

# Naviguer entre des écrans

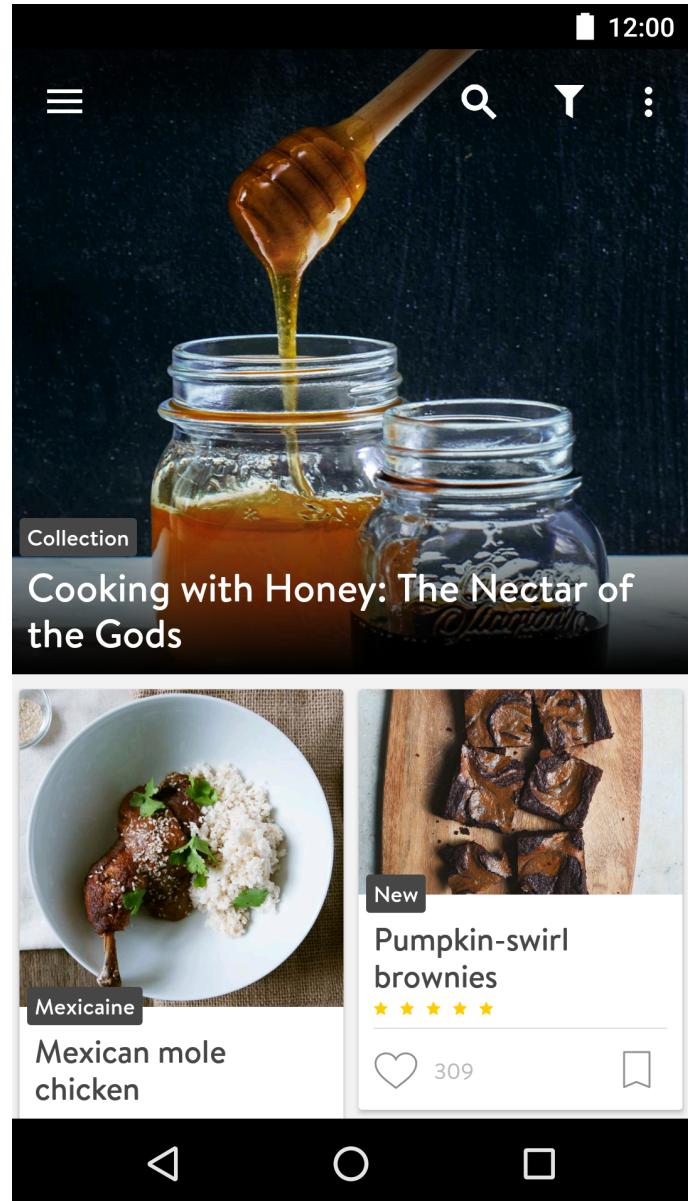
Adrian Bracigliano

# Objectif

- Intéragir avec une interface graphique
- Lancer et enchaîner des écrans

# Activity

- Ecran = Activity
- Un des composants de base dans Android
- Classe Java située dans dossier src/main/java



