

Développement Android

Module 02 - Premiers pas

WARNING

Le contenu de cette présentation est basé sur la documentation anglophone officielle d'Android, diffusée sous licence *Creative Commons Attribution 2.5* :

developer.android.com

La plupart des schémas qui composent ce cours proviennent de cette documentation et sont, par conséquent, soumis à cette même licence.

<http://creativecommons.org/licenses/by/2.5/>

ANDROID DEVELOPER TOOLS



- Eclipse + SDK + Emulateur + Outils
- Disponible sur developer.android.com
- Une simple archive à décompresser.

UN PREMIER PROJET

The screenshot shows the Eclipse IDE interface for an Android project named 'MyApp'. The Package Explorer on the left shows the project structure, including the 'src' directory and the 'fr.inria.myapp' package containing 'MainActivity.java'. The main editor displays the code for 'MainActivity.java', which extends 'Activity' and implements 'onCreate' and 'onOptionsItemSelected' methods. The Outline on the right shows the class hierarchy, including 'fr.inria.myapp' and 'MainActivity'. The Problems window at the bottom is empty, indicating no errors or warnings.

```
package fr.inria.myapp;

import android.os.Bundle;

public class MainActivity extends Activity {

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

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

}
```

UN PREMIER PROJET

- Par défaut, l'IDE a généré :
 - Une activité (MainActivity.java)
 - Un manifest (AndroidManifest.xml)
 - Un dossier de ressources (/res) diverses contenant notamment une vue, des icônes et divers fichiers (gestion des tailles, du texte, etc.).
- Lors de la compilation, /gen est généré.
- /gen/<package>/R.java permet de faire le lien entre les ressources XML et la logique de l'application.

LE MANIFEST

- Décrit l'ensemble des composants qui constituent l'application.
- Décrit les permissions requises par l'application (accès internet, envoi de SMS, etc.), ainsi que les fonctionnalités matérielles et logicielles nécessaires (ex : appareil photo).
- Autres (SDK préféré, résolution supportées, etc.)

LE MANIFEST

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="fr.inria.myapp"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk android:minSdkVersion="8" android:targetSdkVersion="18" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="fr.inria.myapp.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

LE MANIFEST

```
<uses-permission android:name="android.permission.CALL_EMERGENCY_NUMBERS" />  
<uses-permission android:name="android.permission.RECEIVE_SMS" />  
<uses-permission android:name="android.permission.WRITE_PROFILE" />
```

...

```
<supports-screens android:resizeable=["true" | "false"]  
    android:smallScreens=["true" | "false"]  
    android:normalScreens=["true" | "false"]  
    android:largeScreens=["true" | "false"]  
    android:xlargeScreens=["true" | "false"]  
    android:anyDensity=["true" | "false"]  
    android:requiresSmallestWidthDp="integer"  
    android:compatibleWidthLimitDp="integer"  
    android:largestWidthLimitDp="integer"/>
```

```
<uses-feature android:name="android.hardware.bluetooth" />  
<uses-feature android:name="android.hardware.camera" />  
<uses-feature android:name="android.software.app_widgets" />
```

...

LES RESSOURCES

/res

/layout	Layout descriptions
/menu	Menu descriptions
/values	
/strings.xml	Default strings
/dimens.xml	Default dimensions
/styles.xml	Default styles
/values-*	Specific values
/drawable-*	Graphical resources (images)

/res/values-en/strings.xml : traduction anglaise.

/res/drawable-ja : images optimisés pour le public japonais.

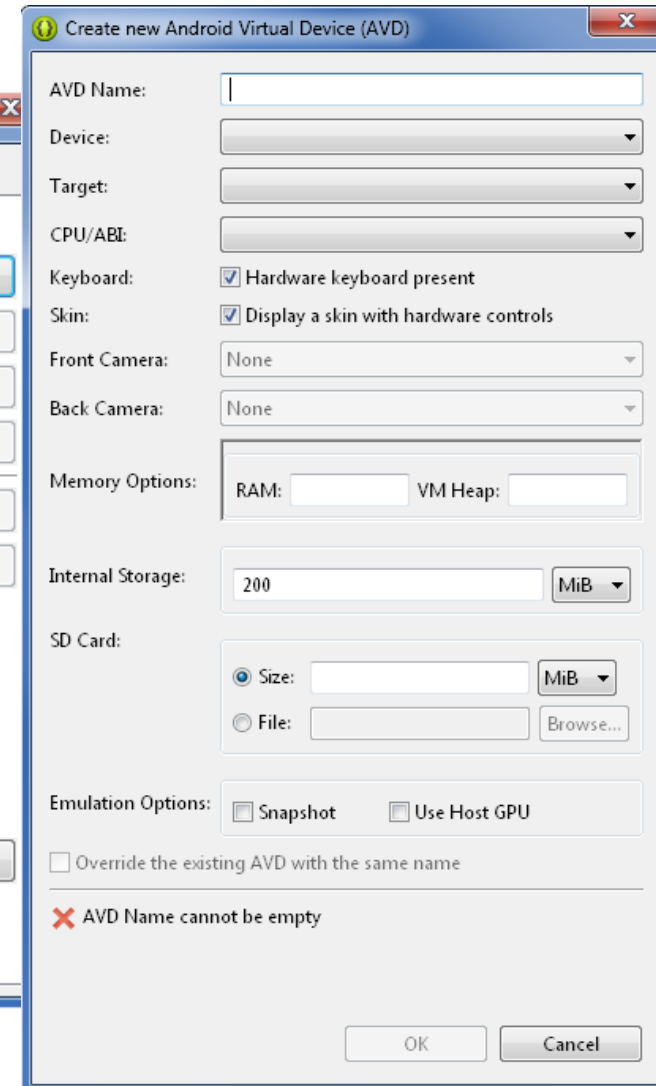
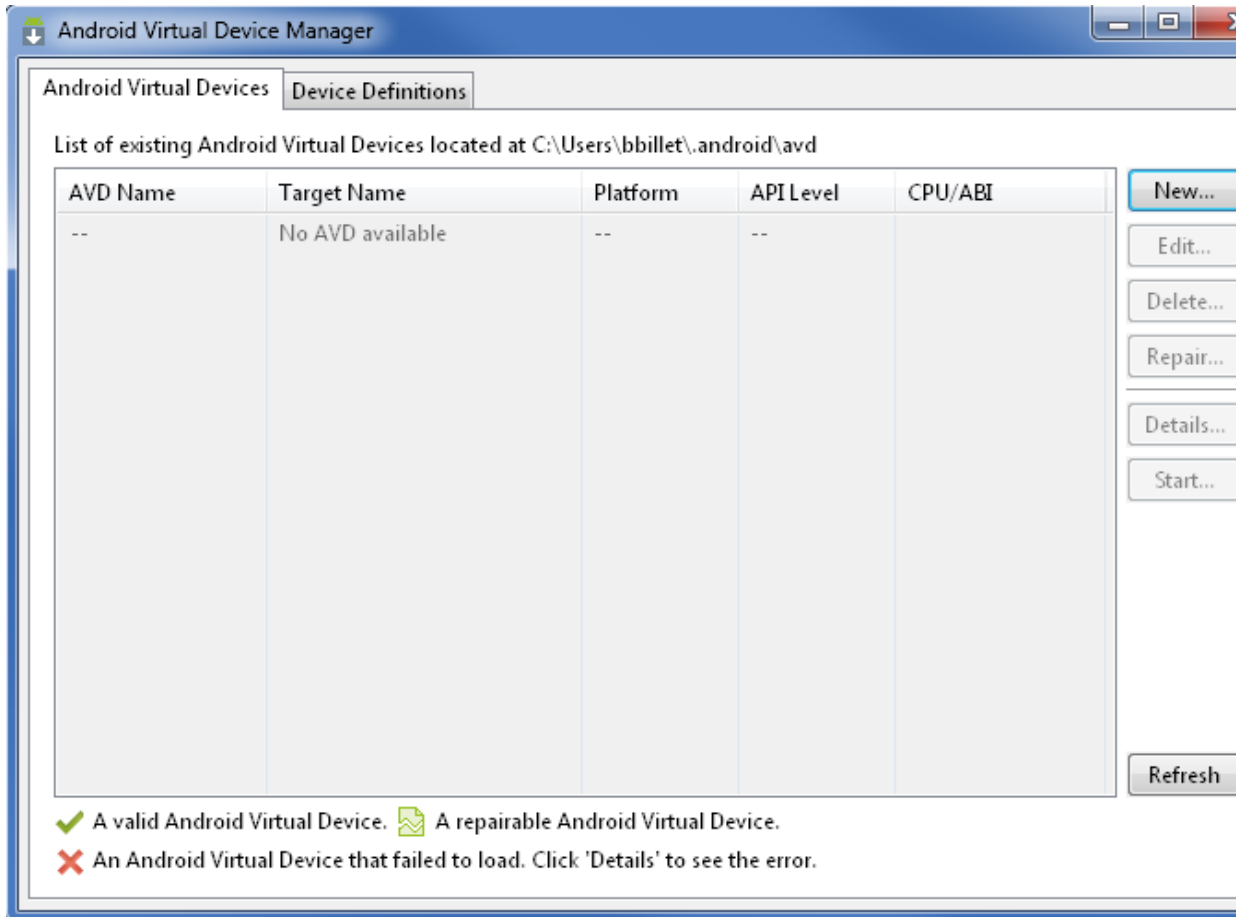
/res/values-v14/styles.xml : style spécifique à l'API 14

