragment);
      transaction.addToBackStack(null);
      // Commit the transaction
      transaction.commit();

      setTransition() allows transition animation during changes

Fragments

- Gestion des événements en liaison avec l'activité
  - Créer une interface interface dans la classe fragment
  - L' activityparente implemente l'interface
  - Sur le callback onAttach fragment obtenir la référence à l'interface de l'activité
  - Sur un événement le Fragment déclenche le callback on event
  - L'Activity execute le callback
- Adding items to actionBar

Fragment

- Fragment
- Code appel du fragment
- **public class MainActivity extends FragmentActivity {**

- @Override
- **protected void onCreate(Bundle savedInstanceState) {**
- **super.onCreate(savedInstanceState);**
- setContentView(R.layout.*activity_main);*

- FragmentManager fragmentmanager =getFragmentManager();

- android.app.FragmentTransaction fragmentTransaction = fragmentmanager.beginTransaction();

- Fragment myfragment = **new FragmentSaisie();//fragmentmanager.findFragmentById(R.id.monFragment);**
- fragmentTransaction.attach(myfragment);
- fragmentTransaction.add(R.id.*ui_container, myfragment);*
- fragmentTransaction.commit();

- }

Fragment

- Fragment
  - Sans interface utilisateur
  - Possibilité de garder l'instance du fragment active quand l'activité est interrompue(pour maintenir une tâche de fond par ex): appeler : setRetainInstance
  - Utilisation dans un contexte d'affichage de données master/details

Data Persistence

- Preferences
  - Paires clé/valeur ; key/value
- Fichiers
  - Utilisation des Java files I/O
- Fournisseurs de contenus
  - Relational DB : SQLite

Persistence des Donnees

- Sauvegarde de paires key_value
  - SharedPreferenced API
    - getSharedPreferences(file) from context
    - getPreferences() from Activity
      - Context context = getActivity();

      SharedPreferences sharedPref = context.getSharedPreferences(
      getString(R.string.preference_file_key),Context.MODE_PRIVATE);
      - SharedPreferences sharedPref = getActivity().getPreferences(Context.MODE_PRIVATE);
  - Lecture & Ecriture
    - Obtenir uneditor
      - SharedPreferences sharedPref = getActivity().getPreferences(Context.MODE_PRIVATE);

      SharedPreferences.Editor editor = sharedPref.edit();
    - Puis write /read + commit
      - editor.putInt(getString(R.string.saved_high_score), newHighScore);
      - editor.commit();

Persistence des Donnees

- Lecture /Ecriture
  - SharedPreferences sharedPref = getActivity().getPreferences(Context.MODE_PRIVATE);

    int defaultValue = getResources().getInteger(R.string.saved_high_score_default);

    long highScore = sharedPref.getInt(getString(R.string.saved_high_score), defaultValue);

Persistence des Donnees

- Files I/O; gestion ecriture/lecture de fichiers
  - Utiliser File API
    - Stockage interne vs stockage externe
    - getInternalFilesDir
    - Préciser le stockage dans le manifeste. Android: installLocation within manifest
    - Droit d'écriture à préciser dans le manifeste pour le stockage externe
      - <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    - Droit de lecture à préciser dans le manifeste pour le stockage externe
      - <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
  - Sauvegarde de fichier sur le support interne
    - getFileDir()
    - getCacheDir()
    - Crée un nouvezu fichier dans le répertoire courant
    - File file = new File(context.getFilesDir(), filename);

Persistence des Donnees

- Files I/O
  - Ecriture
    - Obtenir un FileOutputStream
    - Ecriture des données en utilisant un outputStream

```
String filename = "myfile";
String string = "Hello world!";
try {
 outputStream = openFileOutput(filename,
Context.MODE_PRIVATE);
outputStream.write(string.getBytes());
 outputStream.close();
}
 catch (Exception e) {
 e.printStackTrace();
}
```

Data Persistence

- Mise de données en cache (Fichier)
  - public File getTempFile(Context context, String url) {

```
File file;
try {
 String fileName =Uri.parse(url).getLastPathSegment();
 file = File.createTempFile(fileName, null,context.getCacheDir());
catch (IOException e) {
 // Error while creating file
}
 return file;
```

Persistence des Donnees

- Sauvegarde sur support externe : SD
  - Toujours tester si sd est monté
    - /* Verification si le support est monté en read et write */
      public boolean isExternalStorageWritable() {
      String state = Environment.getExternalStorageState();
       if (Environment.MEDIA_MOUNTED.equals(state)) {
      return true;
      }
      return false;
      }
    - /* Vérification si le support SD est accessible en read */
      public boolean isExternalStorageReadable() {
      String state = Environment.getExternalStorageState();
      if (Environment.MEDIA_MOUNTED.equals(state)
      ||Environment.MEDIA_MOUNTED_READ_ONLY.equals(state)) {
      return true;
      }
      return false;
      }

Persistence des Donnees

- Files I/O
  - public File getAlbumStorageDir(Context context, String albumName) {

```
// Get the directory for the app's private pictures directory.
File file = new File(context.getExternalFilesDir(
 Environment.DIRECTORY_PICTURES), albumName);
if (!file.mkdirs()) {
 Log.e(LOG_TAG, "Directory not created");
}
return file;
}
```

  - Passing null, creates dir at root

Persistence des Donnees

- Files I/O
  - Public vs Private files
    - Create file in public dir
      - public File getAlbumStorageDir(String albumName) {

```
// Get the directory for the user's public pictures directory.
File file = new
File(Environment.getExternalStoragePublicDirectory(
 Environment.DIRECTORY_PICTURES), albumName);
 if (!file.mkdirs()) {
 Log.e(LOG_TAG, "Directory not created");
 }
 return file;
}
```

  - Create file in private Dir
    - Acquire the dir to write to using getExternalFilesDir()

Persistence des Donnees

- Query free space
  - getFreeSpace()
  - getTotalSpace()
- Delete file
  - myfile.delete()

Persistence des Donnees

- Utilisation DB ou de fournisseurs de contenu
  - Sqlite proposée en natif. Données privées d'une application.
  - Fournisseur de contenu = mise à disposition d'une interface pour la publication et la consommation des données. Données partagées entre applications.

Persistence des Donnees

- Bases de Données SQLite
  - Open Source
  - Mono client
  - DB privée d'une application
  - /data/data/<package_name>/databases
  - Création/Ouverture SQLiteOpenHelper (Classe abstraite)
  - Enregistrement = ContentValues
  - Resultat requête = Cursor
  - Nombreuses méthodes pour naviguer dans les résultats de requêtes.
  - Consommateur de ressources -> privilégier les actions asynchrones

Persistence des Donnees

- Bases de Données SQLite
  - API : android.database.sqlite
    - SQLiteOpenHelper :
    - SQLiteQueryBuilder :
      - buildQueryString(boolean distinct, String tables, String[] columns, String where, String groupBy, String having, String orderBy, String limit)Build an SQL query string from the given clauses.
    - Executer une query sur un objet db
      - db.execSQL
        - String DATABASE_CREATE = "create table " + TABLE_COMMENTS + "(" + COLUMN_ID + " integer primary key autoincrement, " + COLUMN_COMMENT + " text not null);";
        - database.execSQL(DATABASE_CREATE);
      - db.rawQuery
        - Cursor cursor = getReadableDatabase(). rawQuery("select * from todo where _id = ?", **new** String[] { id });
      - Query
        - database.query(DATABASE_TABLE, **new** String[] { KEY_ROWID, KEY_CATEGORY, KEY_SUMMARY, KEY_DESCRIPTION }, null, null, null, null, null);

Persistence des Donnees

- ContentValue
  - Objet contenant les données qui seront enregistrées en db
  - Utilisé par les content providers
  - Traite des paires key/value
  - Methodes get / put(key/value)
- Cursor
  - Objet résultat des queries
  - Méthodes pour se déplacer dans les enregistrements : moveToFirst(), moveToNext() methods, isAfterLast()
  - Méthodes get*() pour lire le contenu des enregistrements, paires valeurs/index. getLong(columnIndex), getString(columnIndex)

Persistence des Donnees

- Content provider / fournisseur de contenu
  - Interface pour la publication de données qui seront consommées par des résolveurs de contenu
  - Un founisseur de contenu est identifié par un URI
  - Utilisation
    - Obtenir un contenResolver
      - ContentResolver cr = geContentResolver();
    - Ecrire une requête (paramètrable)
      - String[] liste_colonnes = new String[] {provider.Key_ID, provider.Key_Nom, provider.Key_Prenom};
      - String where = provider.Key_Prenom + « = Pierre » ;
      - String wherArgs[] = null; //passage de paramètres
      - String order=null;
    - Executer la requête
    - Cursor resultat = cr.query(provider.CONTENT_URI, colonnes,where, whereArgs,order);
  - Création d'un ContentProvider
    - Etendre ContentProvider
    - Redéfnir les méthodes onCreate, query, update, delete insert, getType
    - Déclarer le fournisseur de contenu dans le manifeste

Persistence des Donnees

- Preference Framework /Settings
  - subclass acctivity using PreferenceFragment (Android >=3.0 API 10) or subcalss PreferenceActivity (Android < 3.0)
  - UI defined in XML
  - Each preference subclass is an item in the XML
    - stays res/xml as preferences.xml.
    - Key/Value pairs saved in application SharedPreferences file
    - Value types : Boolean, Float, Int, Long,S tring, String Set
    - Root node is <PreferenceScreen>
      - Subclasses
        - CheckBoxPreference, EditTextPreference, ListPreference, MultiSelectListPreference, PreferenceCategory, PreferenceScreen, SwitchPreference

Persistence des Donnees

- Preference Framework /Settings

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen
   xmlns:android="http://schemas.android.com/apk/res/android"
   >
<CheckBoxPreference
android:key="pref_sync"android:title="@string/pref_sync"
android:summary="@string/pref_sync_summ"
android:defaultValue="true" />
<ListPreference
android:dependency="pref_sync"
android:key="pref_syncConnectionType"
android:title="@string/pref_syncConnectionType"
android:dialogTitle="@string/pref_syncConnectionType"
android:entries="@array/pref_syncConnectionTypes_entries"
android:entryValues="@array/pref_syncConnectionTypes_values"
   android:defaultValue="@string/pref_syncConnectionTypes_defa
   ult" />
</PreferenceScreen>
```

Persistence des Donnees

- Framework Preference
  - Use category and subscreens

```
<PreferenceScreen  xmlns:android="http://schemas.android.com/apk/res/android">
  <!-- opens a subscreen of settings -->
  <PreferenceScreen
    android:key="button_voicemail_category_key"
    android:title="@string/voicemail"
    android:persistent="false">
    <ListPreference
      android:key="button_voicemail_provider_key"
      android:title="@string/voicemail_provider" ... />
    <!-- opens another nested subscreen -->
    <PreferenceScreen
      android:key="button_voicemail_setting_key"
      android:title="@string/voicemail_settings"
      android:persistent="false">
      ...
    </PreferenceScreen>
    <RingtonePreference
      android:key="button_voicemail_ringtone_key"
      android:title="@string/voicemail_ringtone_title"
      android:ringtoneType="notification" ... />
    ...
  </PreferenceScreen>
  ...
</PreferenceScreen>
```

  - Use intents

```
<Preference android:title="@string/prefs_web_page" >
  <intent android:action="android.intent.action.VIEW"
      android:data="http://www.example.com" />
</Preference>
```

Data Persistance

- Framework Preference
  - Create an activity
    - public class SettingsActivity extends PreferenceActivity {
    - @Override
    - public void onCreate(Bundle savedInstanceState) {
    - super.onCreate(savedInstanceState);
    - addPreferencesFromResource(R.xml.preferences);
    - }
    - }
- Using PreferenceFragments
  - Subclass PreferenceFragment
  - Inflate the Preferences

Persistence des Donnees

- PreferenceFragment
  - Subclass PreferenceFragment
  - Inflate the Preferences

Notifications

- Structure d'une Notification
  - Icon
  - Title
  - Text
  - TimeStamp
  - May content actions
- Creation
  - Get a builder
    - NotificationCompat.Builder
  - Get the notification
    - Notification NotificationCompat.Builder.build()
  - Get a NotificationManager
  - Send notification
    - NotificationManager.Notify()
- Priorites
  - Five levels

Notification