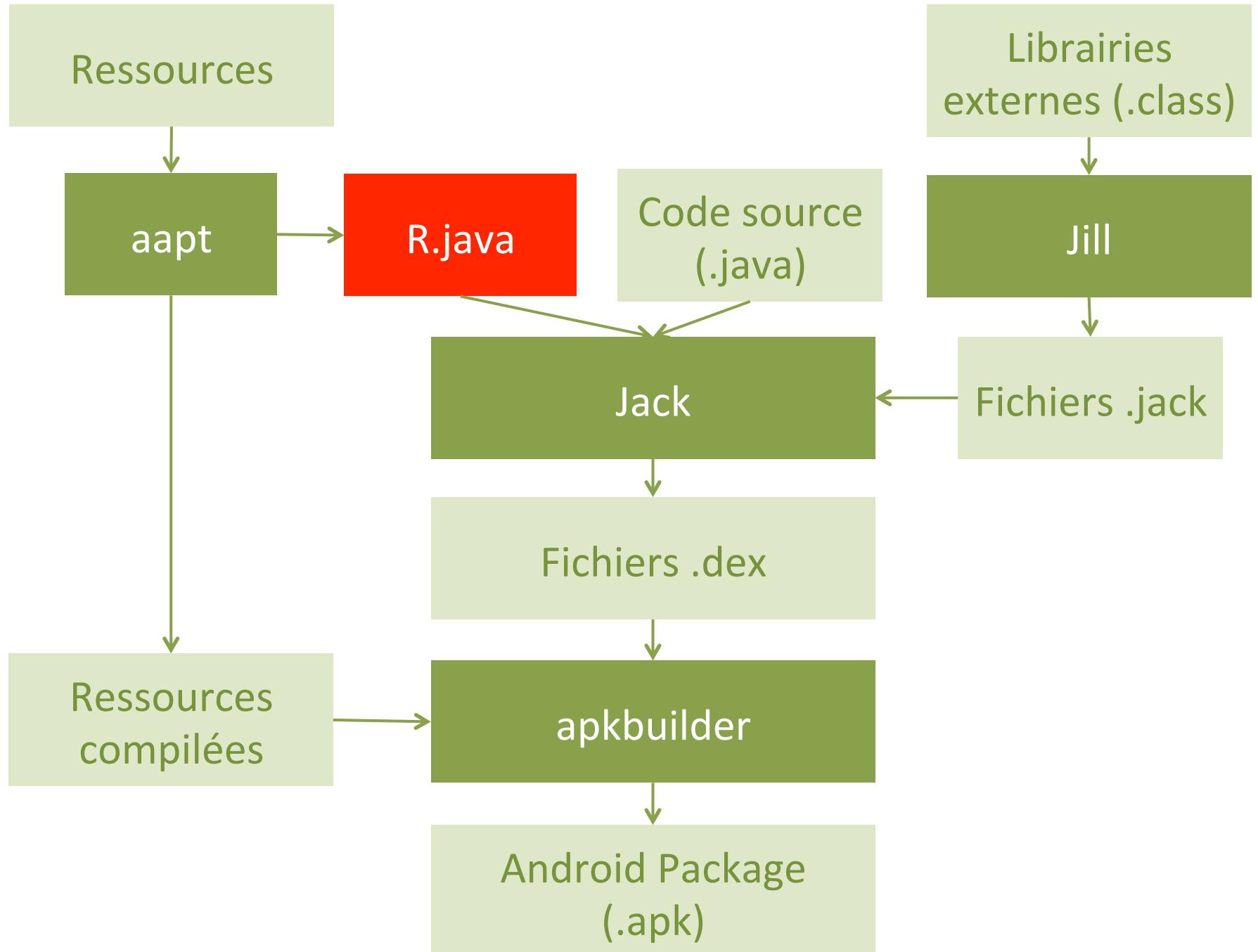
Kitchen Stories - Liste de recettes

# Activity

```
public class MyActivity extends Activity {  
  
    // Called when the Activity is created  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.my_layout);  
    }  
}
```

# Activity

```
public class MyActivity extends Activity {  
  
    // Called when the Activity is created  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.my_layout);  
    }  
}
```



# Fichier R.class

```
public final class R {  
    public static final class layout {  
        public static final int my_layout=0x7f04001a;  
        ...  
    }  
    public static final class id { ... }  
    public static final class drawable { ... }  
    public static final class dimen { ... }  
    public static final class color { ... }  
    public static final class string { ... }  
}
```

# Fichier my\_layout.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:orientation="vertical">  
  
    <TextView  
        android:id="@+id/textview"  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content" />  
  
    <EditText  
        android:id="@+id/edittext"  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content" />  
  
    <Button  
        android:id="@+id/button"  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content" />  
  
</LinearLayout>
```

# Fichier R.class

```
public final class R {  
    public static final class layout { ... }  
    public static final class id {  
        ...  
        public static final int button=0x7f0b0056;  
        public static final int edittext=0x7f0b0055;  
        public static final int textView=0x7f0b0054;  
        ...  
    }  
    public static final class drawable { ... }  
    public static final class dimen { ... }  
    public static final class color { ... }  
    public static final class string { ... }  
}
```

# Communication avec la vue

```
public class MyActivity extends Activity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.my_layout_activity);  
  
        TextView title = (TextView) findViewById(R.id.textview);  
        EditText password = (EditText) findViewById(R.id.edittext);  
        Button validate = (Button) findViewById(R.id.button);  
  
        String text = getResources().getString(R.string.title);  
        int color = getResources().getColor(R.color.color);  
        int dimen = getResources().getDimension(R.dimen.dimen);  
    }  
}
```

# Communication avec la vue

```
public class MyActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        ...

        title.setText(text);
        title.setTextColor(color);
        password.setInputType(InputType.TYPE_TEXT_VARIATION_PASSWORD);
        validate.setOnClickListener(clickListener);
    }

    private View.OnClickListener clickListener = new View.OnClickListener(){
        @Override
        public void onClick(View v) {
            //Handle event
        }
    };
}
```

# AndroidManifest.xml

- Fichier xml situé dans src/main/

# AndroidManifest.xml

- Permet de déclarer :
  - les Activities

```
<manifest package="com.ensiie" ...>  
  
    <application ...>  
        <activity android:name="com.ensiie.MyActivity">  
            <intent-filter>  
                <action android:name="android.intent.action.MAIN" />  
                <category android:name="android.intent.category.LAUNCHER" />  
            </intent-filter>  
        </activity>  
    </application>  
  
</manifest>
```

Declaration de l'Activity MyActivity

# AndroidManifest.xml

- Permet de déclarer :
  - les Activities

```
<manifest package="com.ensiie" ...>

    <application ...>
        <activity android:name="com.ensiie.MyActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Declaration de l'Activity MyActivity comme Activity principale de l'application

# AndroidManifest.xml

- Permet de déclarer :
  - le package de l'application

```
<manifest package="com.soundcloud.android" ...>  
  ...  
</manifest >
```

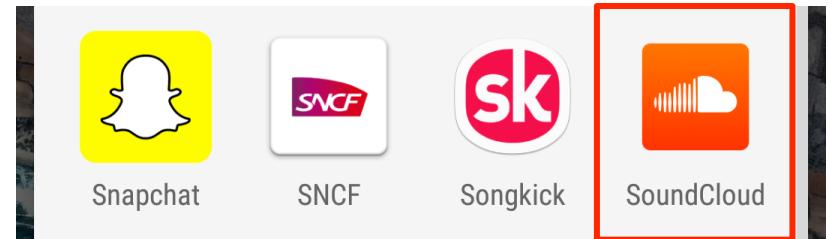
 <https://play.google.com/store/apps/details?id=com.soundcloud.android>

Url d'accès à l'application SoundCloud

# AndroidManifest.xml

- Permet de déclarer :
  - l'icône et le nom de l'application

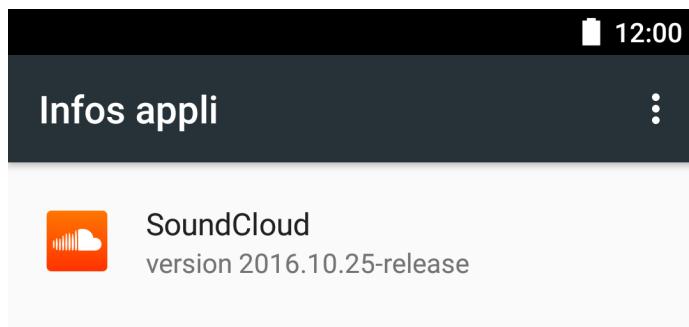
```
<manifest package="com.soundcloud.android" ...>
    <application
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name">
        ...
    </application >
</manifest >
```



Déclaration du nom et de l'icône de l'application SoundCloud

# AndroidManifest.xml

- Permet de déclarer :
  - le `versionCode` : Nombre utilisé en interne par le système pour déterminer si une version d'application est plus récente qu'une autre
  - le `versionName` : Nom de version affiché à l'utilisateur