/res/drawable-hdpi : images optimisées pour écran haute densité (~240 dpi).

developer.android.com/guide/topics/resources/providing-resources.html

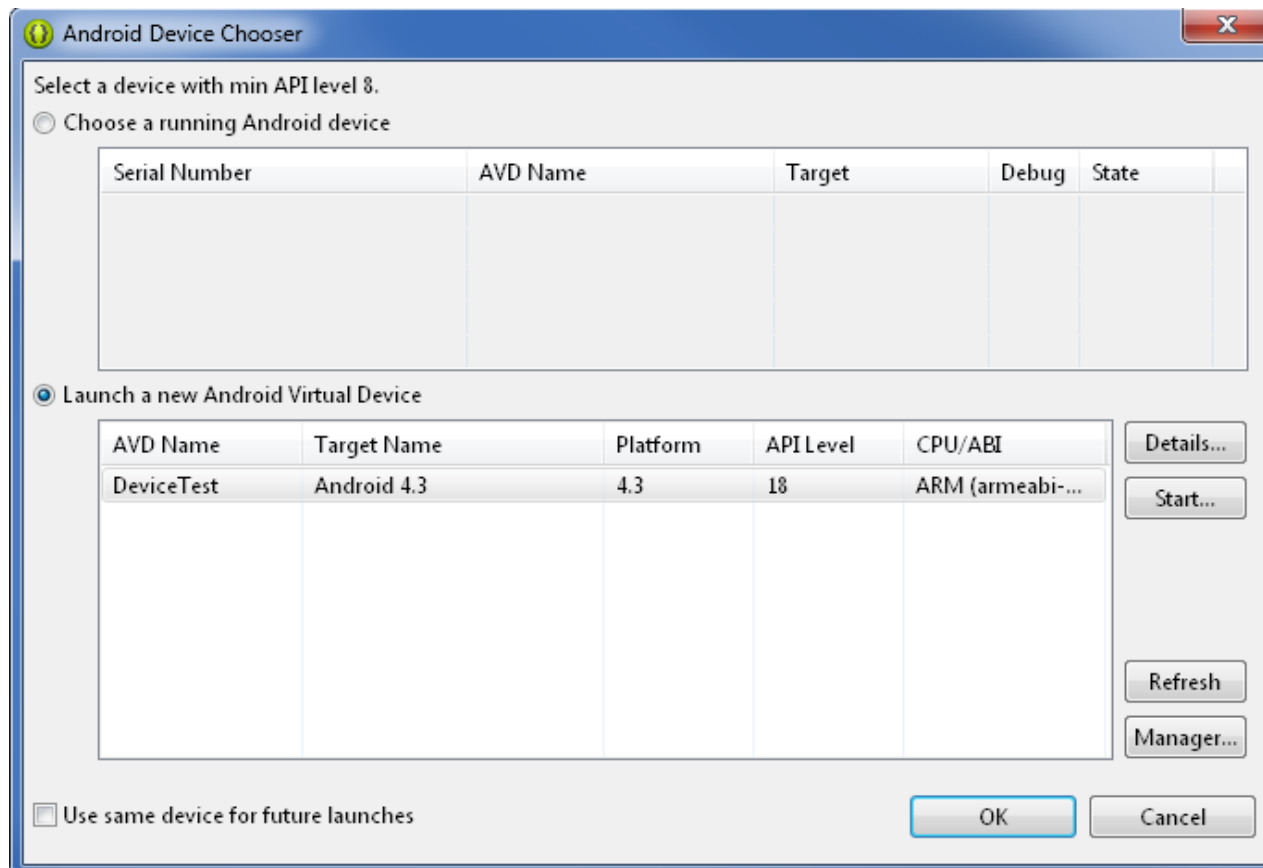
VIRTUAL DEVICE

Window -> Virtual Device Manager



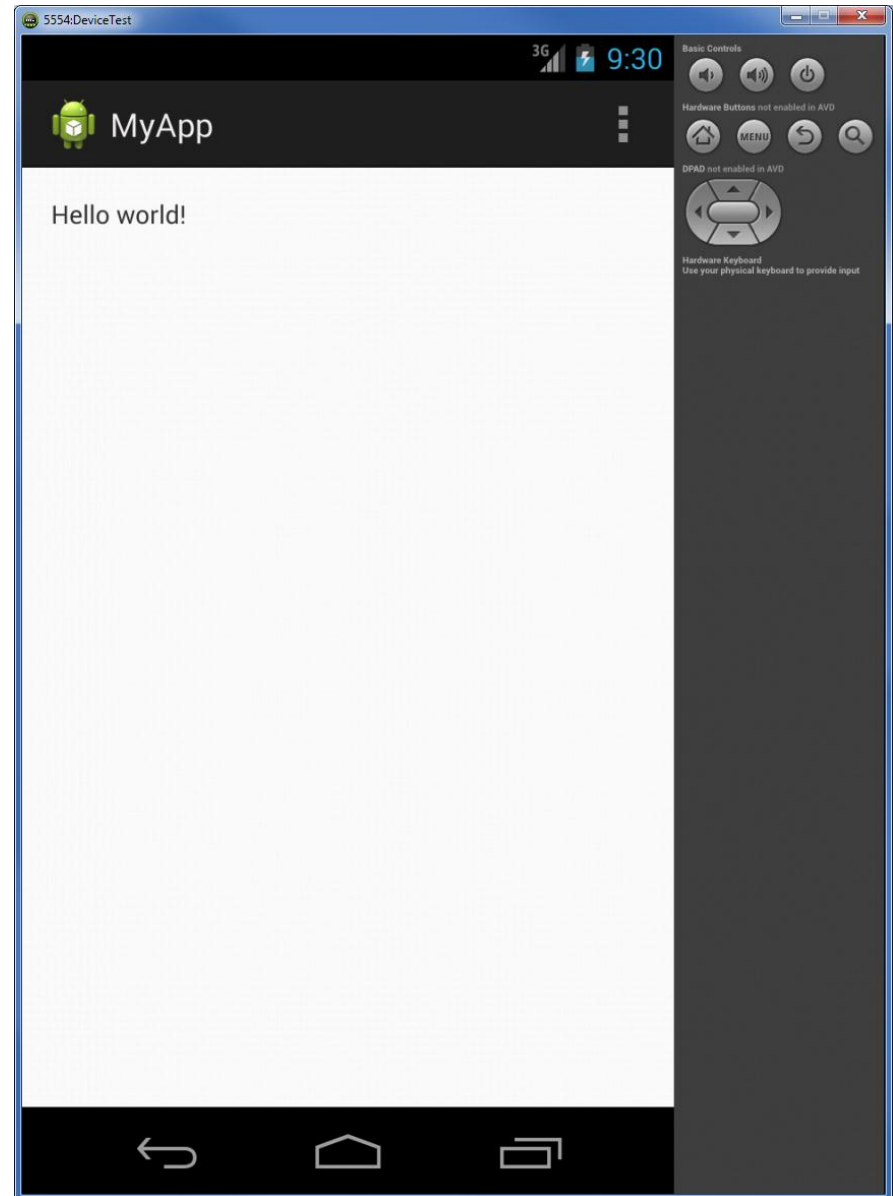
DEPLOIEMENT

- Run As -> Android Application