```
NotificationCompat.Builder mBuilder =  new NotificationCompat.Builder(this
    .setSmallIcon(R.drawable.notification_icon).setContentTitle("My notification")
    .setContentText("Hello World!");
// Creates an explicit intent for an Activity in your app
Intent resultIntent = new Intent(this, ResultActivity.class);
// The stack builder object will contain an artificial back stack for the started Activity.
// This ensures that navigating backward from the Activity leads out of
// your application to the Home screen.
TaskStackBuilder stackBuilder = TaskStackBuilder.create(this);
// Adds the back stack for the Intent (but not the Intent itself)
stackBuilder.addParentStack(ResultActivity.class);
// Adds the Intent that starts the Activity to the top of the stack
stackBuilder.addNextIntent(resultIntent);
PendingIntent resultPendingIntent =stackBuilder.getPendingIntent  0,
PendingIntent.FLAG_UPDATE_CURRENT);
mBuilder.setContentIntent(resultPendingIntent);
NotificationManager mNotificationManager =
(NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
// mId allows you to update the notification later on.
NotificationManager.notify(mId, mBuilder.build());
```

Notification

- Expanded layout
  - Get an inboxstyle

```
NotificationCompat.InboxStyle inboxStyle =
new NotificationCompat.InboxStyle();
```

Notification

- Gestion des notifications
  - Updating
    - Need notification ID
  - Removing
    - User clears
    - setautoCancel
    - Call Cancel() with an ID
    - Call CancelAll()
- Preserving navigation flow

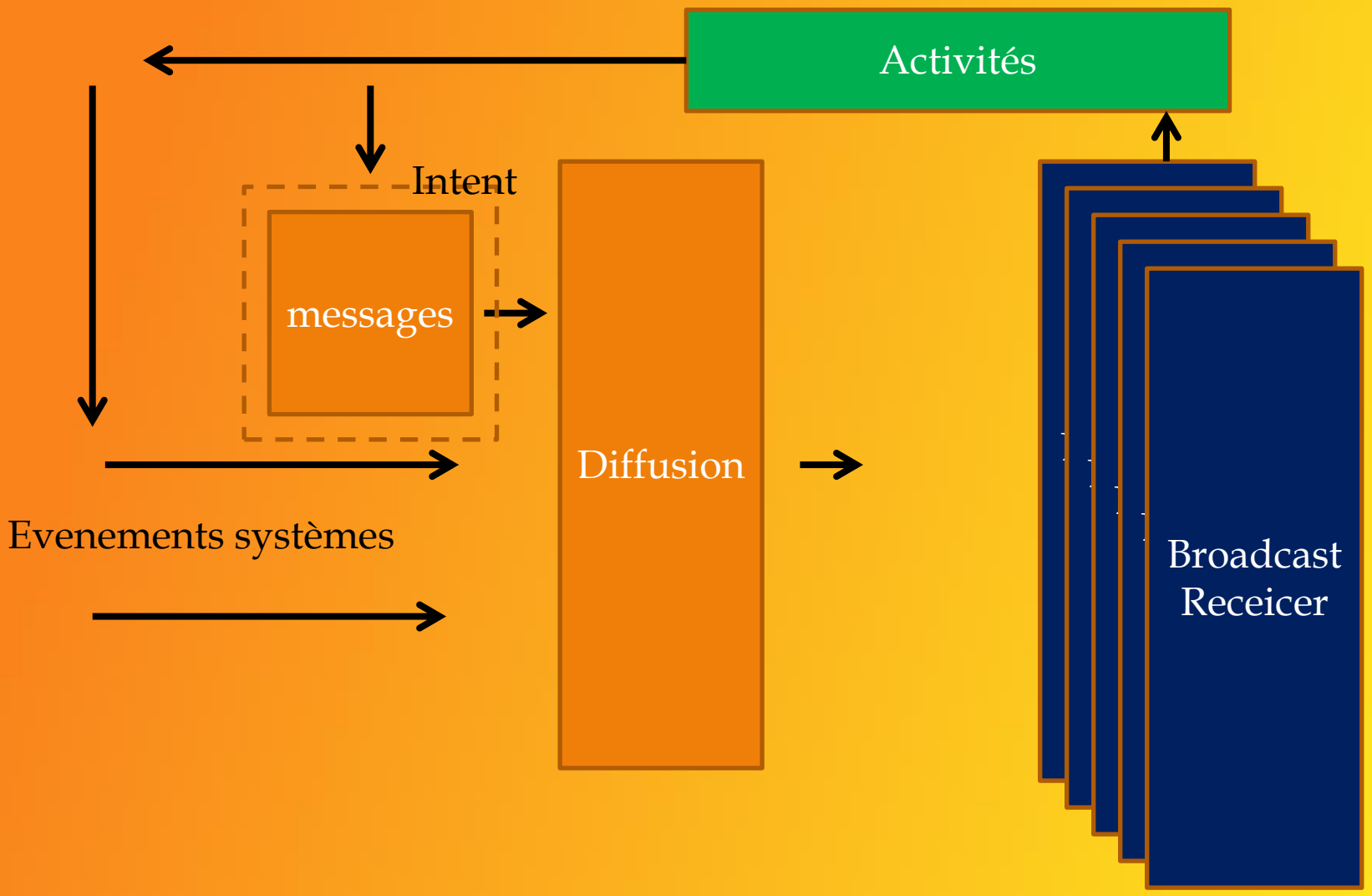
Notification

- Display progress
  - setProgress()
- MetaData
  - Sort notifications based on
    - setPriority()
    - setCategory()
    - addPerson()
- Custom layout
  - Using remoteView

Alarms

- Planification des taches depuis une alarme

Broadcast Receiver



Service-IntentService

- Service
  - no UI
  - Several services can be active within an application
  - A service can remain active even if the host activity is not the current forground application
  - The system is aware of running services, and will avoid killing the processes services are attached to
- Queuing work with intentService
  - Intent service is a subclass of service
  - Owns a background work queue
  - Use a single HandlerThread to manage queue processing
  - If the user quits the app, the queued work is completely processed

Service-IntentService

- 3 Types de Service
  - scheduled
  - Started
  - Bound
- Un service est une sous classe de Service
- Se lance par
  - startService()
    - onStartCommand()
    - stopSelf()
    - stopService()
  - onBindService()

Service-IntentService

- Intent intent = new Intent(context, MyIntentService.class);
  intent.setData(uri);
  intent.putExtra("param", "some value");
  startService(intent);
- public class MyIntentService extends IntentService {
  public MyIntentService() {
  super("thread-name");
  }
  protected void onHandleIntent(Intent intent) {
  // executes on the background HandlerThread.
  }

  Manifeste
- <service android:name="com.mypackage.MyIntentService"/>
- <service android:name=".MyIntentService"**android:exported="false"/>**

Service-IntentService

- Passage des résultats inter-thread
  - PendingIntent
  - Notification
  - Message
  - BroadcastReceiver

Service-IntentService

- Passage des résultats inter-thread
  - PendingIntent
    - public static final String PENDING_RESULT = "pending_result";
    - public static final String RESULT = "result";
    - p private static final int REQUEST_CODE = 0; ublic static
    - Dans main activity
      - final int RESULT_CODE = Result.Code();
    - PendingIntent pending = createPendingResult(REQUEST_CODE, new Intent(),(), 0);

Service-IntentService

- Passage des résultats inter-thread
  - PendingIntent

```
private void triggerIntentService(int nombreATraiter) {
PendingIntent pending = createPendingResult(
REQUEST_CODE, new Intent(), 0);
Intent intent = new
   Intent(this,TraitementIntentService.class);
intent.putExtra(TraitemetIntentService.PARAM,nombreAT
   raiter);
intent.putExtra(
TraitementIntentService.PENDING_RESULT, pending);
startService(intent);
}
```

Service-IntentService

- Passage des résultats inter-thread
- Callback dans le thread UI

```
protected void onActivityResult(int req, int res, Intent
   data) {
if (req == REQUEST_CODE &&
res == TraitementIntentService.RESULT_CODE) {
BigInteger result = (BigInteger)
data.getSerializableExtra(TraitementIntentService.RESULT
   );
// … update UI with the result
}
super.onActivityResult(requestCode, resultCode, data);
}
```

Service-IntentService

- Passage des résultats inter-thread
- Envoi des résultats par notification

```
private void notifyUser(int nombre, String result) {
String msg = String.format(
"The %sth nombre is %s", nombre, result);
NotificationCompat.Builder builder =
new NotificationCompat.Builder(this)
.setSmallIcon(R.drawable.nombre_notification_icon)
.setContentTitle(getString(R.string.nombre_app))
.setContentText(msg);
NotificationManager nm = (NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);
nm.notify(nombre, builder.build());
}
```

Service-IntentService

- Passage des résultats inter-thread
- Envoi des résultats par message
- Utilisation de broadCastReceiver

- **final WeakReference<Callback> maybeCallback =**

Service-IntentService

- Service
  - no UI
  - Several services can be active within an application
  - A service can remain active even if the host activity is not the current forground application
  - The system is aware of running services, and will avoid killing the processes services are attached to
- Queuing work with intentService
  - Intent service is a subclass of service
  - Owns a background work queue
  - Use a single HandlerThread to manage queue processing
  - If the user quits the app, the queued work is completely processed

Service-IntentService

- 3 Types de Service
  - scheduled
  - Started
  - Bound
- Un service est une sous classe de Service
- Se lance par
  - startService()
    - onStartCommand()
    - stopSelf()
    - stopService()
  - onBindService()

Service-IntentService

- Intent intent = new Intent(context, MyIntentService.class);
  intent.setData(uri);
  intent.putExtra("param", "some value");
  startService(intent);
- public class MyIntentService extends IntentService {
  public MyIntentService() {
  super("thread-name");
  }
  protected void onHandleIntent(Intent intent) {
  // executes on the background HandlerThread.
  }

  Manifeste
- <service android:name="com.mypackage.MyIntentService"/>
- <service android:name=".MyIntentService"**android:exported="false"/>**

Service-IntentService

- Passage des résultats inter-thread
  - PendingIntent
  - Notification
  - Message
  - BroadcastReceiver

Service-IntentService

- Passage des résultats inter-thread
  - PendingIntent
    - public static final String PENDING_RESULT = "pending_result";
    - public static final String RESULT = "result";
    - p private static final int REQUEST_CODE = 0; ublic static
    - Dans main activity
      - final int RESULT_CODE = Result.Code();
    - PendingIntent pending = createPendingResult(REQUEST_CODE, new Intent(),(), 0);

Service-IntentService

- Passage des résultats inter-thread
  - PendingIntent

```
private void triggerIntentService(int nombreATraiter) {
PendingIntent pending = createPendingResult(
REQUEST_CODE, new Intent(), 0);
Intent intent = new
   Intent(this,TraitementIntentService.class);
intent.putExtra(TraitemetIntentService.PARAM,nombreAT
   raiter);
intent.putExtra(
TraitementIntentService.PENDING_RESULT, pending);
startService(intent);
}
```

Service-IntentService

- Passage des résultats inter-thread
- Callback dans le thread UI

```
protected void onActivityResult(int req, int res, Intent
   data) {
if (req == REQUEST_CODE &&
res == TraitementIntentService.RESULT_CODE) {
BigInteger result = (BigInteger)
data.getSerializableExtra(TraitementIntentService.RESULT
   );
// … update UI with the result
}
super.onActivityResult(requestCode, resultCode, data);
}
```

Service-IntentService

- Passage des résultats inter-thread
- Envoi des résultats par notification