Affichage du `versionName` dans les infos de l'application SoundCloud

INFORMATIONS COMPLÉMENTAIRES		
Mise à jour	Installations	Version actuelle
27 octobre 2016	Entre 100 000 000 et 500 000 000	2016.10.25-release
Nécessite Android	Classification du contenu :	Éléments interactifs
4.0 ou version ultérieure	Accord parental	Interaction entre utilisateurs

Affichage du `versionName` de l'application SoundCloud sur la fiche info du store

# AndroidManifest.xml

- Permet de déclarer :
  - le `versionCode` : Nombre utilisé en interne par le système pour déterminer si une version d'application est plus récente qu'une autre
  - le `versionName` : Nom de version affiché à l'utilisateur

```
<manifest package="com.ensiie"
    android:versionCode="1"
    android:versionName="V1.0.0"
    ...>

    <application ... </application >

</manifest >
```

Déclaration du `versionCode` et du `versionName` d'une application

# AndroidManifest.xml

- Permet de déclarer :
  - minSdkVersion : Version d'Android minimum sur laquelle l'app peut être lancée
  - targetSdkVersion : Version d'Android avec laquelle l'app a été conçue

## INFORMATIONS COMPLÉMENTAIRES

Mise à jour  
27 octobre 2016

Installations  
Entre 100 000 000 et  
500 000 000

Version actuelle  
2016.10.25-release

Nécessite Android  
4.0 ou version ultérieure

Classification du  
contenu :  
Accord parental

Éléments interactifs  
Interaction entre  
utilisateurs

Affichage de la version minimum de l'application SoundCloud  
sur la fiche info sur store. minSdk à API level 14

# AndroidManifest.xml

- Permet de déclarer :
  - les permissions

```
<manifest
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission
        android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>

    <application> ... </application>
</manifest>
```

Declaration des permissions internet et écriture de fichiers

# Permissions

- Chaque application est sandboxée
  - Impossible d'impacter l'exécution d'autres applications
  - Impossible d'impacter le système d'exploitation
  - Impossible d'impacter l'utilisateur
- La déclaration de permissions permet de demander au système d'outrepasser les limites du sandbox

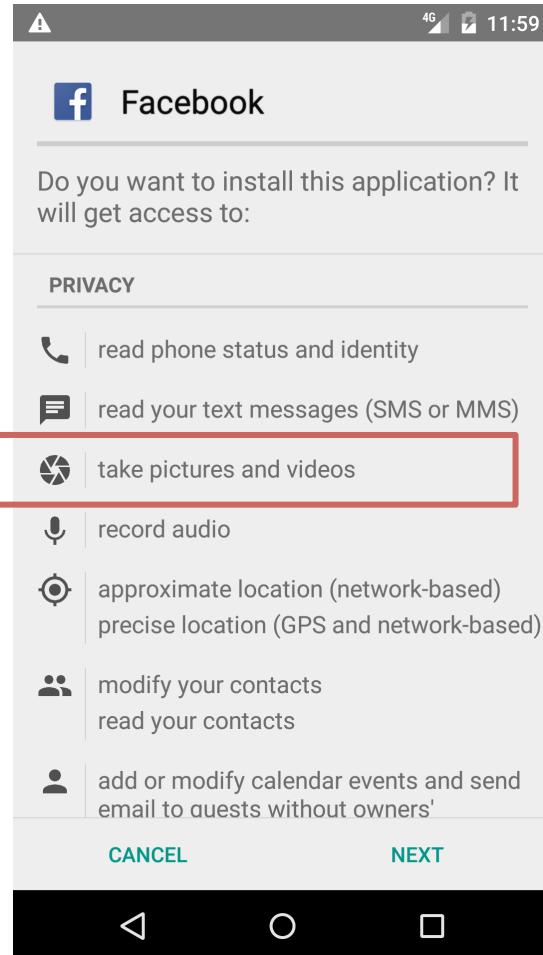
# Permissions normales

- Pas de risque pour la vie privée et la sécurité de l'utilisateur
- Automatiquement acceptées par le système
- Exemple :
  - INTERNET
  - USE\_FINGERPRINT
  - VIBRATE

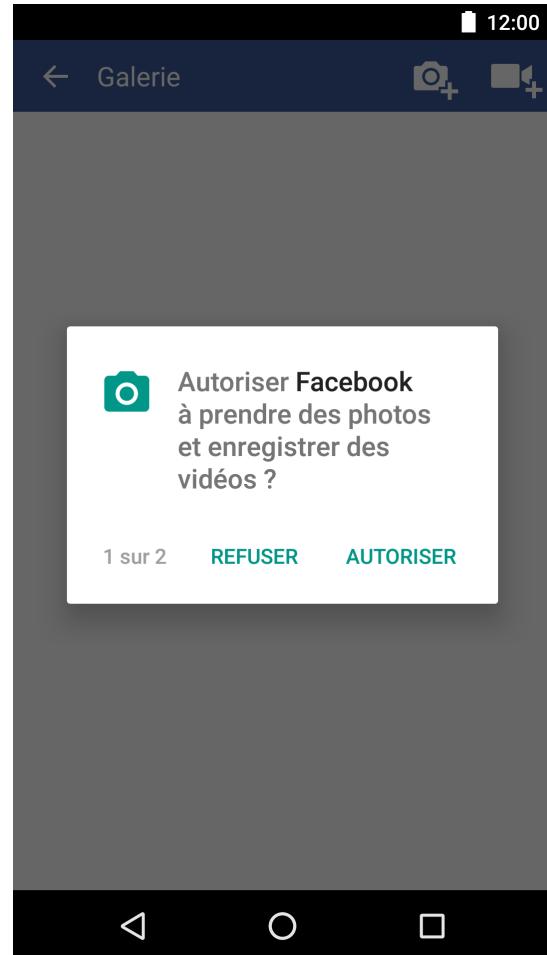
# Permissions dangereuses

- Possibilité d'affecter la vie privée de l'utilisateur, le stockage de ses données ou l'exécution d'autres applications
- Regroupées dans des groupes de permission
- Exemple :
  - **CONTACTS**  
READ\_CONTACTS  
WRITE\_CONTACTS
  - **STORAGE**  
READ\_EXTERNAL\_STORAGE  
WRITE\_EXTERNAL\_STORAGE