- Par défaut, utilise le virtual device



DEPLOIEMENT

L'émulateur ne doit être démarré qu'une seule fois. Il suffit ensuite de le laisser en arrière plan ; il sera automatiquement réutilisé pour les futures exécutions.

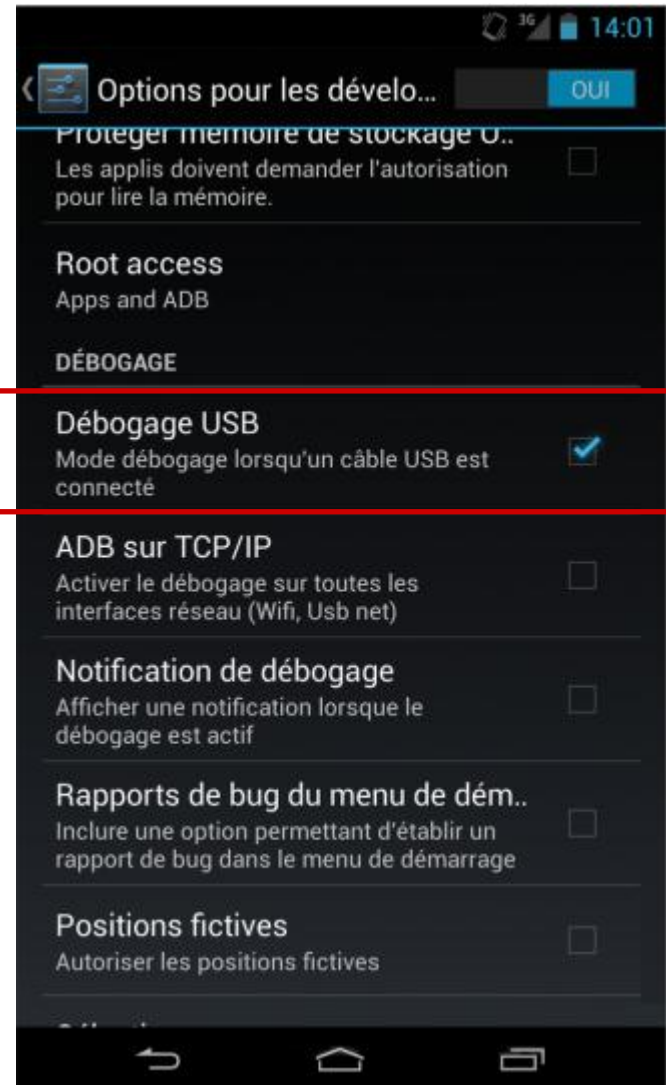
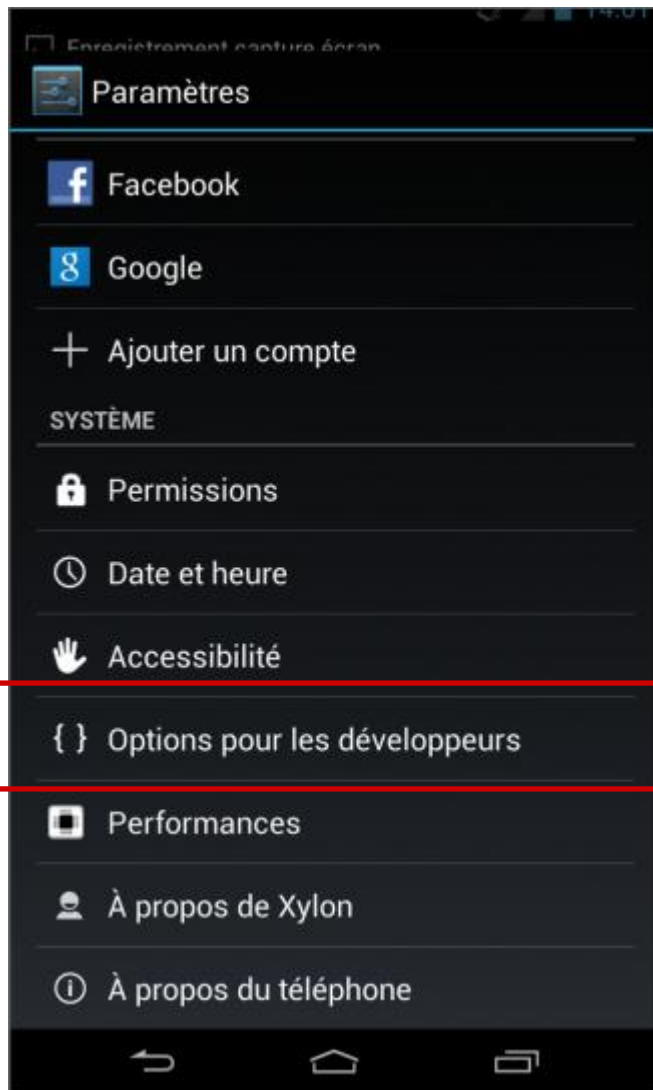