```
private void notifyUser(int nombre, String result) {
String msg = String.format(
"The %sth nombre is %s", nombre, result);
NotificationCompat.Builder builder =
new NotificationCompat.Builder(this)
.setSmallIcon(R.drawable.nombre_notification_icon)
.setContentTitle(getString(R.string.nombre_app))
.setContentText(msg);
NotificationManager nm = (NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);
nm.notify(nombre, builder.build());
}
```

Service-IntentService

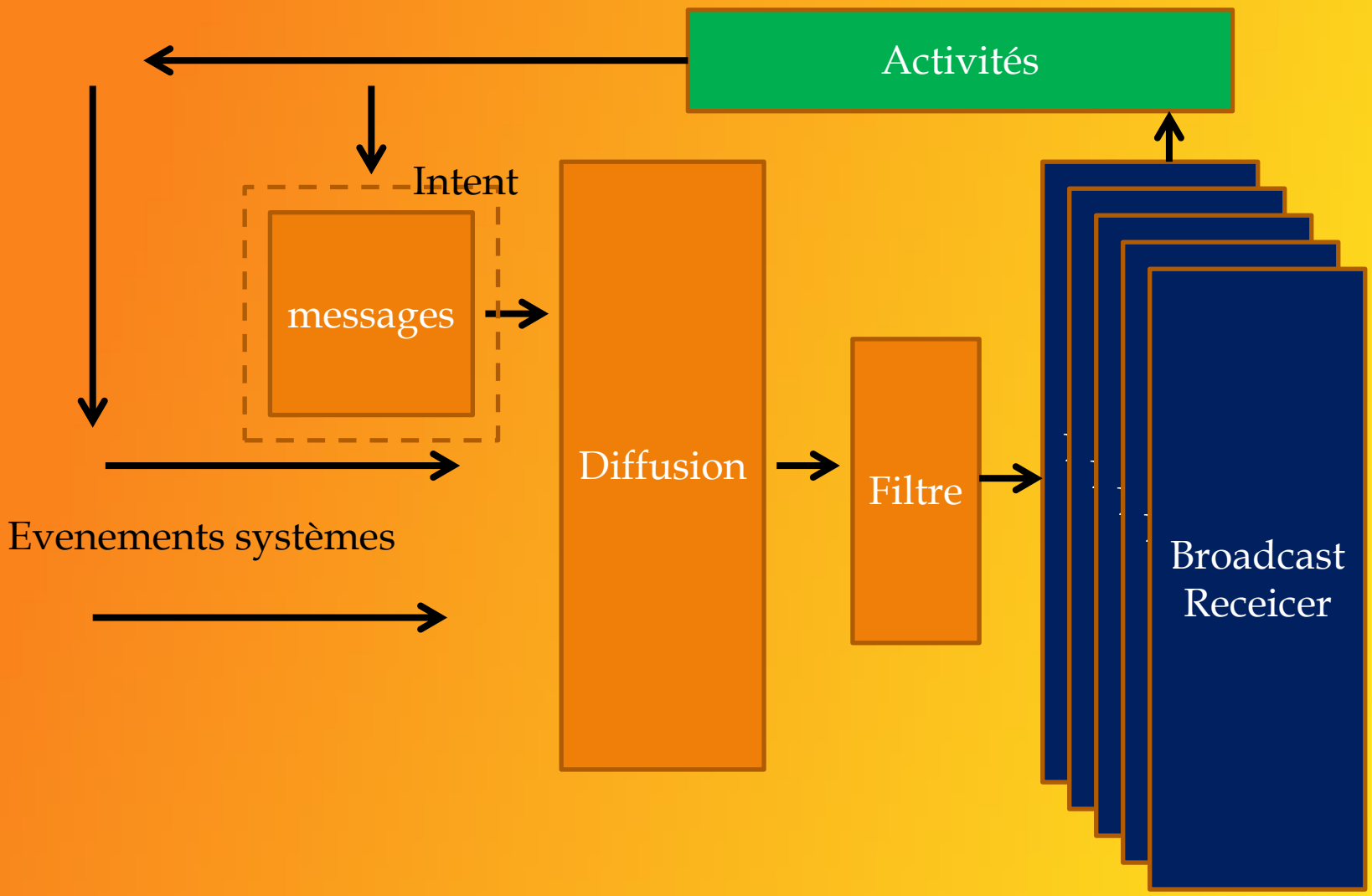
- Passage des résultats inter-thread
- Envoi des résultats par message
- Utilisation de broadCastReceiver

- **final WeakReference<Callback> maybeCallback =**

BroadcastReceiver

- Reception par les activités
  - Déclaration dans le manifeste
  - ```
    <receiver android:name=".MyBroadcastReceiver"  android:exported="true">
      <intent-filter>
        <action android:name="android.intent.action.BOOT_COMPLETED"/>
        <action android:name="android.intent.action.INPUT_METHOD_CHANGED" />
      </intent-filter>
    </receiver>
    ```
  - Implémenter onReceive callback
    - ```
      public class MyBroadcastReceiver extends BroadcastReceiver {
          private static final String TAG = "MyBroadcastReceiver";
          @Override
          public void onReceive(Context context, Intent intent) {
              StringBuilder sb = new StringBuilder();
              sb.append("Action: " + intent.getAction() + "\n");
              sb.append("URI: " +
      intent.toUri(Intent.URI_INTENT_SCHEME).toString() + "\n");
              String log = sb.toString();
              Log.d(TAG, log);
              Toast.makeText(context, log, Toast.LENGTH_LONG).show();
          }
      }
      ```

Broadcast Receiver



BroadcastReceiver

- Reception par les activités
  - Dans un contexte dynamique
    - Instancier la classe du boadcastReceiver
      - BroadcastReceiver br = new MyBroadcastReceiver();
  - Enregistrer le broadcastReceiver
    - IntentFilter filter = new IntentFilter(ConnectivityManager.CONNECTIVITY_ACTION);
      intentFilter.addAction(Intent.ACTION_AIRPLANE_MODE_CHANGED);
      this.registerReceiver(br, filter);
    - unregisterReceiver(android.content.BroadcastReceiver)

BroadcastReceiver

- Dans un contexte statique
  - Mettre à jour le manifest
    - <receiver android:name=".MyBroadcastReceiver"
      android:permission="android.permission.SEND_SMS">
      <intent-filter>
      <action
      android:name="android.intent.action.AIRPLANE_MODE"/>
      </intent-filter>
      </receiver>
    - Possibilité d'utiliser des permissions pour restreindre les récepteurs
      - sendBroadcast(new Intent("com.example.NOTIFY"),
        Manifest.permission.SEND_SMS);
  - Possibilité derestreindre les récepteurs aux membres d'un package

BroadcastReceiver

- Diffuser un BroadCast
  - sendOrderedBroadcast(Intent, String)
    - Un seul recepteur activé à la fois, selon un ordre de priorité
  - sendBroadcast(Intent)
    - Diffusion générale
      - Intent intent = new Intent();
        intent.setAction("com.example.broadcast.MY_NOTIFICATION");
        intent.putExtra("data","Notice me senpai!");
        sendBroadcast(intent);
  - LocalBroadcastManager.sendBroadcast
    - Diffusion restreinte à l'application
  - Possibilité de restreindre les récepteurs aux membres d'un package

Services

- Composant qui s'execute sans interface
- Peut exécuter des taches longues
- Types
  - Scheduled – lancé par JobScheduler
  - Started – lancé par sartService(intent)
  - Bound – lancé par bindService()
- Créer une classe qui étend Service
  - Callbacks
    - onStartCommand()
    - onBind()
    - onCreate()
    - onDestroy()
  - Doit être explicitement arrêté
    - stopService()
    - stopSelf()

Services

- Déclaration
  - Dans le manifeste
  - <manifest ... >
    ...
    <application ... >
      <service android:name=".ExampleService" />
      ...
    </application>
    </manifest>
  - Threads
    - Un service s'éxécute dans le même thread que l'activité appelante
    - Un IntentService s'éxécute dans u thread en background

Service-IntentService

- Service
  - no UI
  - Several services can be active within an application
  - A service can remain active even if the host activity is not the current forground application
  - The system is aware of running services, and will avoid killing the processes services are attached to
- Queuing work with intentService
  - Intent service is a subclass of service
  - Owns a background work queue
  - Use a single HandlerThread to manage queue processing
  - If the user quits the app, the queued work is completely processed

Service-IntentService

- 3 Types de Service
  - scheduled
  - Started
  - Bound
- Un service est une sous classe de Service
- Se lance par
  - startService()
    - onStartCommand()
    - stopSelf()
    - stopService()
  - onBindService()

Service-IntentService

- Intent intent = new Intent(context, MyIntentService.class);
  intent.setData(uri);
  intent.putExtra("param", "some value");
  startService(intent);
- public class MyIntentService extends IntentService {
  public MyIntentService() {
  super("thread-name");
  }
  protected void onHandleIntent(Intent intent) {
  // executes on the background HandlerThread.
  }

  Manifeste
- <service android:name="com.mypackage.MyIntentService"/>
- <service android:name=".MyIntentService"**android:exported="false"/>**

Service-IntentService

- Passage des résultats inter-thread
  - PendingIntent
  - Notification
  - Message
  - BroadcastReceiver

Service-IntentService

- Passage des résultats inter-thread
  - PendingIntent
    - public static final String PENDING_RESULT = "pending_result";
    - public static final String RESULT = "result";
    - p private static final int REQUEST_CODE = 0; ublic static
    - Dans main activity
      - final int RESULT_CODE = Result.Code();
    - PendingIntent pending = createPendingResult(REQUEST_CODE, new Intent(),(), 0);

Service-IntentService

- Passage des résultats inter-thread
  - PendingIntent

```
private void triggerIntentService(int nombreATraiter) {
PendingIntent pending = createPendingResult(
REQUEST_CODE, new Intent(), 0);
Intent intent = new
   Intent(this,TraitementIntentService.class);
intent.putExtra(TraitemetIntentService.PARAM,nombreAT
   raiter);
intent.putExtra(
TraitementIntentService.PENDING_RESULT, pending);
startService(intent);
}
```

Service-IntentService

- Passage des résultats inter-thread
- Callback dans le thread UI

```
protected void onActivityResult(int req, int res, Intent
   data) {
if (req == REQUEST_CODE &&
res == TraitementIntentService.RESULT_CODE) {
BigInteger result = (BigInteger)
data.getSerializableExtra(TraitementIntentService.RESULT
   );
// … update UI with the result
}
super.onActivityResult(requestCode, resultCode, data);
}
```

Service-IntentService

- Passage des résultats inter-thread
- Envoi des résultats par notification

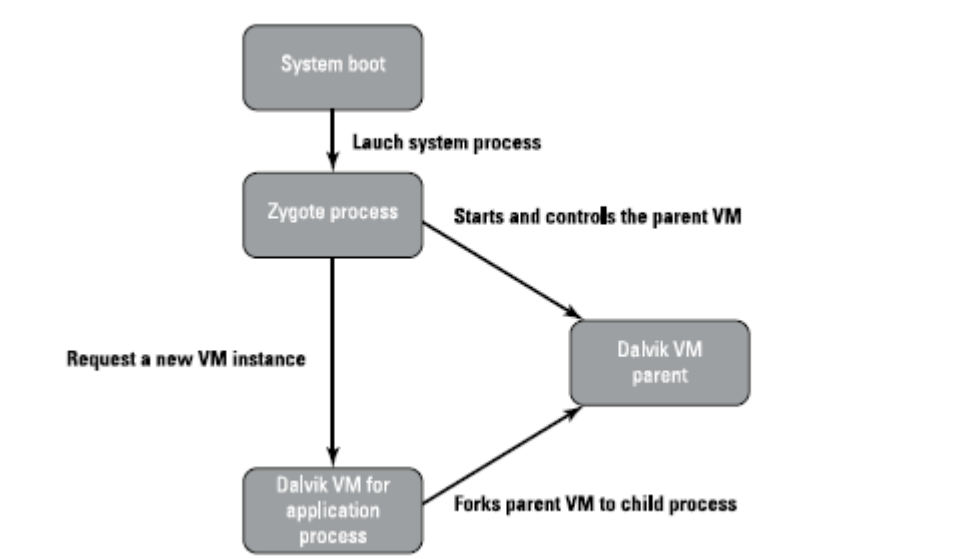
```
private void notifyUser(int nombre, String result) {
String msg = String.format(
"The %sth nombre is %s", nombre, result);
NotificationCompat.Builder builder =
new NotificationCompat.Builder(this)
.setSmallIcon(R.drawable.nombre_notification_icon)
.setContentTitle(getString(R.string.nombre_app))
.setContentText(msg);
NotificationManager nm = (NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);
nm.notify(nombre, builder.build());
}
```