# Permissions dangereuses

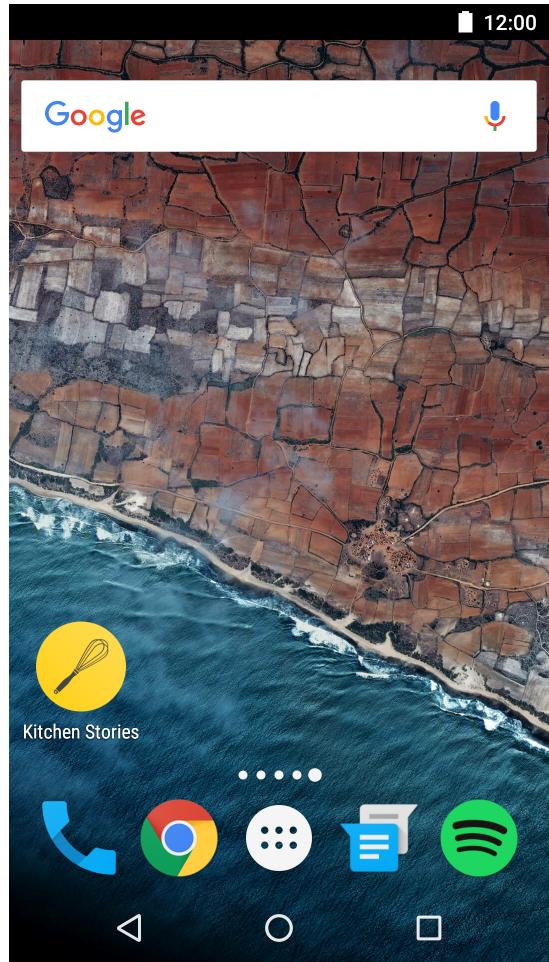


**Android ≤ 6.0 ou TargetSdkVersion ≤ 22**  
**Permission demandée avant de lancer l'app**



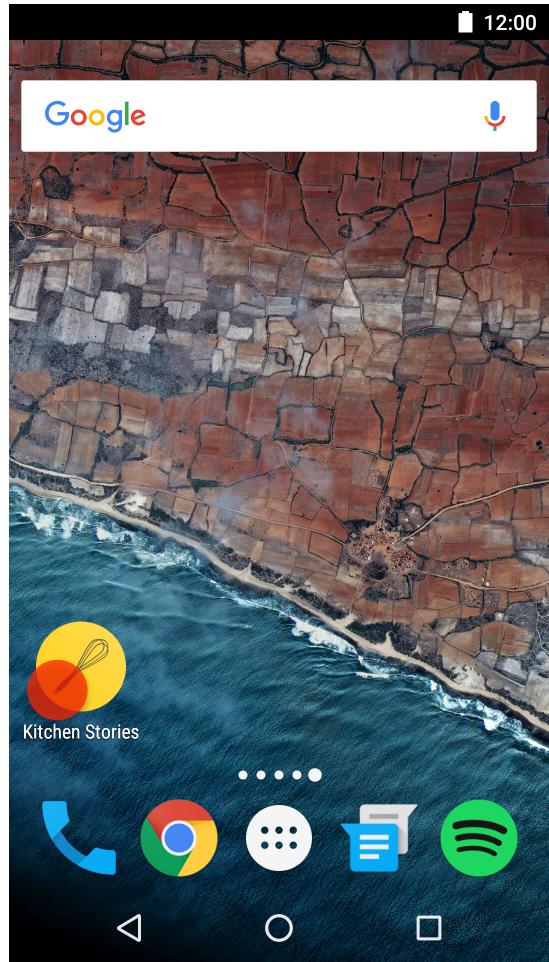
**Android ≥ 6.0 et TargetSdkVersion ≥ 23**  
**Permission demandée à partir du code quand nécessaire.**