UTILISER UN VRAI DEVICE

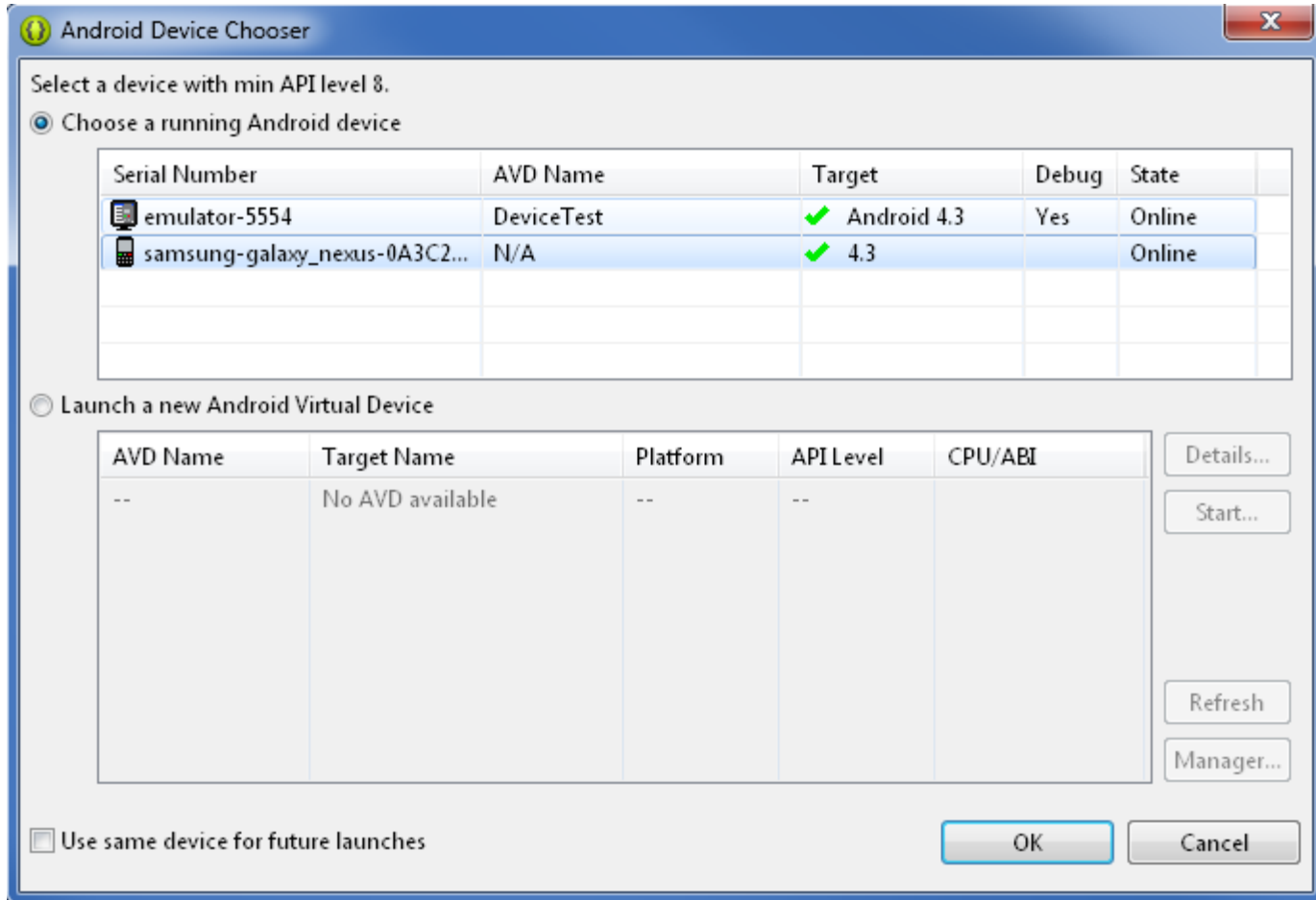
- Passer le device en mode debug USB.
- Depuis Android 4.2, les options pour développeur doivent être réactivées en premier lieu :
 1. Aller dans le menu Paramètres/A propos.
 2. Taper sept fois sur le numéro de build.
 3. ~~Tourner huit fois sur vous-même en brandissant un crucifix, puis...~~



UTILISER UN VRAI DEVICE



UTILISER UN VRAI DEVICE



ANDROID DEBUG BRIDGE

- ADB est un outil de débogage en trois parties :
 - Sur la machine du développeur :
 - Un client, utilisable en ligne de commande.
 - Un serveur (port 5037), qui gère la communication avec les différents appareils.
 - Un démon qui tourne sur chaque appareil (ports 5554 – 55585).
- L'exécutable adb se situe dans `<sdk>/platform-tools`.