Service-IntentService

- Passage des résultats inter-thread
- Envoi des résultats par message
- Utilisation de broadCastReceiver

- **final WeakReference<Callback> maybeCallback =**

MultiThreading - Staying responsive



- Application runs in the Main Thread

- Single thread model limits
  - computation consuming tasks
  - long network data transfer operations
  - long database loading or search operations

  - > risk for blocking UI tasks -> ANR message
  - since API 11 (Honeycomb) ExceptionRunTime error
  - if you try to execute NetWork tasks in the Main thread

MultiThreading

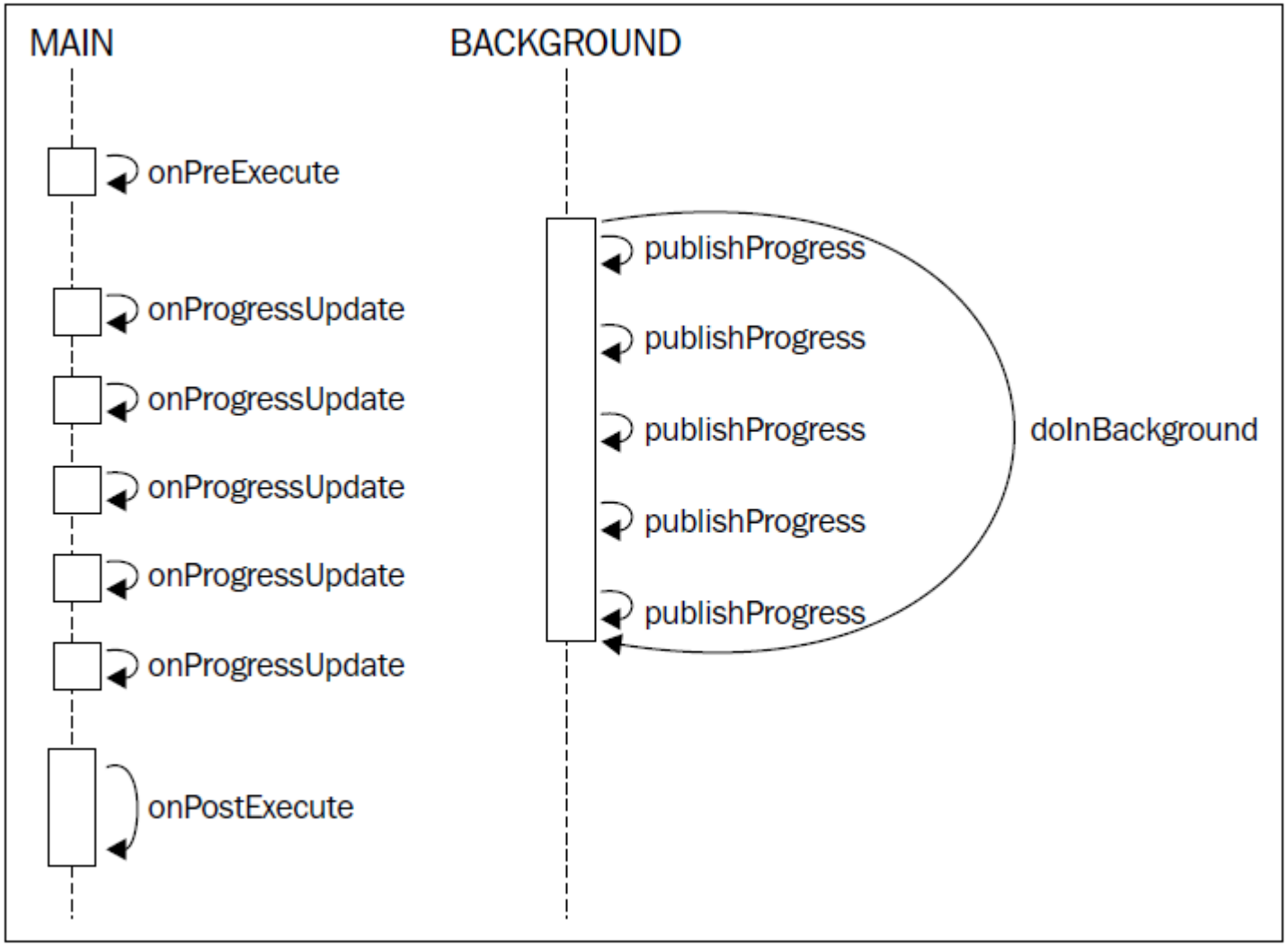
- To keep your application responsive
  - Offload time consuming task from Main Thread , handle them in background
  - All Java threading / concurrency mechanisms are usable (java.lang.Thread, java.lang.Runnable, synchronized , volatile keywords)
  - Correctness, liveness deadlock problems
  - Android specific
    - Memory leak problems – activity and garbage collector
    - Cannot manage UI views from other thread than main thread

Multi-threading

- Async task
  - Abstract class must be subclassed for use.
    - Mandatory
      - Implement
        - Result doInBackground(Params… params) method, (work to get done) runs in background
    - Optional
      - Implement
        - protected void onPreExecute()
        - protected void onProgressUpdate(Progress… values)
        - protected void onPostExecute(Result result)
        - protected void onCancelled(Result result)
        - Run in Main Thread

Multi-threading

Async task



Multi-Threading

- Declaring async task
  - abstract class AsyncTask<Params, Progress, Result>
  - private class DownloadFilesTask extends AsyncTask<URL, Integer, Long> {

```
protected Long doInBackground(URL... urls) {
    int count = urls.length;
    long totalSize = 0;
    for (int i = 0; i < count; i++) {
        totalSize += Downloader.downloadFile(urls[i]);
        publishProgress((int) ((i / (float) count) * 100));
        // Escape early if cancel() is called
        if (isCancelled()) break;
    }
    return totalSize;
}

protected void onProgressUpdate(Integer... progress) {
    setProgressPercent(progress[0]);
}

protected void onPostExecute(Long result) {
    showDialog("Downloaded " + result + " bytes");
}
}
```

Multi-Threading

- Executing async task
  - Two ways
    - public final AsyncTask<Params, Progress, Result> execute(Params… params)
      - new DownloadFilesTask().execute(url1, url2, url3);
    - Async task is a single use instance once started cannot be re-used
    - Executor SERIAL_EXECUTOR
    - Executor THREAD_POOL_EXECUTOR
    - AsyncTask<Params, Progress, Result> execute (Params... params)
    - AsyncTask<Params, Progress, Result> executeOnExecutor (Executor exec, Params... params)

Multi-Threading

- Executing async task
  - From the main activity

```
public class PrimesActivity extends Activity {
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_example_1);
final TextView resultView =
(TextView) findViewById(R.id.result);
Button goButton = (Button) findViewById(R.id.go);
goButton.setOnClickListener(
new View.OnClickListener() {
@Override
public void onClick(View view) {
new PrimesTask(resultView).execute(500);
}
});
}
}
```

Multi-Threading

- Executing async task
  - Providing feedback to the user

```
public class PrimesTask extends AsyncTask<Integer, Void, BigInteger>{
private Context ctx;
private ProgressDialog progress;
private TextView resultView;
public PrimesTask(Context ctx, TextView resultView) {
this.ctx = ctx;
this.resultView = resultView;
}
@Override
protected void onPreExecute() {
progress = new ProgressDialog(ctx);
progress.setTitle(R.string.calculating);
progress.setCancelable(false);
progress.show();
}
// … doInBackground …
@Override
protected void onPostExecute(BigInteger result) {
resultView.setText(result.toString());
progress.dismiss();
}
}

In the main activity pass the context on click event
goButton.setOnClickListener(new View.OnClickListener() {
public void onClick(View view) {
new PrimesTask(
PrimesActivity.this, resultView).execute(500);
}
});
```

Multi-Threading

- Executing async task
  - Updating a progress bar
    - From background task , publish progress values :
      - protected final void publishProgress(Progress... values)
    - From main thread use callback to get values
      - protected void onProgressUpdate(Progress… values)
    - Create an progress bar in main activity and update it

Multi-Threading

- Executing async task
  - Updating the progrees bar

```
public class PrimesTask extends
    AsyncTask<Integer, Integer, BigInteger>{
private Context ctx;
private ProgressDialog progress;
private TextView resultView;
public PrimesTask(Context ctx, TextView
    resultView) {
this.ctx = ctx;
this.resultView = resultView;
}
@Override
protected void onPreExecute() {
progress = new ProgressDialog(ctx);
progress.setTitle(R.string.calculating);
progress.setCancelable(false);
progress.setProgressStyle(ProgressDialog.STYL
    E_HORIZONTAL);
progress.setProgress(0);
progress.setMax(100);
progress.show();
}
@Override
protected void onProgressUpdate(Integer...
    values) {
progress.setProgress(values[0]);
}
// … doInBackground …
protected BigInteger doInBackground(Integer...
    params) {
int primeToFind = params[0];
BigInteger prime = new BigInteger("2");
int progress = 0;
for (int i=0; i<primeToFind; i++) {
prime = prime.nextProbablePrime();
int percent = (int)((i * 100f)/primeToFind);
if (percent > progress) {
publishProgress(percent);
progress = percent;
}
}
re
@Override
protected void onPostExecute(BigInteger
    result) {
resultView.setText(result.toString());
progress.dismiss();
}
}

In the main activity pass the context on click
    event
goButton.setOnClickListener(new
    View.OnClickListener() {
public void onClick(View view) {
new PrimesTask(
PrimesActivity.this, resultView).execute(500);
}
});
```

Multi-Threading

- Cancelling async task
  - Use async task cancel method
    - public final boolean cancel(boolean mayInterruptIfRunning)
    - Let the dialog box allow cancellation event, within onPreExecute

```
progress.setCancelable(true);
progress.setOnCancelListener(
new DialogInterface.OnCancelListener() {
public void onCancel(DialogInterface dialog) {
PrimesTask.this.cancel(false);
}
});
```

    - Stop the doInBackground work

```
protected BigInteger doInBackground(Integer... params) {
int primeToFind = params[0];
BigInteger prime = new BigInteger("2");
for (int i=0; i<primeToFind; i++) {
prime = prime.nextProbablePrime();
int percentComplete = (int)((i * 100f)/primeToFind);
publishProgress(percentComplete);
if (isCancelled())
break;
}
return prime;
}
```

Override onCancelled() or onCancelled5resut result)

```
protected void onCancelled(BigInteger result) {
if (result != null)
resultView.setText("cancelled at " + result.toString());
progress.dismiss();
}
```

Multi-Threading

- Handing async task exception
  - To fill
- Controlling concurrency
  - Use an executor enabling serial or parallel execution