# Lancement de l'application MyActivity



**Home**

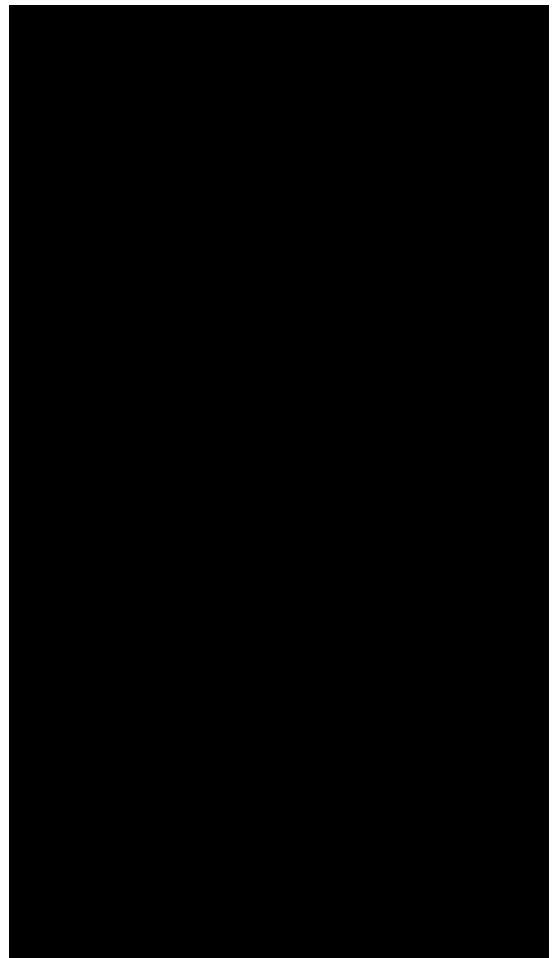
# Lancement de l'application MyActivity



Tap sur Kitchen Stories

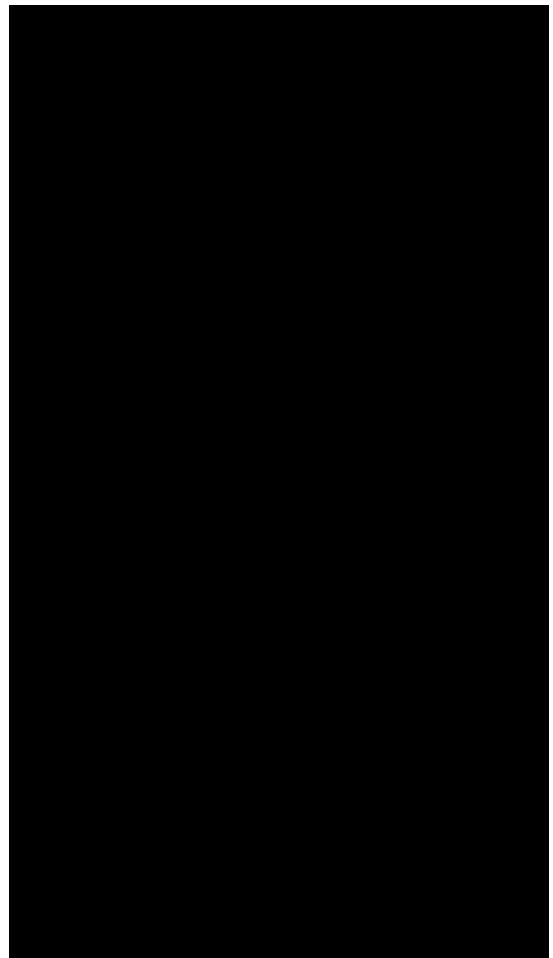
# Lancement de l'application MyActivity

1. Le système recherche dans le AndroidManifest.xml l'Activity avec l'action MAIN et category LAUNCHER



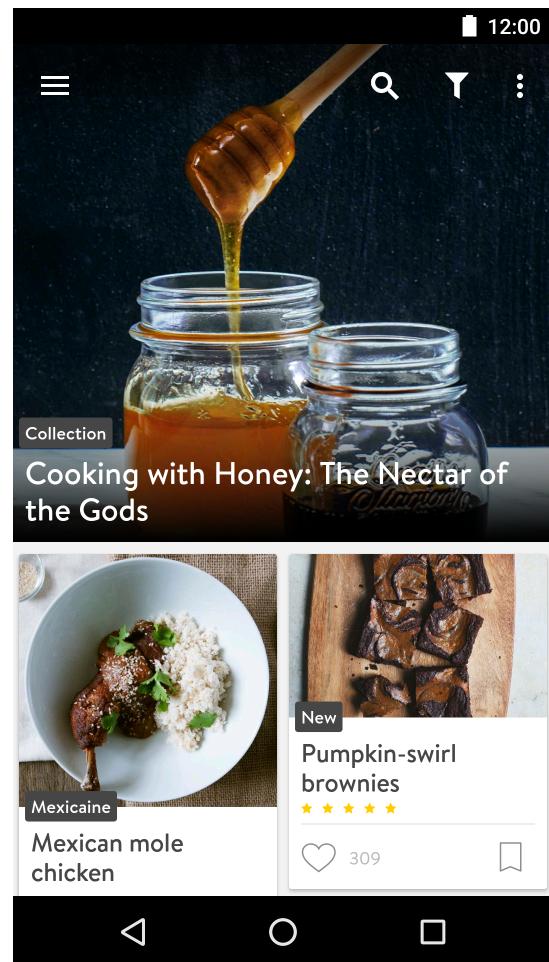
# Lancement de l'application MyActivity

1. Le système recherche dans le AndroidManifest.xml l'Activity avec l'action MAIN et category LAUNCHER
2. Activity MyActivity trouvée, appel par le système de la méthode onCreate() de la classe Activity



# Lancement de l'application MyActivity

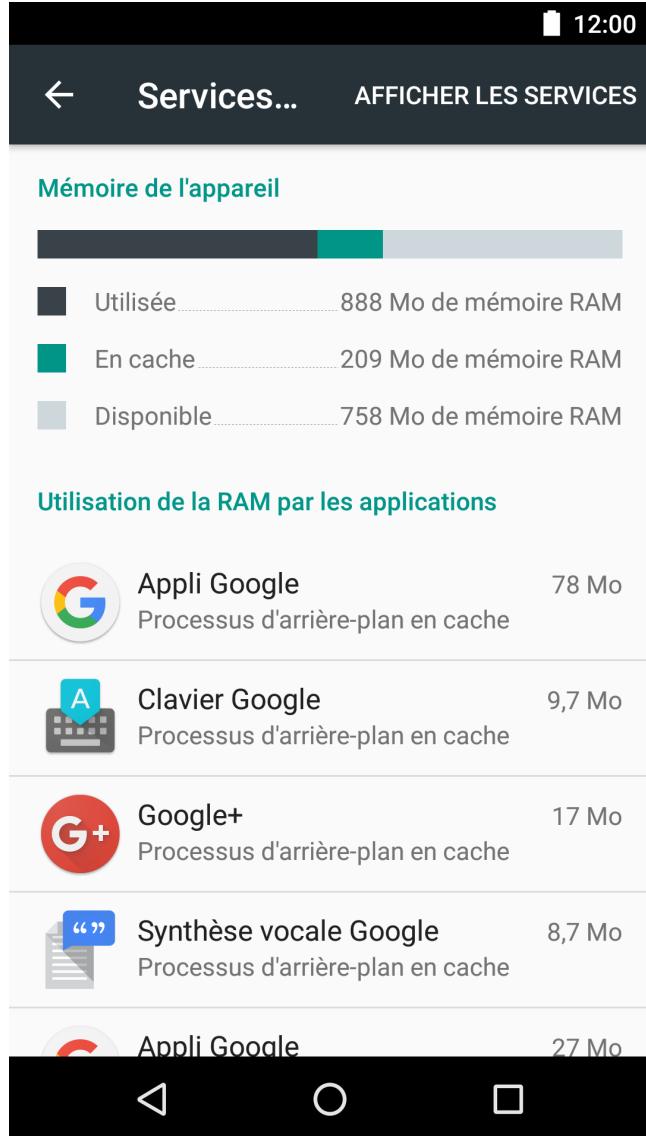
1. Le système recherche dans le AndroidManifest.xml l'Activity avec l'action MAIN et category LAUNCHER
2. Activity MyActivity trouvée, appel par le système de la méthode onCreate() de la classe Activity
3. Override de l'appel à onCreate() par MyActivity et appel à setContentView()