ANDROID DEBUG BRIDGE

```
> adb devices
```

```
List of devices attached
```

```
0A3C23C719015019      device
emulator-5554         offline
emulator-5556         device
```

```
> adb -s emulator-5556 install helloWorld.apk
```

```
> adb -s emulator-5556 push foo.txt /sdcard/foo.txt
```

```
> adb -s emulator-5556 pull /sdcard/foo.txt foo.txt
```

```
> adb -s emulator-5554 shell
```

```
shell@maguro:/ $ sqlite3 /data/data/fr.inria.example/databases/somedb.db
```

```
> adb -s emulator-5556 shell screenrecord /sdcard/demo.mp4
```

LOGCAT

- Logcat permet d'accéder aux différents log enregistrés par les appareils.

```
> adb -s emulator-5554 logcat
11-26 05:46:27.260: D/AlertService(649): No fired or scheduled alerts
11-26 05:46:27.300: D/AlertService(649): Scheduling next alarm with AlarmScheduler.
sEventReminderReceived: null
11-26 05:46:27.330: D/AlarmScheduler(649): No events found starting within 1 week.
11-26 05:46:28.450: D/dalvikvm(279): GC_EXPLICIT freed 383K, 61% free 5334K/13492K, paused
5ms+14ms, total 155ms
...
```

Search for messages. Accepts Java regexes. Prefix with pid, app, tag; or text: to limit scope. verbose 📄 🔍 📄 ⏴ ⏵

Level	Time	PID	TID	Application	Tag	Text
W	11-26 05:46:2...	912	912	com.android...	dalvikvm	kernel is compiled with file capabilities support enabled PR_CAPSET_DROP 33 failed: Invalid argument. Please make kernel is compiled with file capabilities support enabled
I	11-26 05:46:2...	279	472	system_process	ActivityManager	Start proc com.android.quicksearchbox for broadcast com.a cksearchbox/.CorporaUpdateReceiver: pid=912 uid=10033 gid 3003, 1028}
D	11-26 05:46:2...	749	752	com.android...	dalvikvm	GC_CONCURRENT freed 205K, 12% free 2700K/3064K, paused 35 tal 645ms
V	11-26 05:46:2...	875	875	com.android...	SmsReceiverService	onStart: #2 mResultCode: -1 = Activity.RESULT_OK
D	11-26 05:46:2...	426	429	com.android...	dalvikvm	GC_CONCURRENT freed 322K, 16% free 3140K/3712K, paused 73 total 222ms
D	11-26 05:46:2...	426	429	com.android...	dalvikvm	GC_EXPLICIT freed 33K, 10% free 3085K/3210K, paused 5ms

CONSOLE DE L'ÉMULATEUR

- L'émulateur peut être manipulé avec une console très polyvalente, directement accessible depuis telnet.

```
> telnet localhost 5554
Android Console: type 'help' for a list of commands
OK
```

```
power capacity 75
power status charging
```

```
gsm call 012041293123
sms send 12345 Will be home soon
```

```
# Longitude Latitude <altitude>
geo fix 2.0983248486704418 48.836726551297495 120
```

```
network delay gprs
network delay 50 100
network speed gprs
```

DALVIK DEBUG MONITOR SERVER

- DDMS est un outil de débogage (graphique), basé sur ADB, Logcat, etc.
 - Mesurer l'utilisation de la mémoire (allocation, heap).
 - Monitorer les threads et les processus.
 - Monitorer le trafic réseau.
 - Simuler le GPS, les sms/appels et les différents changements d'état dans l'émulateur.
 - Gérer les redirections de port dans l'émulateur.
 - Analyser les logs.
 - Envoyer/récupérer des fichiers.
 - ...

DALVIK DEBUG MONITOR SERVER

The screenshot displays the DDMS interface within an IDE. The title bar reads "DDMS - SampleApp/src/fr/inria/sampleapp/MainActivity.java - ADT". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar contains various icons for file operations and debugging. On the left, the "Devices" pane shows a list of running processes, including "emulator-5554" and "samsung-galaxy_nexus-0A3C23C". The main pane displays a "Threads" table with columns for ID, Tid, Status, utime, stime, and Name. Thread 12, "AsyncTask #2", is selected. Below the table, a "Refresh" button and a timestamp "Tue Nov 26 13:27:16 CET 2013" are visible. The stack trace for the selected thread is shown in a text area.

ID	Tid	Status	utime	stime	Name
*5	25204	VmWait	20	8	Compiler
*6	25205	Wait	0	0	ReferenceQueueDaemon
*7	25206	Wait	0	0	FinalizerDaemon
*8	25207	Wait	0	0	FinalizerWatchdogDaemon
9	25210	Native	0	0	Binder_1
10	25214	Native	0	0	Binder_2
11	25250	Wait	1	0	AsyncTask #1
12	25453	Wait	1	0	AsyncTask #2
13	25454	Wait	0	3	AsyncTask #3
14	25455	Wait	0	0	AsyncTask #4
15	25456	Wait	0	0	AsyncTask #5

Refresh Tue Nov 26 13:27:16 CET 2013

```
at java.lang.Object.wait(Native Method)
at java.lang.Thread.parkFor(Thread.java:1205)
at sun.misc.Unsafe.park(Unsafe.java:325)
at java.util.concurrent.locks.LockSupport.park(LockSupport.java:159)
at java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject.await(AbstractQ...
at java.util.concurrent.LinkedBlockingQueue.take(LinkedBlockingQueue.java:413)
at java.util.concurrent.ThreadPoolExecutor.getTask(ThreadPoolExecutor.java:1013)
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1073)
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:573)
at java.lang.Thread.run(Thread.java:841)
```

DALVIK DEBUG MONITOR SERVER

DDMS - SampleApp/src/fr/inria/sampleapp/MainActivity.java - ADT

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access Java DDMS

Devices

Name

Name	State
emulator-5554	Online
system_process	289
android.process.media	370
com.android.inputmethod.la	391
com.android.phone	410
com.android.launcher	423
com.android.settings	444
android.process.acore	471
com.android.music	482
com.android.defcontainer	515
com.android.exchange	559
com.android.calendar	572
com.android.email	595
com.android.providers.calenc	625
com.android.systemui	639
com.android.mms	652
com.android.deskclock	691
com.svox.pico	739
com.android.quicksearchbox	756
fr.inria.sampleapp	785
samsung-galaxy_nexus-0A3C23C	Online
fr.inria.sampleapp	25197

Threads Heap Allocation T... Network Stat... File Explorer Emulator Co... System Infor...