Handler – Handler Thread

- Distributing work – Handlers
- Looper
  - Loop
    - Create and start a loop

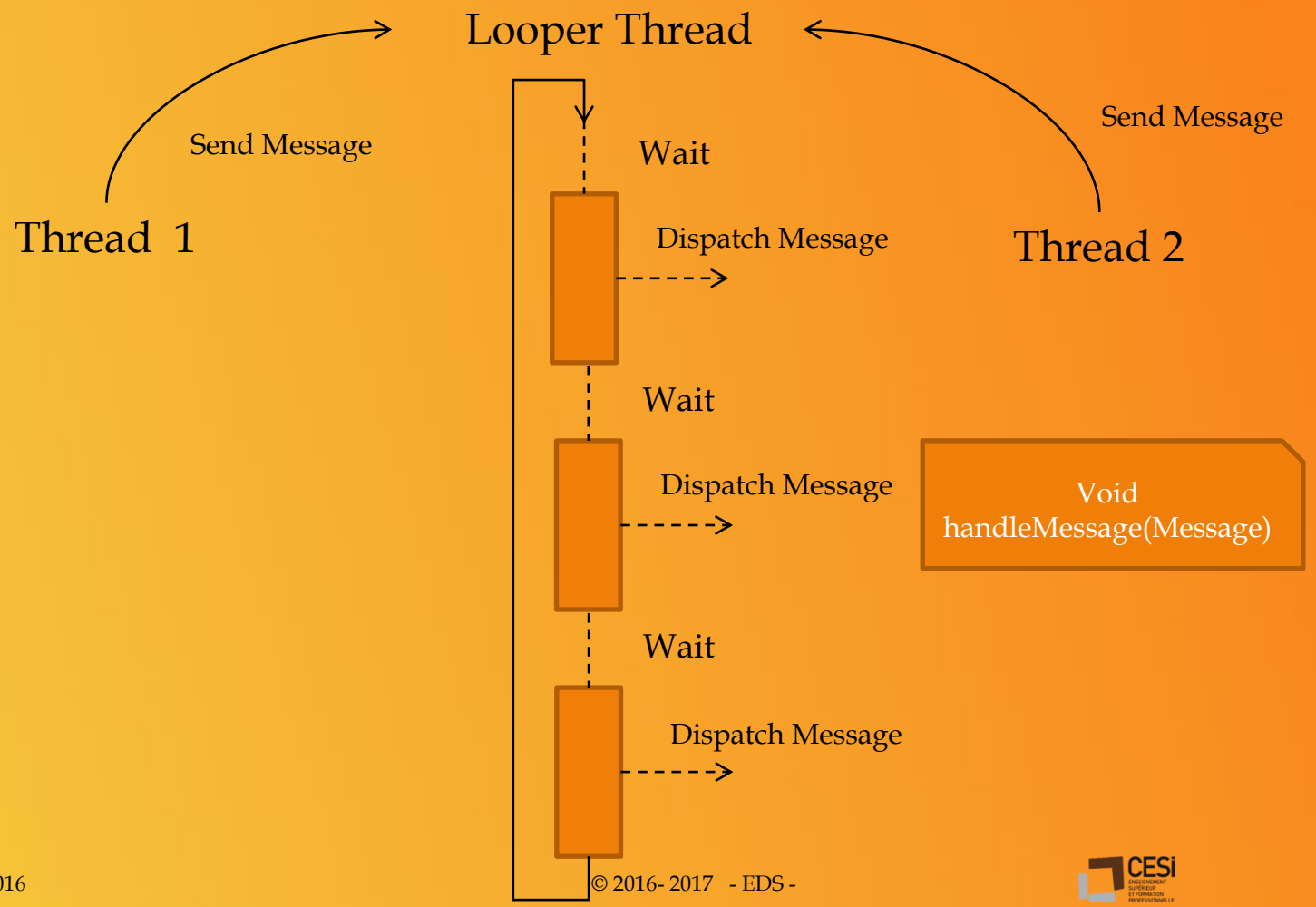
```
class SimpleLooper extends Thread {
public void run() {
Looper.prepare();
Looper.loop();
}
}
```

      - Setup handler

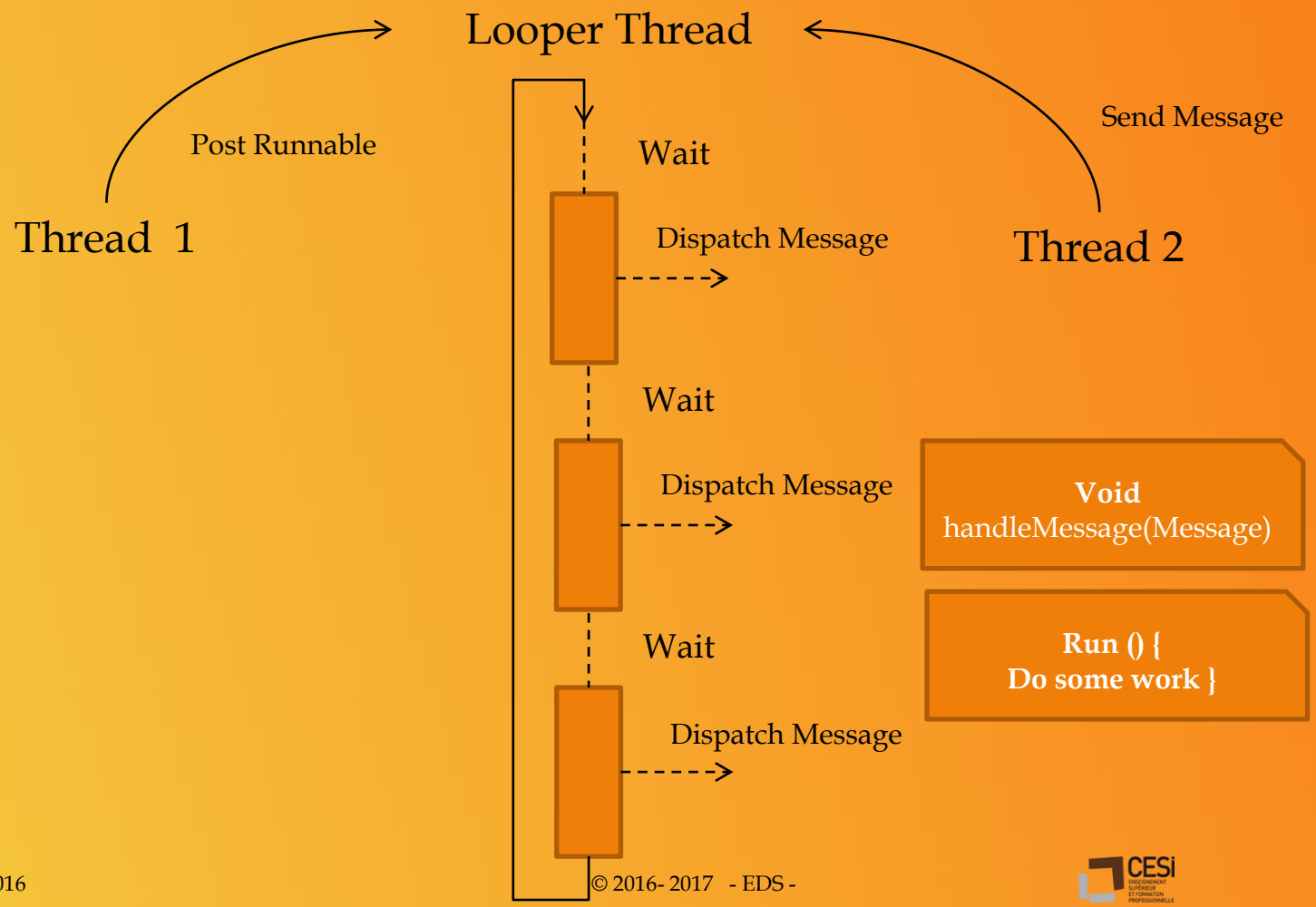
```
class SimpleLooper extends Thread {
public Handler handler;
public void run() {
Looper.prepare();
handler = new Handler();
Looper.loop();
}
}
```

    - A looper dispatches messages or work to do – work = runnable
    - Handler send what to do: messages, runnables to the looper

Handler



Handler



Handler – Handler Thread

- Post runnables

```
//anonymous work
handler.post(new Runnable(){
public void run() {
// do some work on the main thread.
}
});
//post ahead of other waiting runnables
handler.postAtFrontOfQueue(new Runnable(){
public void run() {
// do some work on the main thread.
}
});
//post with delay
handler.postDelayed(new Runnable(){
public void run() {
// do some work on the main thread
// in 10 seconds time
}
}, TimeUnit.SECONDS.toMillis(10));
//post at time
handler.postAtTime(new Runnable(){
public void run() {
// … do some work on the main thread
}
}, SystemClock.uptimeMillis() + TimeUnit.SECONDS.toMillis(10));
```

Handler – Handler Thread

- Post runnables
  - The handler can be used from any thread

```
//From main thread
handler.post(new Runnable(){
public void run() {
TextView text = (TextView) findViewById(R.id.text);
text.setText("updated on the UI thread");
}
});
//From an other thread , the way to pass data back to the mainthread
Thread thread = new Thread(){
public void run(){
final BigInteger result = calculatePrime(500);
handler.post(new Runnable(){
public void run(){
TextView text = (TextView)
findViewById(R.id.text);
text.setText(result.toString());
}
});
}
};
thread.setPriority(Thread.MIN_PRIORITY);
thread.start();
```

Handler – Handler Thread

- Post runnables
  - Within activity, use runOnUiThread as an alternative to execute task in the main thread.
  - Cancel a pending runnable

```
final Runnable runnable = new Runnable(){
public void run() {
// … do some work
}
};
handler.postDelayed(runnable, TimeUnit.SECONDS.toMillis(10));
Button cancel = (Button) findViewById(R.id.cancel);
cancel.setOnClickListener(new OnClickListener(){
public void onClick(View v) {
handler.removeCallbacks(runnable);
}
});
// Need to keep a reference to the runnable
//
```

Handler – Handler Thread

- Scheduling work with send
  - Get message from handler
  - Handle message from call back
    - handleMessage(Message)
  - Handle a message

```
public static class SpeakHandler extends Handler {
public static final int SAY_HELLO = 0;
public static final int SAY_BYE = 1;
@Override
public void handleMessage(Message msg) {
switch(msg.what) {
case SAY_HELLO:
sayWord("hello"); break;
case SAY_BYE:
sayWord("goodbye"); break;
default:
super.handleMessage(msg);
}
}
//
```

  - To attach the handler to main thread
    - Get an instance anywher in the main activity

```
private Handler handler;
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
handler = new SpeakHandler();
// …
}
```

Handler – Handler Thread

- Scheduling work with send
  - Recycling message

```
public static class SpeakHandler extends Handler {
public static final int SAY_HELLO = 0;
public static final int SAY_BYE = 1;
public static final int SAY_WORD = 2;
@Override
public void handleMessage(Message msg) {
switch(msg.what) {
case SAY_HELLO:
sayWord("hello"); break;
case SAY_BYE:
sayWord("goodbye"); break;
case SAY_WORD:
sayWord((String)msg.obj); break;
default:
super.handleMessage(msg);
}
}
private void sayWord(String word) { … }
}
//messsage  set up
handler.sendMessage(
Message.obtain(handler, SpeakHandler.SAY_WORD, "something"));
```

  - Scheduling message delivery
    - handler.sendMessageAtFrontOfQueue(msg);
    - handler.sendMessageAtTime(msg, time);
    - handler.sendMessageDelayed(msg, delay
  - Empty messages
    - handler.sendEmptyMessageAtTime(what, time);
    - handler.sendEmptyMessageDelayed(what, delay);

Handler – Handler Thread

- Cancelling message

```
handler.removeMessages(SpeakHandler.SAY_WORD);
```

  - Constructing handler
    - We use inheritance and subclassed Handler
    - Possibly use interface Handler.callback

```
public static class Speaker implements Handler.Callback {
public static final int SAY_HELLO = 0;
public static final int SAY_BYE = 1;
public static final int SAY_WORD = 2;
@Override
public boolean handleMessage(Message msg) {
switch(msg.what) {
case SAY_HELLO:
sayWord("hello"); break;
case SAY_BYE:
sayWord("goodbye"); break;
case SAY_WORD:
sayWord((String)msg.obj); break;
default:
return false;
}
return true;
}
private void sayWord(String word) { … }
}
//the handleMessage signature is different
Instanciate it
Handler handler = new Handler(new Speaker());
```

Handler – Handler Thread

- HandlerThread

```
handler.removeMessages(SpeakHandler.SAY_WORD);
```

PendingIntent

Material Design

- Règles de base pour établir un langage visuel.
  - Planifiées pour être évolutives
- Definit les règles de représentation du positionnement dans l'espace des composants (3D), la représentation de leur mouvement…
- Catégories de règles
  - Animation
  - Style
  - Layout
  - Composants
  - Patterns
  - Usability
  - Ressources

Material Design

- Règles de base pour établir un langage visuel.
  - Planifiées pour être évolutives
- Definit les règles de représentation du positionnement dans l'espace des composants (3D), la représentation de leur mouvement…
- Catégories de règles
  - Animation
  - Style
  - Layout
  - Composants
  - Patterns
  - Usability
  - Ressources