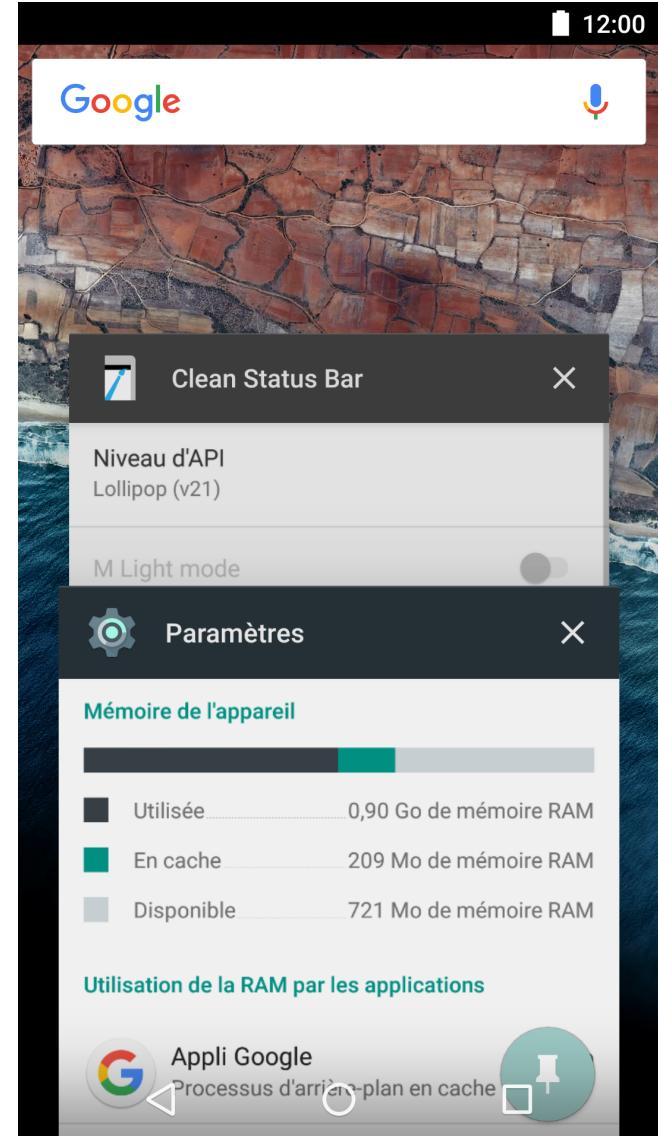
Affichage de l'Activity  
MyActivity

# Behind the scene

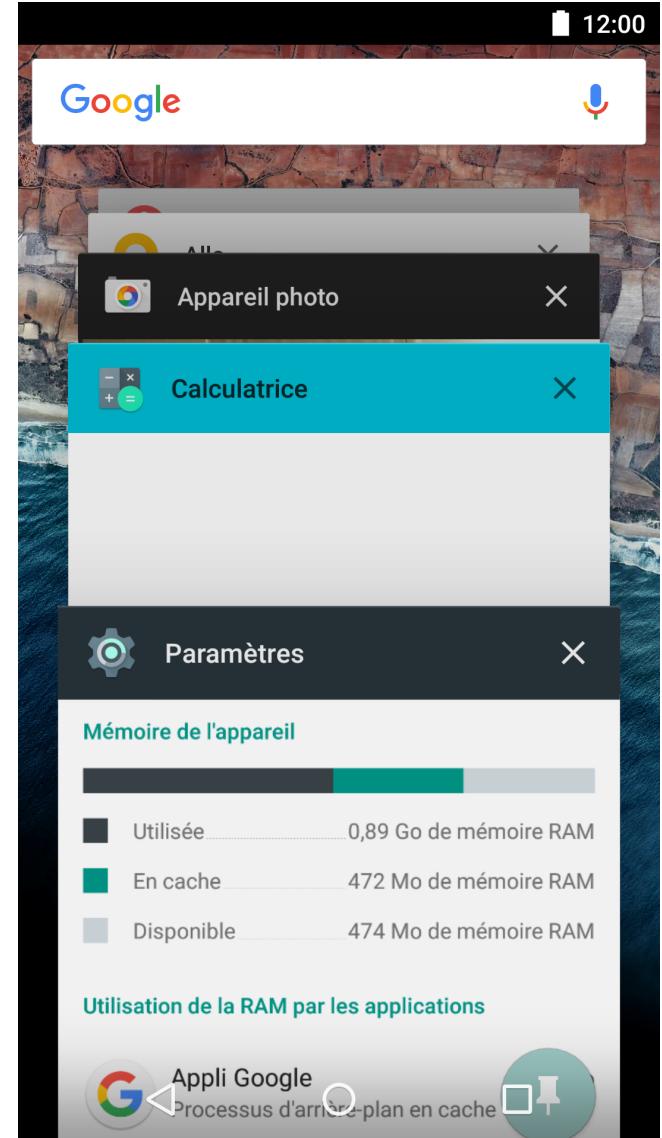
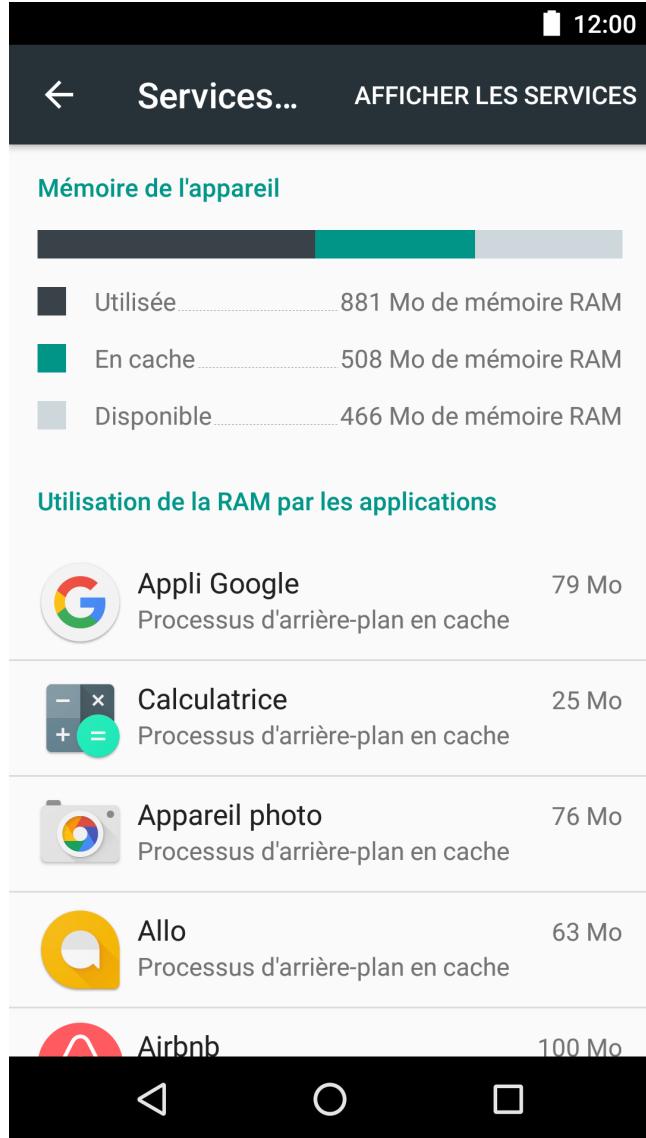
- Création d'un process Linux dédié lors du lancement de chaque application pour exécuter les Activities
- Allocation de RAM pour chaque classe, objet, etc... de chaque application
- Process gardé aussi longtemps qu'il reste de la mémoire