Speed: Fast (100ms) Start Reset

6.8KB/s
5.9KB/s
4.9KB/s
3.9KB/s
2.9KB/s
2KB/s
1000B/s
0B/s
1000B/s
2KB/s

13:25:00 13:25:05 13:25:10 13:25:15 13:25:20 13:25:25

Tag	RX bytes	RX packets	TX bytes	TX packets
Total	5 826	14	2 035	19

LogCat

DALVIK DEBUG MONITOR SERVER

DDMS - SampleApp/src/fr/inria/sampleapp/MainActivity.java - ADT

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access Java DDMS

Devices

Name	Online
emulator-5554	Online
system_process	289
android.process.media	370
com.android.inputmethod.la	391
com.android.phone	410
com.android.launcher	423
com.android.settings	444
android.process.acore	471
com.android.music	482
com.android.defcontainer	515
com.android.exchange	559
com.android.calendar	572
com.android.email	595
com.android.providers.calenc	625
com.android.systemui	639
com.android.mms	652
com.android.deskclock	691
com.svox.pico	739
com.android.quicksearchbox	756
fr.inria.sampleapp	785
samsung-galaxy_nexus-0A3C23C	Online
fr.inria.sampleapp	25197

Threads Heap Allocation T... Network Stat... File Explorer Emulator Co... System Infor...

Telephony Status

Voice: home Speed: Full

Data: home Latency: None

Telephony Actions

Incoming number:

Voice

SMS

Message:

Call Hang Up

Location Controls

Manual GPX KML

Decimal

Sexagesimal

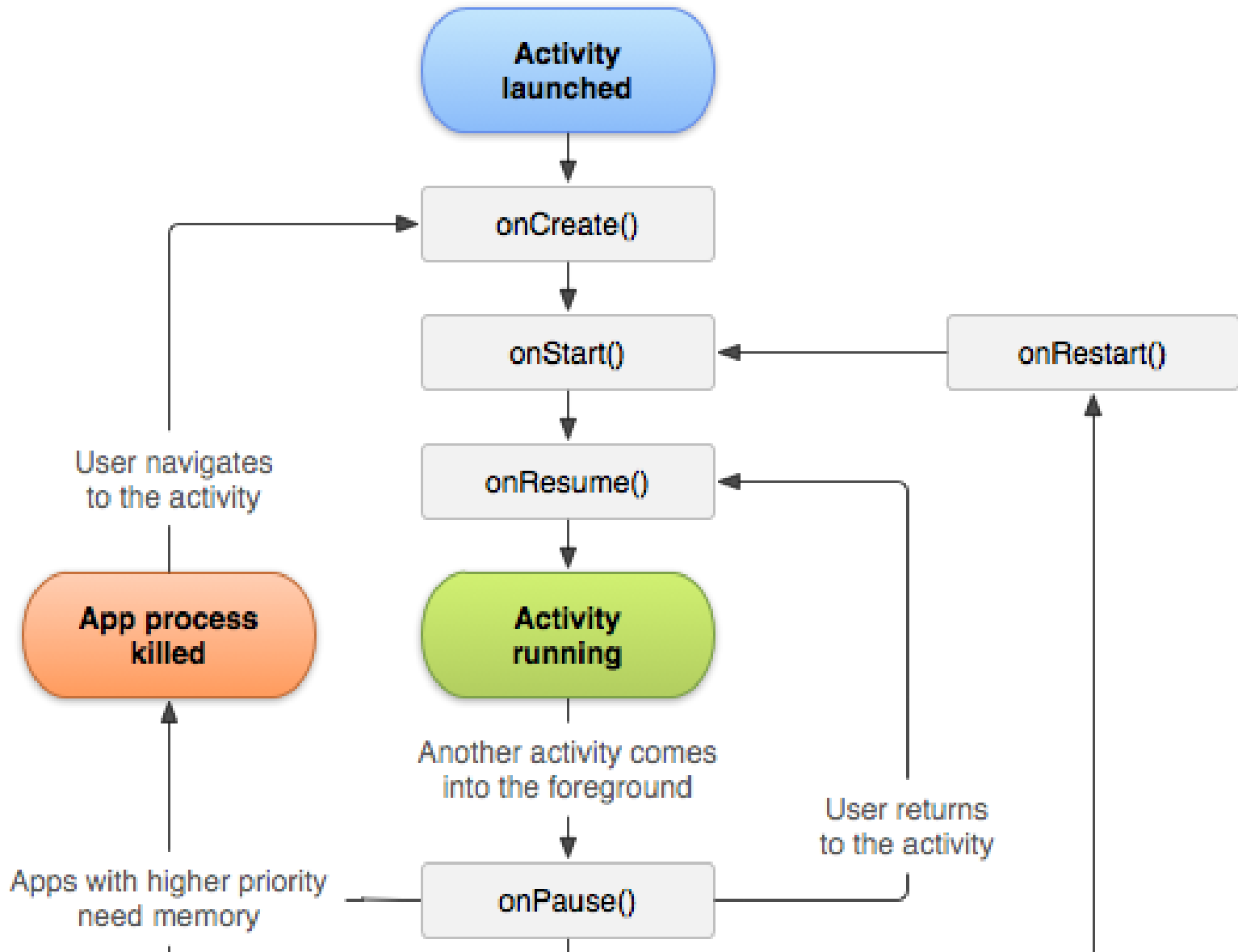
Longitude -122,084095

Latitude 37,422006

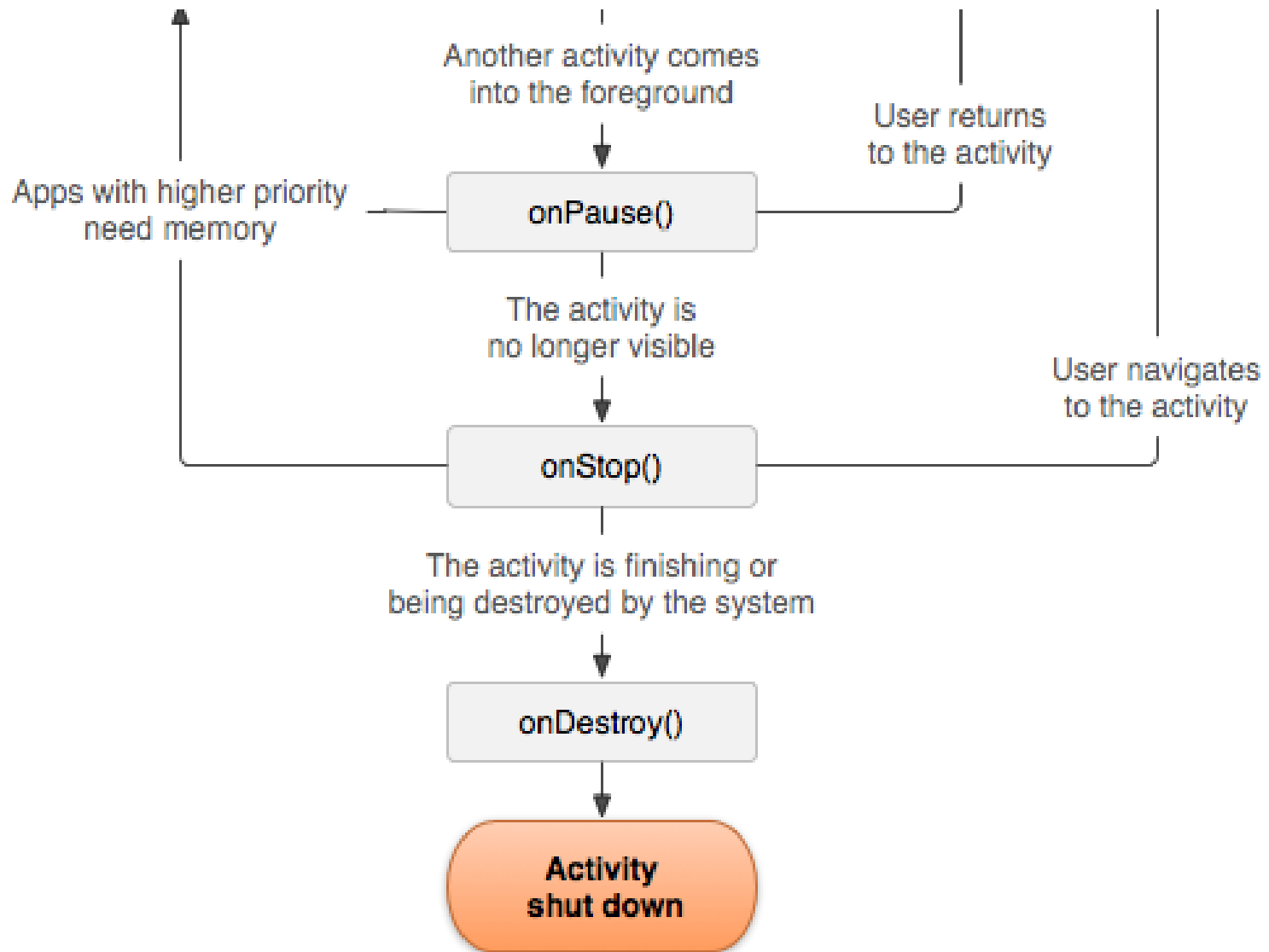
Send

LogCat Console

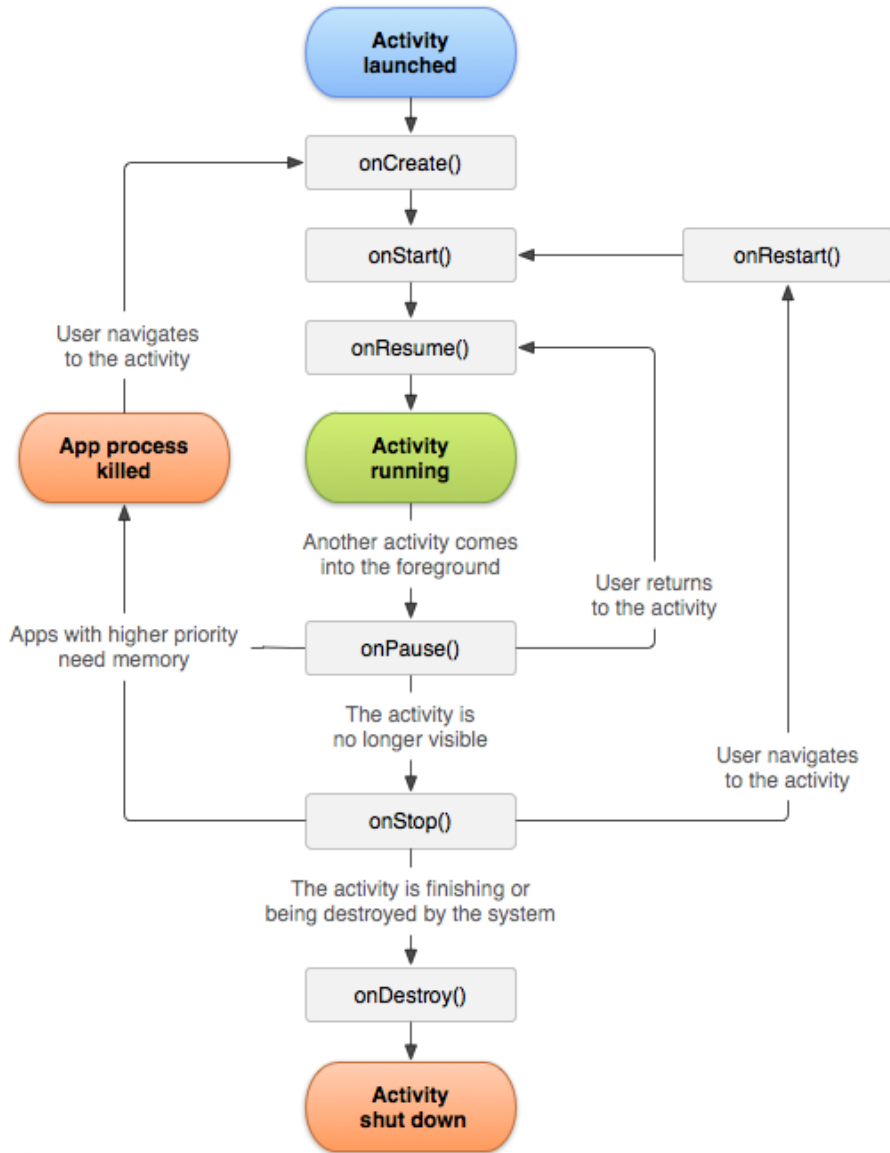
LE COMPOSANT ACTIVITY



LE COMPOSANT ACTIVITY



LE COMPOSANT ACTIVITY



- Durée de vie totale : de `onCreate()` à `onDestroy()`
- Arrière plan : de `onStart()` à `onStop()`, l'application s'exécute en fond et n'est plus visible par l'utilisateur.
- Premier plan : de `onResume()` à `onPause()`, l'application est au premier-plan et l'utilisateur interagit avec celle-ci.