Style et Theme

- Personnalisation de l'aspect visuel d'une vue, d'une activité d'une application
  - <TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:textColor="#00FF00"
    android:typeface="monospace"
    android:text="@string/hello" />
  - Équivalent à du Css
- Style : regroupe dans un fichier xml les attributs applicables à une vue
  - <TextView
    style="@style/CodeFont"
    android:text="@string/hello" />

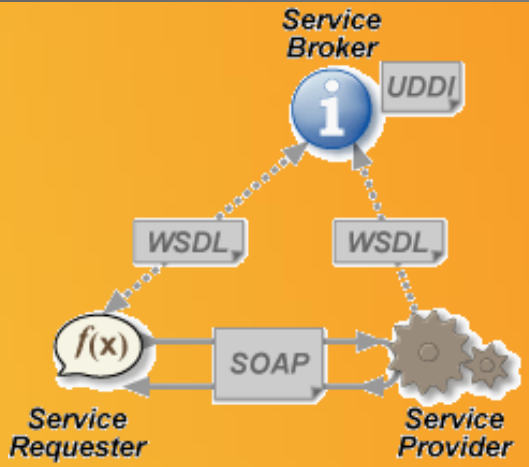
Styles et Themes

- Thème ; applicable à une activité, une application
  - <application android:theme="@style/CustomTheme">
  - <activity android:theme="@android:style/Theme.Translucent">
  - <color name="custom_theme_color">#b0b0ff</color>
    <style name="CustomTheme" parent="android:Theme.Light">
    <item name="android:windowBackground">@color/custom_theme_color</item>
    <item name="android:colorBackground">@color/custom_theme_color</item>
    </style>
- Thèmes associés aux versions de plateformes
  - Voir Classe R.style et R.att
  - Voir https://android.googlesource.com/platform/frameworks/base/+/refs/heads/master/core/res/res/values/styles.xml
  - Voir https://android.googlesource.com/platform/frameworks/base/+/refs/heads/master/core/res/res/values/themes.xml

WEB SERVICES

- Les Web Services sont un ensemble de moyens permettant d'exposer une API au travers d'un réseau et de façon à s'abstraire des technologies de mise en oeuvre (Technology-Neutral) .
- Les standrads de Web services sont encore en évolution.
  - Des tendances émergent:
    - SOAP pour les services de communication,
    - WSDL pour les services de description,
    - UDDI pour l'enregistrementeet la découverte de services,
    - BPEL(Business Process Execution Language) pour la composition de services .
    - Les tentatives de spécifications WS* sont pléthoriques et abordent l'ensemble du spectre des aspects clés des activites Web Service tels que : fiabilité des messages securité, données privées, coordination, traitement des événements ...
  - SOAP est un standard complexe. Quelques packages existent sous Android.

Architectures Web Service



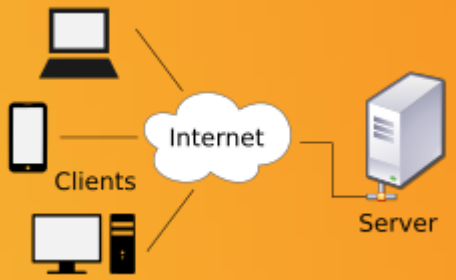
WSDL,
Web Services Description Language
UDDI
Universal Description Discovery and Infrastruc
SOAP
Simple Object Access Protocol

Limitations
Les services Web autres que RestFul sont complexes et construits sur des solutions logicielles non opensource

Problèmes potentiels de performance
Messages au format XML et SOAP/HTTP pour envelopper et transporter les messages.

SOAP est un protocole et non pas une architecture

Architectures Web Service



RestFull

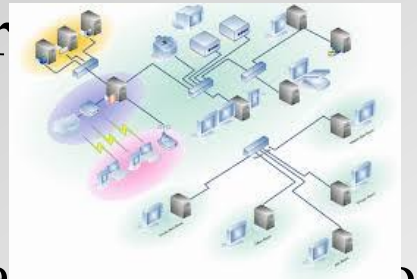
- Client-server
- Stateless
- Cacheable
- Layered system
- Code on demand
- Uniform interface

WEB SERVICES

- Ex de catalogues de web services
  - http://www.publicapis.com/
  - http://www.webservicex.net
  - http://www.programmableweb.com/
  - http://www.webservicelist.com/

WEB SERVICES

- Rappels sur les NetWork
- NetWork: groupe d'ordinateurs inter



- TCP/IP
  - Node: chaque device accessible sur un network est un node
  - Protocols: ensemble de règles prédéfinies et agrées qui permettent une communication.
  - Les protocoles sont organisés en couches / layers
    - TCP/IP stack
      - Link Layer—Physical. Un device fait appel à des protocoles résolution tels que ARP and RARP
      - Internet Layer—IP , a de multiples versions, protocole ping et ICMP, entre autres
      - Transport Layer—Met en œuvre differents types de protocole tels que TCP et UDP
      - Application Layer—Met en œuvre des protocoles tels que HTTP, FTP, SMTP, IMAP, POP, DNS, SSH, et SOAP

WEB SERVICES

- IP (Internet Packet)
  - IP adress ipv4 = 32 bits, ipv6 = 48 bits
- Protocoles de transport
  - TCP UDP (user datagram protocol) protocoles utilisés en TCP/IP
- Protocoles Application
  - (Simple Mail Transfer Protocol) SMTP , HTTP
- Clients and servers
  - Ports
    - Well-known ports—0 à 1023
    - Registered ports—1024 à 49151
    - Dynamic et/ou private ports—49152 à 65535

WEB SERVICES

- Obtenir l'état du Network
  - utiliser ConnectivityManager

```
@Override
public void onStart() {
super.onStart();
ConnectivityManager cMgr = (ConnectivityManager)
this.getSystemService(Context.CONNECTIVITY_SERVICE);
NetworkInfo netInfo = cMgr.getActiveNetworkInfo();
this.status.setText(netInfo.toString());
}
```

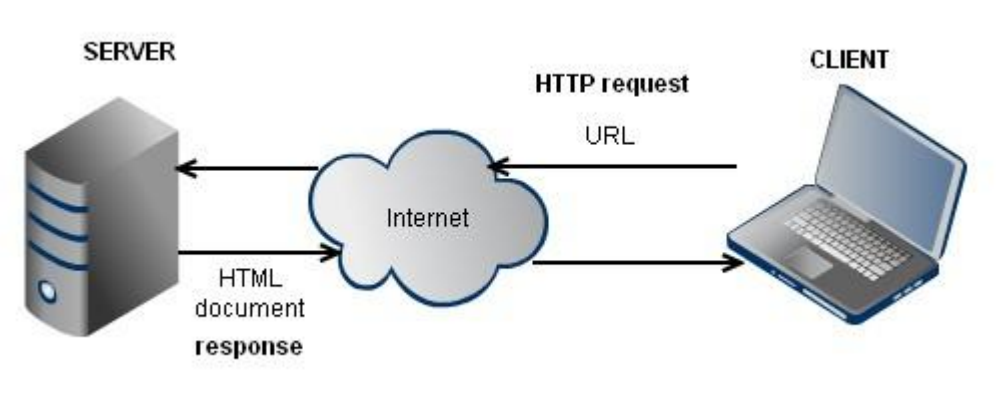
  - Un fois connecté vous pouvez utiliser l'IP Network

WEB SERVICES

- Communication avec un socket server
  - Server Socket
    - Stream pour read/write raw bytes
    - get local IP using ipconfig

SERVICES WEB

- Communication au protocole HTTP
- http



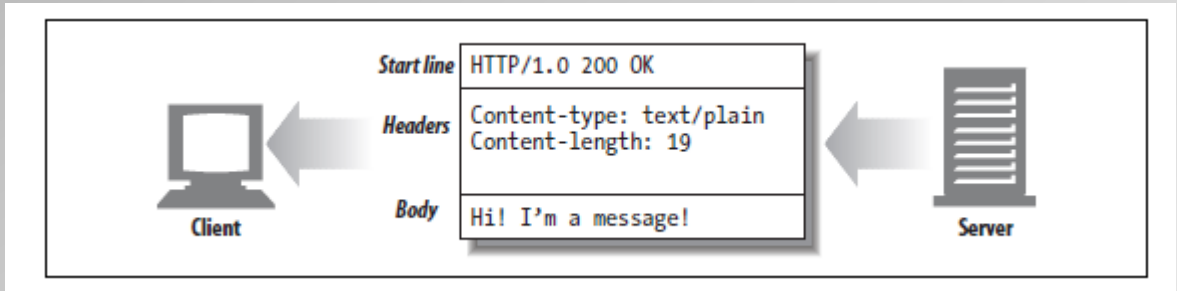
  - Ressources sur les serveurs
    - Db, files, images, html pages, Services …
    - Data types définis par le type MIME
    - Chaque ressources a un identificateur : URI (uniform ressource identifier), URL (uniform ressource locator) donne la localisation.

WEB SERVICES

- URL http://www.fast.hardware.com/specials/hammer.gif
  - Scheme : http://
  - Adress : www.fast.hardware.com
  - Ressource : specials/hammer.gif
- Transactions
  - Requests
  - Responses
  - Messages
  - Methods
    - GET Send named resource from the server to the client.
    - POST Send client data into a server gateway application.
    - PUT Store data from client into a named server resource.
    - DELETE Delete the named resource from a server.
    - HEADER Send HTTP headers from the response for the named resource.
  - Status codes
    - 200 Ok …

SERVICES WEB

- Messages
- 3 blocs



  - VERB URL VERSION (Start line)
  - HEADERS
  - BODY

```
GET / HTTP/1.1
Accept: text/*
Host: www.microsoft.com

HTTP/1.1 200 OK
Date: Thu, 01 Jun 2006 06:34:41 GMT
Server: Microsoft-IIS/6.0
P3P: CP="ALL IND DSP COR ADM CONo CUR CUSo IVAo IVDo PSA PSD TAI TELo OUR
SAMo CNT COM INT NAV ONL PHY PRE PUR UNI"
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
Cache-Control: private
Content-Type: text/html; charset=utf-8
Content-Length: 30430
```

SERVICES WEB

- using HTTP protocol
- Android support two Http clients
- HttpURLConnection
- Or
- HttpClient (Apache)

- Cas / Patron d'utilisation de HttpURLConnection
  - Methodes Htttp
    - Get utilisé par défaut
    - Post si setDoOutput(true)

SERVICES WEB

- Performance
  - Problème ANR
  - solution
    - Exécuter les opérations longues dans un Thread séparé
  - Vous ne pouvez pas effectuer des taches réseau dans le thread principal (Main Thread)
  - Vous ne pouvez pas mettre à jour les vues de l'IHM depuis les taches de fonds.