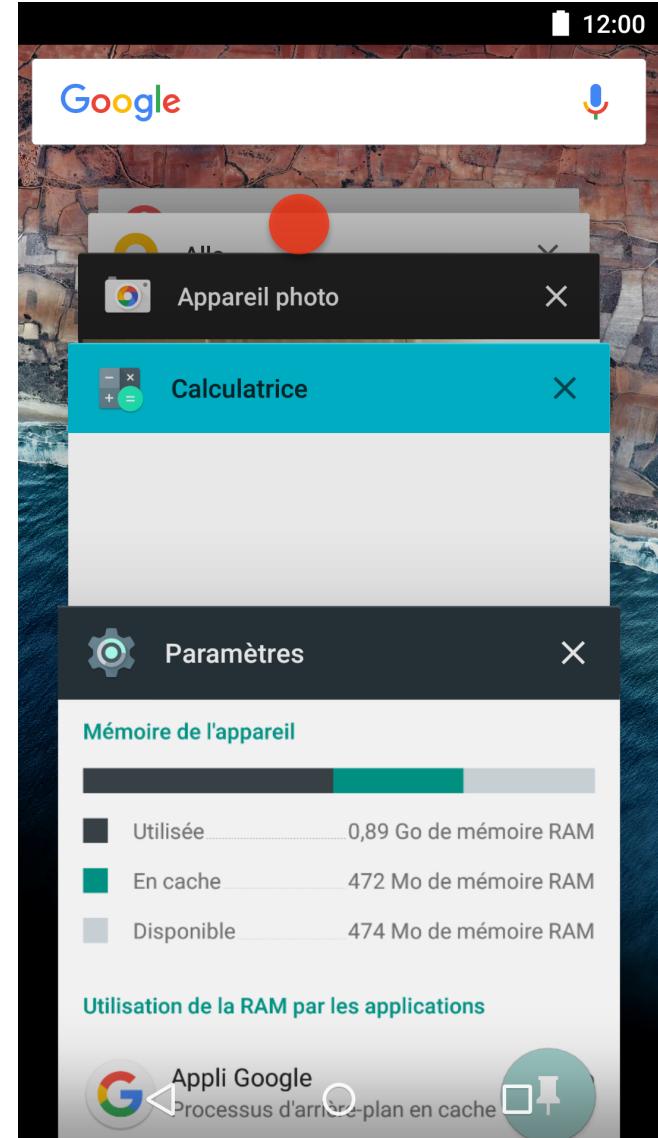
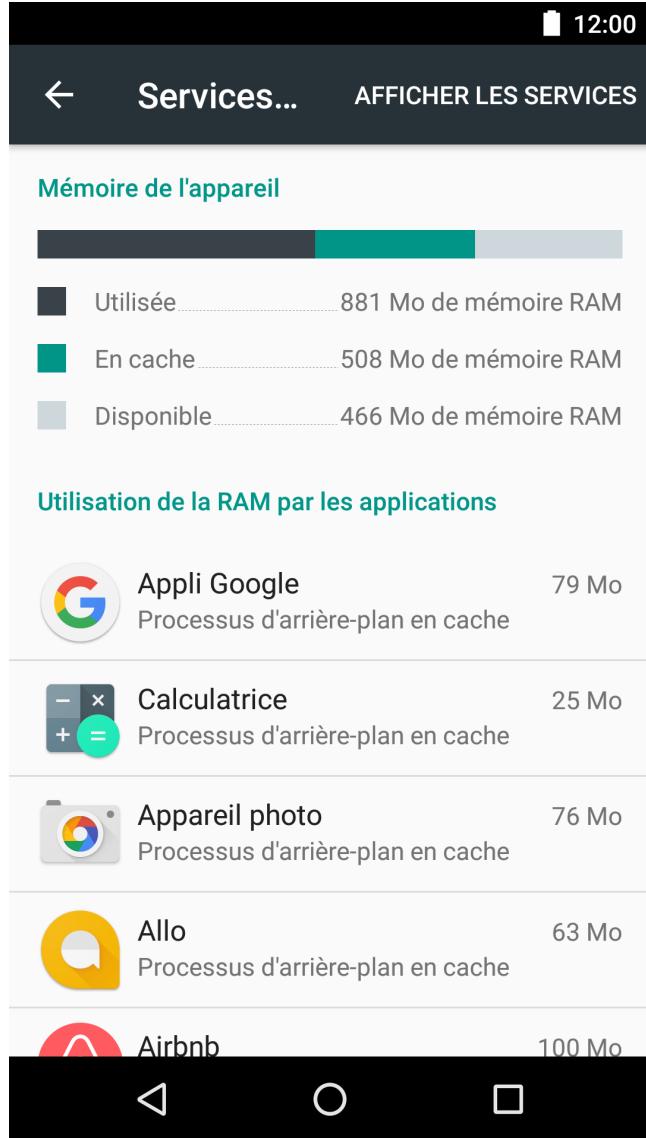