INTENT MESSAGING

- L'échange de message entre les composants applicatifs se fait au travers de messages, appelés Intent.
- Un Intent est une structure de données qui décrit soit une opération à effectuer, soit un évènement.

ComponentName	La classe et le package d'un composant.
Action	Une action qui doit être, ou a été, réalisée.
Data	Une URI (tel:, http:, geo:, ...) vers des données à traiter, et un type MIME.
Category	Une catégorie de composant.
Extras	Des paramètres (clé/valeur).
Flags	Options diverses.

INTENT MESSAGING

- Android route les Intent d'un composant à un autre en exploitant les informations fournies dans le manifest.
- Un Intent qui définit un ComponentName est directement transmis (intent explicite).
- Les autres Intent (implicites) nécessitent de trouver les composants qui matchent les différents champs, avec trois issues possibles :
 - Un seul composant a été trouvé, le message est transmis.
 - Plusieurs composants ont été trouvés, une boîte de dialogue demande alors à l'utilisateur de faire un choix.
 - Aucun composant n'a été trouvé, une exception est levée.

UN SIMPLE INTENT EXPLICITE

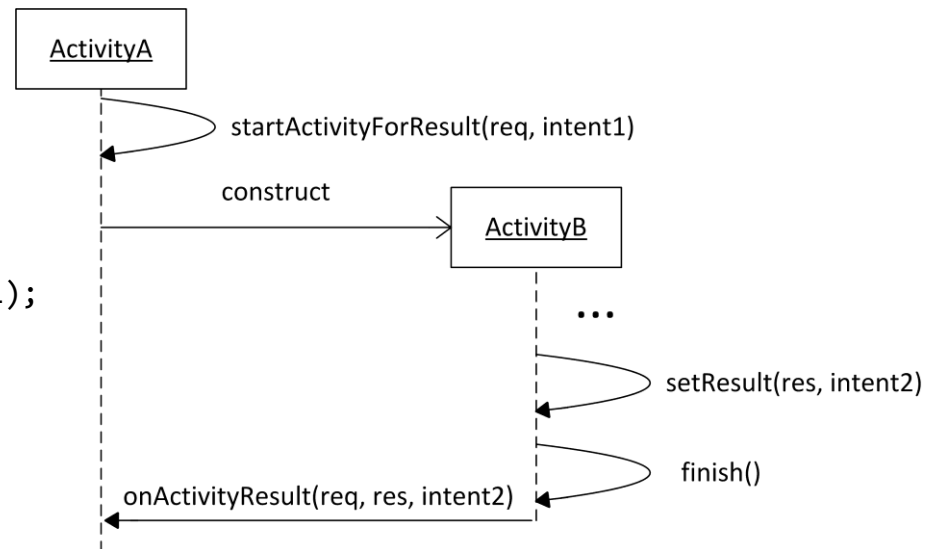
ActivityA

```
Intent i = new Intent(this, ActivityB.class);
i.putExtra("something", 10);
startActivityForResult(i, 42);
```

```
@Override
public void onActivityResult(int requestCode, int resultCode, Intent data)
{
    super.onActivityResult(requestCode, resultCode, data);
    if(requestCode == 42 && resultCode == Activity.RESULT_OK)
        int val = data.getExtras().getInt("somethingPlusOne");
}
```

ActivityB

```
Intent request = getIntent();
if(request.hasExtra("something"))
{
    int val = request.getIntExtra("something");
    Intent response = new Intent();
    response.putExtra("somethingPlusOne", val + 1);
    setResult(Activity.RESULT_OK, response);
}
else
    setResult(1); // error
finish();
```



INTENT RESOLUTION

- Seuls les champs action, data (URI, type MIME) et category sont utilisés pour le matching.
- Lorsqu'un composant est évalué, celui-ci doit passer trois tests pour être considéré comme un destinataire potentiel :
 - Action test.
 - Category test.
 - Data test.

ACTION TEST

Component

Intent		No action	Set of actions A
	No action	Failure	Success
	Set of actions B	Failure	Success if A and B shares at least one element : $ A \cap B \geq 1$

CATEGORY TEST

Component

	No category	Set of categories A
Intent	Success	Success
Set of categories B	Failure	Success if every category of A is defined in B : $A \subseteq B$

- Android considère que tous les Intents implicites passés à `startActivity()` appartiennent à une catégorie “default” ...
- ... il faut donc l’ajouter dans le manifest si l’on désire que l’activité puisse recevoir les intents...
- ... sauf pour les activités qui définissent l’action “main” ou la catégorie “launcher”.

LES INTENT-FILTERS

```
<intent-filter ... >
  <action android:name="com.example.project.SHOW_CURRENT" />
  <action android:name="com.example.project.SHOW_RECENT" />
  <action android:name="com.example.project.SHOW_PENDING" />
  ...
  <category android:name="android.intent.category.DEFAULT" />
  <category android:name="android.intent.category.BROWSABLE" />
  ...
  <data android:mimeType="video/mpeg" android:scheme="http" ... />
  <data android:mimeType="audio/mpeg" android:scheme="http" ... />
  ...
</intent-filter>
<intent-filter ... >
  ...
</intent-filter>
<intent-filter ... >
  ...
</intent-filter>
```

- Un Intent doit matcher au moins un filtre.
- Les étiquettes de catégorie/action prédéfinies incluent le nom de package complet :

Intent.CATEGORY_DEFAULT = android.intent.category.DEFAULT

QUELQUES INTENT IMPLICITES

```
// exemple 1
```

```
Intent i = new Intent(Intent.ACTION_VIEW, Uri.parse("http://inria.fr"));
startActivity(i);
```

```
// exemple 2
```

```
Intent i = new Intent();
i.setAction(Intent.ACTION_CALL);
i.setData(Uri.parse("tel:0102030405"));
startActivity(i);
```

```
// exemple 3
```

```
Intent i = new Intent(Intent.ACTION_PICK, ContactsContract.Contacts.CONTENT_URI);
startActivityForResult(i, 42);
```

```
@Override
```

```
protected void onActivityResult(int requestCode, int resultCode, Intent data)
{
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == 42 && resultCode == Activity.RESULT_OK)
    {
        Uri contactData = data.getData();
        ...
    }
}
```