HttpURLConnection

- Manifest
  - <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
- Check network connexion

```
ConnectivityManager connMgr = (ConnectivityManager)
    getSystemService(Context.CONNECTIVITY_SERVICE);
NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();
if (networkInfo != null && networkInfo.isConnected()) {
    // fetch data
} else {
    // display error
}
```

- Execution dans thread séparé

```
private class DownloadWebpageTask extends AsyncTask<String, Void, String> {
    @Override
    protected String doInBackground(String... urls) {

        // params comes from the execute() call: params[0] is the url.
        try {
            return downloadUrl(urls[0]);
        } catch (IOException e) {
            return "Unable to retrieve web page. URL may be invalid.";
        }
    }
    // onPostExecute displays the results of the AsyncTask.
    @Override
    protected void onPostExecute(String result) {
        textView.setText(result);
    }
}
...
```

HttpURLConnection

- Check network connexion

```
private String downloadUrl(String myurl) throws IOException {
    InputStream is = null;
    // Only display the first 500 characters of the retrieved
    // web page content.
    int len = 500;

    try {
        URL url = new URL(myurl);
        HttpURLConnection conn = (HttpURLConnection) url.openConnection();
        conn.setReadTimeout(10000 /* milliseconds */);
        conn.setConnectTimeout(15000 /* milliseconds */);
        conn.setRequestMethod("GET");
        conn.setDoInput(true);
        // Starts the query
        conn.connect();
        int response = conn.getResponseCode();
        Log.d(DEBUG_TAG, "The response is: " + response);
        is = conn.getInputStream();

        // Convert the InputStream into a string
        String contentAsString = readIt(is, len);
        return contentAsString;

    // Makes sure that the InputStream is closed after the app is
    // finished using it.
    } finally {
        if (is != null) {
            is.close();
        }
    }
}
```

HttpURLConnection

- Read inputStream

```
// Reads an InputStream and converts it to a String.
public String readIt(InputStream stream, int len) throws
IOException, UnsupportedEncodingException {
    Reader reader = null;
    reader = new InputStreamReader(stream, "UTF-8");
    char[] buffer = new char[len];
    reader.read(buffer);
    return new String(buffer);
}
```

JSON

- Structure

```
{"menu": {
  "id": "file",
  "value": "File",
  "popup": {
    "menuitem": [
      {"value": "New", "onclick":
"CreateNewDoc()"},
      {"value": "Open", "onclick":
"OpenDoc()"},
      {"value": "Close", "onclick":
"CloseDoc()"}
    ]
  }
}}
```

- Parsing

The same text expressed as XML:

```
<menu id="file" value="File">
  <popup>
    <menuitem value="New"
onclick="CreateNewDoc()" />
    <menuitem value="Open"
onclick="OpenDoc()" />
    <menuitem value="Close"
onclick="CloseDoc()" />
  </popup>
</menu>
```

JSON

- Structure

```
{"widget": {
    "debug": "on",
    "window": {
        "title": "Sample Konfabulator Widget",
        "name": "main_window",
        "width": 500,
        "height": 500
    },
    "image": {
        "src": "Images/Sun.png",
        "name": "sun1",
        "hOffset": 250,
        "vOffset": 250,
        "alignment": "center"
    },
    "text": {
        "data": "Click Here",
        "size": 36,
        "style": "bold",
        "name": "text1",
        "hOffset": 250,
        "vOffset": 100,
        "alignment": "center",
        "onMouseUp": "sun1.opacity = (sun1.opacity / 100)
* 90;"
    }
}}
```

The same text expressed as XML:

```
<widget>
    <debug>on</debug>
    <window title="Sample Konfabulator Widget">
        <name>main_window</name>
        <width>500</width>
        <height>500</height>
    </window>
    <image src="Images/Sun.png" name="sun1">
        <hOffset>250</hOffset>
        <vOffset>250</vOffset>
        <alignment>center</alignment>
    </image>
    <text data="Click Here" size="36" style="bold">
        <name>text1</name>
        <hOffset>250</hOffset>
        <vOffset>100</vOffset>
        <alignment>center</alignment>
        <onMouseUp>
            sun1.opacity = (sun1.opacity / 100) * 90;
        </onMouseUp>
    </text>
</widget>
```

WEBSERVICES-XML

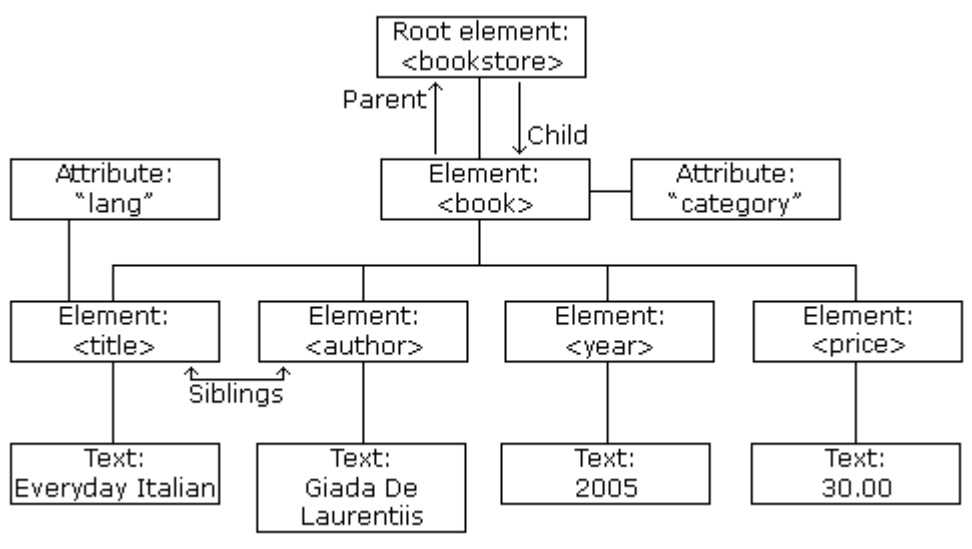
- Data representation/description
  - XML document structure
    - Organized as a tree structure starting from root

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <dest>Marc</dest>
  <de>Jeanne</de>
  <sujet>Rappel</sujet>
  <cont>C'est l'anniversaire de Julie ce week-end!</cont>
</note>
```

  - Contains root element <note>
  - Contains 4 child elements : dest, de, sujet,cont

```
<root>
  <child>
    <subchild>.....</subchild>
  </child>
</root>
```

- Data representation/description
  - Elements can have text content and attributes
  - Parent, children, sibling



WEBSERVICES-XML

- Data representation/description
  - Elements

```
<bookstore>
  <book category="COOKING">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="CHILDREN">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="WEB">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

WEBSERVICES-XML

- Data representation/description
- Language is case sensitive
- Well formed document
  - Must have a root element
  - Open tag must have a closing tag
  - Elements must be properly nested
  - Attributes values must be quoted
  - Predefined entity references

&lt;	<	less than
&gt;	>	greater than
&amp;	&	ampersand
&apos;	'	apostrophe
&quot;	"	quotation mark

Comments

```
<!-- This is a comment -->
```

WEBSERVICES-XML

- Data representation/description
- Well formed document
  - Element names must start with a letter or underscore
  - Element names cannot start with the letters xml (or XML, or Xml, etc)
  - Element names can contain letters, digits, hyphens, underscores, and periods
  - Element names cannot contain spaces

WEBSERVICES-XML

- Data representation/description
  - Attributes are quoted
    - <gangster name='George "Shotgun" Ziegler'>
    - <gangster name="George "Shotgun" Ziegler">
  - Elements versus attribute

```
<person gender="female">
  <firstname>Anna</firstname>
  <lastname>Smith</lastname>
</person>

<person>
  <gender>female</gender>
  <firstname>Anna</firstname>
  <lastname>Smith</lastname>
</person>
```

WEBSERVICES-JSON

- Data representation/description
  - JSON Java Script Open Notation
  - An alternative to XML

```
{"employees":[
   {"firstName":"John", "lastName":"Doe"},
   {"firstName":"Anna", "lastName":"Smith"},
   {"firstName":"Peter", "lastName":"Jones"}
]}
```

```
<employees>
   <employee>
      <firstName>John</firstName> <lastName>Doe</lastName>
   </employee>
   <employee>
      <firstName>Anna</firstName> <lastName>Smith</lastName>
   </employee>
   <employee>
      <firstName>Peter</firstName> <lastName>Jones</lastName>
   </employee>
</employees>
```

WEBSERVICES-JSON

- Data representation/description
  - JSON Java Script Open Notation
  - An alternative to XML
  - JSON
    - Data
      - Data: pair name/value
      - "firstName":"John«
      - Data separator ,
    - Object
      - {…}
      - {"firstName":"John", "lastName":"Doe"}
    - Array
      - […]

      "employees":[
      {"firstName":"John", "lastName":"Doe"},
      {"firstName":"Anna", "lastName":"Smith"},
      {"firstName":"Peter","lastName":"Jones"}
      ]

      MIME type for JSON text is "application/json

Generating XML

- Create XML data

```
public static String writeUsingNormalOperation(Study study) {
   String format =
         "<?xml version='1.0' encoding='UTF-8'?>" +
         "<record>" +
         "   <study id='%d'>" +
         "      <topic>%s</topic>" +
         "      <content>%s</content>" +
         "      <author>%s</author>" +
         "      <date>%s</date>" +
         "   </study>" +
         "</record>";
   return String.format(format, study.mId, study.mTopic,
study.mContent, study.mAuthor, study.mDate);
}
```

Generating XML

- Create XML data using DOM

```
public static String writeUsingDOM(Study study) throws Exception {
   Document doc = DocumentBuilderFactory.newInstance().newDocumentBuilder().newDocument();
   // create root: <record>
   Element root = doc.createElement(Study.RECORD);
   doc.appendChild(root);

   // create: <study>
   Element tagStudy = doc.createElement(Study.STUDY);
   root.appendChild(tagStudy);
   // add attr: id =
   tagStudy.setAttribute(Study.ID, String.valueOf(study.mId));

   // create: <topic>
   Element tagTopic = doc.createElement(Study.TOPIC);
   tagStudy.appendChild(tagTopic);
   tagTopic.setTextContent(study.mTopic);

   // create: <content>
   Element tagContent = doc.createElement(Study.CONTENT);
   tagStudy.appendChild(tagContent);
   tagContent.setTextContent(study.mContent);

   // create: <author>
   Element tagAuthor = doc.createElement(Study.AUTHOR);
   tagStudy.appendChild(tagAuthor);
   tagAuthor.setTextContent(study.mAuthor);

   // create: <date>
   Element tagDate = doc.createElement(Study.DATE);
   tagStudy.appendChild(tagDate);
   tagDate.setTextContent(study.mDate);

   // create Transformer object
   Transformer transformer = TransformerFactory.newInstance().newTransformer();
   StringWriter writer = new StringWriter();
   StreamResult result = new StreamResult(writer);
   transformer.transform(new DOMSource(doc), result);

   // return XML string
   return writer.toString();
}
```

Generating XML

- Create XML data using XML serializer

```
public static String writeUsingXMLSerializer(Study study) throws Exception {
   XmlSerializer xmlSerializer = Xml.newSerializer();
   StringWriter writer = new StringWriter();

   xmlSerializer.setOutput(writer);
   // start DOCUMENT
   xmlSerializer.startDocument("UTF-8", true);
   // open tag: <record>
   xmlSerializer.startTag("", Study.RECORD);
   // open tag: <study>
   xmlSerializer.startTag("", Study.STUDY);
   xmlSerializer.attribute("", Study.ID, String.valueOf(study.mId));

   // open tag: <topic>
   xmlSerializer.startTag("", Study.TOPIC);
   xmlSerializer.text(study.mTopic);
   // close tag: </topic>
   xmlSerializer.endTag("", Study.TOPIC);

   // open tag: <content>
   xmlSerializer.startTag("", Study.CONTENT);
   xmlSerializer.text(study.mContent);
   // close tag: </content>
   xmlSerializer.endTag("", Study.CONTENT);

   // open tag: <author>
   xmlSerializer.startTag("", Study.AUTHOR);
   xmlSerializer.text(study.mAuthor);
   // close tag: </author>
   xmlSerializer.endTag("", Study.AUTHOR);

   // open tag: <date>
   xmlSerializer.startTag("", Study.DATE);
   xmlSerializer.text(study.mDate);
   // close tag: </date>
   xmlSerializer.endTag("", Study.DATE);

   // close tag: </study>
   xmlSerializer.endTag("", Study.STUDY);
   // close tag: </record>
   xmlSerializer.endTag("", Study.RECORD);

   // end DOCUMENT
   xmlSerializer.endDocument();

   return writer.toString();
}
```

Generating JSON

- Use JSON class
  - Data
  - Object
  - Array

```
"employees":[
 {"firstName":"John", "lastName":"Doe"},
{"firstName":"Anna", "lastName":"Smith"},
{"firstName":"Peter","lastName":"Jones"}
]
JSONObject main = new JSONObject();
JSONArray arrayEmployee = new JSONArray();
for (int i = 0; i<=4; i++)
{
JSONObject employee = new JSONObject();
//La classe de base
try {
employee.put("firstName","Prenom"+"_"+Integer.toString(i));
employee.put("lasttName","Nom"+"_"+Integer.toString(i));
//ajout de la classe à l'array
arrayEmployee.put(employee);
} catch (JSONException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
}
main.put("Employees", arrayEmployee);
return main.toString();
}
```

Parsing - XML

- **Dom Parser** - Loads the complete contents of the document and creates its complete hiearchical tree in memory.
- **SAX Parser** - Does not load the complete document into the memory. Parses the document on event based triggers.
- **JDOM Parser** - Parses the document in similar fashion to DOM parser in a easier way
- **XPath Parser** - Parses the XML based on expression.
- **XmlPullParser** recommended by Goggle

Parsing JSON

- Structure JSON input

```
{
   "contacts": [
      {
            "id": "c200",
            "name": "Ravi Tamada",
            "email": "ravi@gmail.com",
            "address": "xx-xx-xxxx,x - street, x - country",
            "gender" : "male",
            "phone": {
               "mobile": "+91 0000000000",
               "home": "00 000000",
               "office": "00 000000"
            }
      },
      {
            "id": "c201",
            "name": "Johnny Depp",
            "email": "johnny_depp@gmail.com",
            "address": "xx-xx-xxxx,x - street, x - country",
            "gender" : "male",
            "phone": {
               "mobile": "+91 0000000000",
               "home": "00 000000",
               "office": "00 000000"
            }
      },
      .
      .
      .
      .
 ]
}
```

Parsing - JSON

- SAX Parser -

```
{
"person": {
"name": "John",
"age": 30,
"children": [
{
"name": "Billy"
"age": 5
},
{
"name": "Sarah"
"age": 7
},
{
"name": "Tommy"
"age": 9
}
]
}
}
```

```
@Override
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.main);
TextView line1 =
(TextView)findViewById(R.id.line1);
TextView line2 =
(TextView)findViewById(R.id.line2);
TextView line3 =
(TextView)findViewById(R.id.line3);
try {
JSONObject person =
(new
JSONObject(JSON_STRING)).getJSONObject("perso
n");
String name = person.getString("name");
line1.setText("This person's name is " + name);
line2.setText(name + " is " + person.getInt("age")
+ " years old.");
line3.setText(name + " has "
+ person.getJSONArray("children").length()
+ " children.");
} catch (JSONException e) {
e.printStackTrace();
}
```

Parsing - JSON

- **SAX Parser -**

```
@Override
protected Void doInBackground(Void... arg0) {
    HttpHandler sh = new HttpHandler();

    // Making a request to url and getting response
    String jsonStr = sh.makeServiceCall(url);

    Log.e(TAG, "Response from url: " + jsonStr);

    if (jsonStr != null) {
        try {
            JSONObject jsonObj = new JSONObject(jsonStr);

            // Getting JSON Array node
            JSONArray contacts = jsonObj.getJSONArray("contacts");

            // looping through All Contacts
            for (int i = 0; i < contacts.length(); i++) {
                JSONObject c = contacts.getJSONObject(i);

                String id = c.getString("id");
                String name = c.getString("name");
                String email = c.getString("email");
                String address = c.getString("address");
                String gender = c.getString("gender");

                // Phone node is JSON Object
                JSONObject phone = c.getJSONObject("phone");
                String mobile = phone.getString("mobile");
                String home = phone.getString("home");
                String office = phone.getString("office");


        }

        return null;
    }
```

- Decode suite

```
// tmp hash map for single contact
            HashMap<String, String> contact = new HashMap<>();

            // adding each child node to HashMap key => value
            contact.put("id", id);
            contact.put("name", name);
            contact.put("email", email);
            contact.put("mobile", mobile);

            // adding contact to contact list
            contactList.add(contact);
        }
    } catch (final JSONException e) {
        Log.e(TAG, "Json parsing error: " + e.getMessage());
        runOnUiThread(new Runnable() {
            @Override
            public void run() {
                Toast.makeText(getApplicationContext(),
                        "Json parsing error: " + e.getMessage(),
                        Toast.LENGTH_LONG)
                        .show();
            }
        });

    }
} else {
    Log.e(TAG, "Couldn't get json from server.");
    runOnUiThread(new Runnable() {
        @Override
        public void run() {
            Toast.makeText(getApplicationContext(),
                    "Couldn't get json from server. Check LogCat for possible errors!",
                    Toast.LENGTH_LONG)
                    .show();
        }
    });
```

FOURNISSEUR DE CONTENU

- Manages access to a central repository of data
- Applications access a provider to handle repository data.
- Applications use a provider client object
- Ex native Android content provider
  - Contact
  - Agenda
- Data is presented to external applications as one or more tables (Relational DB like, row columns)

word	App_id	frequency	locale	_ID
Kodie	user1	100	En_US	1
completude	user2	45	Fr_FR	2

.

CONTENT PROVIDER

- Application get access the data from a provider with a contentresolver client object
- The ContentResolver expose the basic "CRUD" (create, retrieve, update, and delete) methods of persistent storage access.
- Use permission are usually required for the application to use a content provider
  - ex using User Dictionary Provider
    - call ContentResolver.query().
    - Returns a cursor

CONTENT PROVIDER

- Contentresolver.query
  - mCursor = getContentResolver().query(
    UserDictionary.Words.CONTENT_URI, // The content URI of the words table
    mProjection, // The columns to return for each row
    mSelectionClause, // Selection criteria
    mSelectionArgs, // Selection criteria
    mSortOrder); // The sort order for the returned rows
  - Compared to Sql

query() argument	SELECT keyword/parameter	Notes
Uri	FROM table_name	Uri maps to the table in the provider named table_name.
projection	col,col,col,...	projection is an array of columns that should be included for each row retrieved.
selection	WHERE col = value	selection specifies the criteria for selecting rows.
selectionArgs	(No exact equivalent. Selection arguments replace? placeholders in the selection clause.)	
sortOrder	ORDER BY col,col,...	sortOrder specifies the order in which rows appear in the returned Cursor.

CONTENT PROVIDER

- URI
- content://user_dictionary/words
  - user_dictionary is provider's authority
  - Words is the table to access
  - content:// scheme is allways present , indicates this is a provider
- You append an ID to the URI to get one item
- Otherwise you get the full set of data

- Utils classes to build a URI
  - Uri
  - Uri.Builder
  - ContentUris (to append data to uri)

CONTENT PROVIDER

- To retrieve Data
  - Request read access to provider (use in application manifest)
  - Write code that sens a query to the provider
- Constructing a query
  - ```
    // A "projection" defines the columns that will be returned for each row
    String[] mProjection =
    {
       UserDictionary.Words._ID,    // Contract class constant for the _ID column name
       UserDictionary.Words.WORD,   // Contract class constant for the word column name
       UserDictionary.Words.LOCALE  // Contract class constant for the locale column name
    };
    ```
  - ```
    // Defines a string to contain the selection clause
    String mSelectionClause = null;
    ```
  - ```
    // Initializes an array to contain selection arguments
    String[] mSelectionArgs = {""};
    ```
    ```

Content Provider

- Contacts provider
- Entities

Content Provider

- Create a content provider

Data Persistence

- Shared Preferences
  - Allows to save/read key/values pairs into/from a file
- Content Provider

SQLite

- RDB léger 250 Kb
- Support les data types :
  - TEXT (String)
  - INTEGER(long)
  - REAL(double)
  - Toutes les données à stocker doivent être converties en ces types natifs
  - Pas de contrôle de type à l'écriture
- Le stockage des données a lieu dans le FileSystem d'android
  - DATA/data/APP_NAME/databases/FILENAME
    - DATA = répertoire Environment.getDataDirectory()
    - APP_NAME nom de votre application
    - FILENAME nom déclaré de la DB

SQLite

- Package : android.database.sqlite
- Création et mise à jour
  - Créer une classe dérivée de SQLiteOpenHelper
  - Surcharger onCreate() et onUpdate()
  - Ouverture en lecture/écriture
    - getReadableDatabase()/ getWriteableDatabase()
- CRUD
  - Méthodes : insert(), update(), delete()
  - execSQL()
- Query
  - rawQuery() (SQL statement)
  - query()
  - SQLiteQueryBuilder class .

SQLite

- Ex
  - Cursor cursor = getReadableDatabase(). rawQuery("select * from client where _id = ?", new String[] { id });
  - return database.query(DATABASE_TABLE, new String[] { KEY_ROWID, KEY_CATEGORY, KEY_SUMMARY, KEY_DESCRIPTION }, null, null, null, null, null);

SQLite

- Ex

Paramètre	Commentaire
String dbName	Le nom de la table pour laquelle la requête est compilée.
String[] columnNames	Une liste des colonnes à retourner. En passant "null", toutes les colonnes seront retournées.
String whereClause	Clause "where", filtre pour la sélection des données, "null" permet de sélectionner toutes les données.
String[] selectionArgs	Vous pouvez inclure des ? dans la "whereClause". Ces caractères génériques (placeholders) seront remplacés par les valeurs du tableau selectionArgs.
String[] groupBy	Un filtre qui déclare comment regrouper les lignes, avec "null" les lignes ne seront pas groupées.
String[] having	Filtre pour les groupes, null signifie pas de filtrage.
String[] orderBy	Colonnes de la table utilisées pour ordonner les données, avec null les données ne seront pas ordonnées.

Android

Création d'une application Android

Ph Dutron

URI - URL

- URI : Uniform Resource Identifier –RFC 2396
  - Séquence de caractères qui identifie une ressources abstraite ou physique
  - Ex
    - ftp://ftp.is.co.za/rfc/rfc1808.txt http://www.ietf.org/rfc/rfc2396.txt ldap://[2001:db8::7]/c=GB?objectClass?one mailto:John.Doe@example.com news:comp.infosystems.www.servers.unix tel:+1-816-555-1212 telnet://192.0.2.16:80/ urn:oasis:names:specification:docbook:dtd:xml:4.1.2
  - Syntaxe
    - URI = scheme ":" hier-part ["?" query] ["#" fragment]
    - hier-part = "//" authority path-abempty
      - / path-absolute
      - / path-rootless
      - / path-empty
      - Ex
      - foo://example.com:8042/over/there?name=ferret#nose
      - _/ _____________/_________/ _________/ __/
      - | | | | |
      - scheme authority path query fragment
      - | _____________________|__ / \ / \
      - urn:example:animal:ferret:nose
- URL : Uniform Resource Locator
  - Sous classe d'URI qui en plus précise la localisation d'une ressource (ex la localisation d'un réseau)

URI - URL

Expression Lambda

- Syntaxe
  - (*parameters*) ->*expression*

  ou
  - (parameters) ->{ statements; }
    - Ex :
      - () -> *10* // pas de paramètre renvoie 10
      - (x) -> 2 * x // x en paramètre retourne 2*x
      - (*x, y*) -> *x – y* // x et y en parmètres , retourne la diférence
      - (int x, int y) -> x + y duex entiers en paramètres, retourne la somme
      - (*String s*) -> *System.out.print(s)* // String en parmètres, pas de valeur de retour

Expression Lambda

- Cas d'usages
  - Gestion des évènements
  - Ex:
    - **public void** onClick(View view) {
      Snackbar.*make*(view, "**Replace with your own action**", Snackbar.***LENGTH_LONG***)
      .setAction("**Action**", **null**).show();
  - Snackbar vs Toast

Résumé

- 09/01/2017
  - Installation de GooglePlay Services
  - Généralités sur Location services
  - Obtention des clès API
  - API Location
  - Exemple getLastLocation
  - Exemple MapView
  - Utilisation de l'émulateur
- 10/01/2017
  - Exemple getLastLocation avec updates
  - Gestion des permissions en API 23 +
  - Résolution des problèmes de l'émulateur
  - Geocoder direct et inverse
  - Démarrage de l'appli tremblements de terre
  - Bases de REST services
  - Bases HTTP
  - Parsing de données JSON

Résumé

- 11/01/2017
  - Multithreading, asyncTask
  - Exercice affichage des tremblements de terre
  - parsing Json, asyncTask, mapsView, REST web service
- 12/01/2017
  - Rappels sur les composants
  - Activity
  - BroadcastReceiver
  - Service
  - ContentProvider
  - Exercice calculatrice
- 13/01/2017
    - Finalisation exercice calculatrice
    - Notification
    - Telephony
    - Contrôle