**Mémoire du téléphone  
(Options pour les développeurs >  
Services en cours d'exécution)**

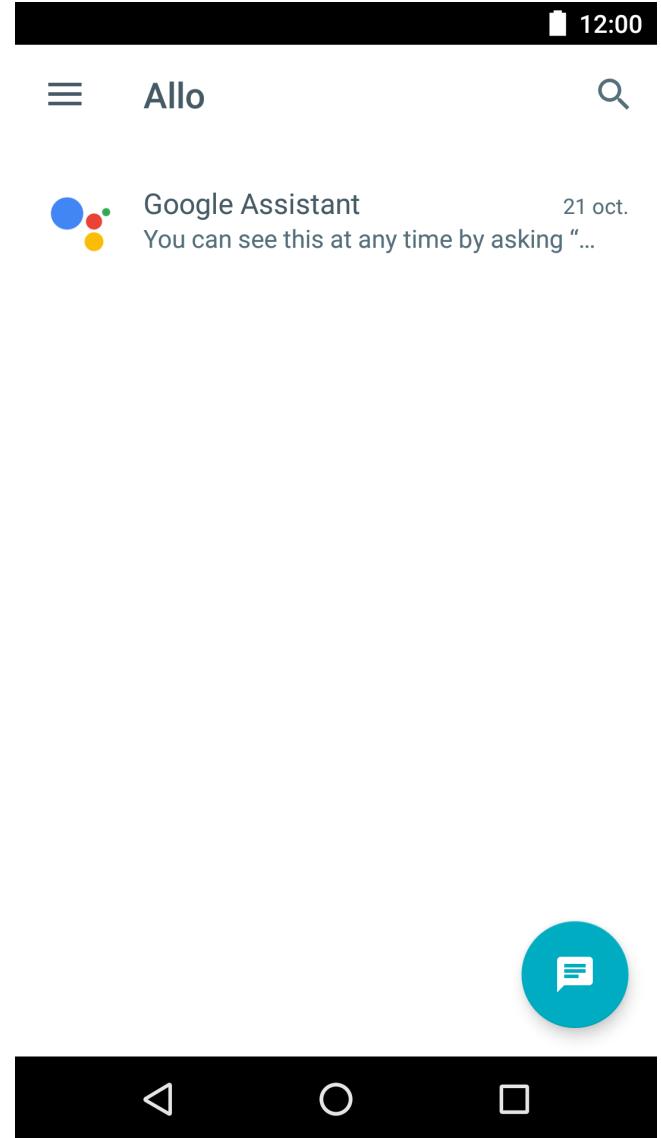
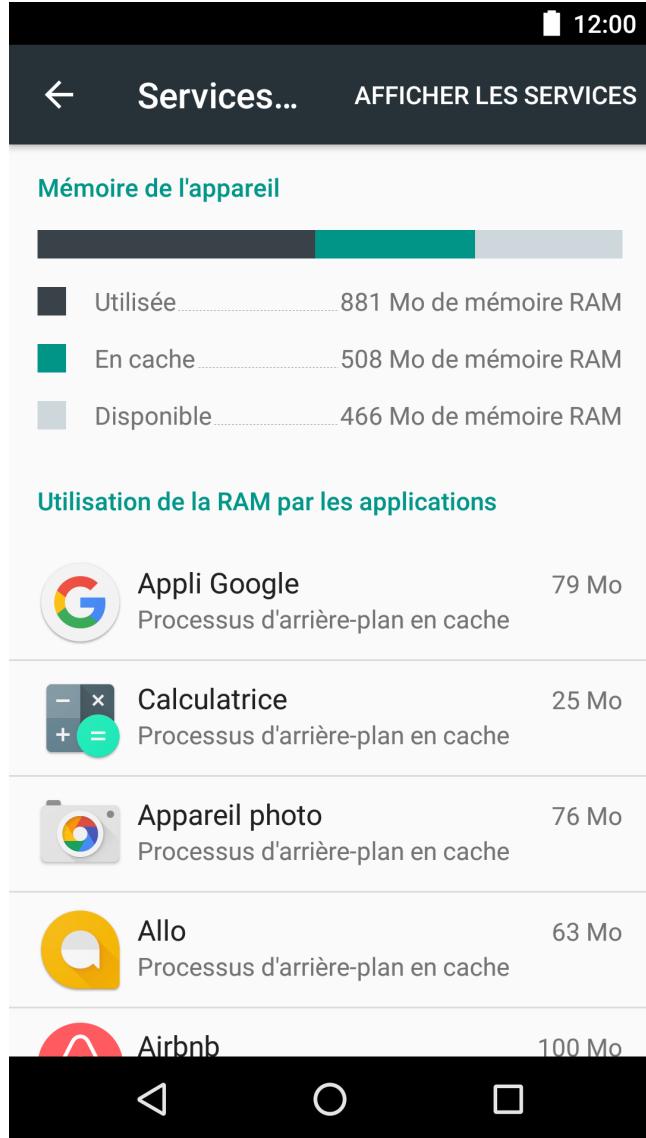


**Applications récentes**



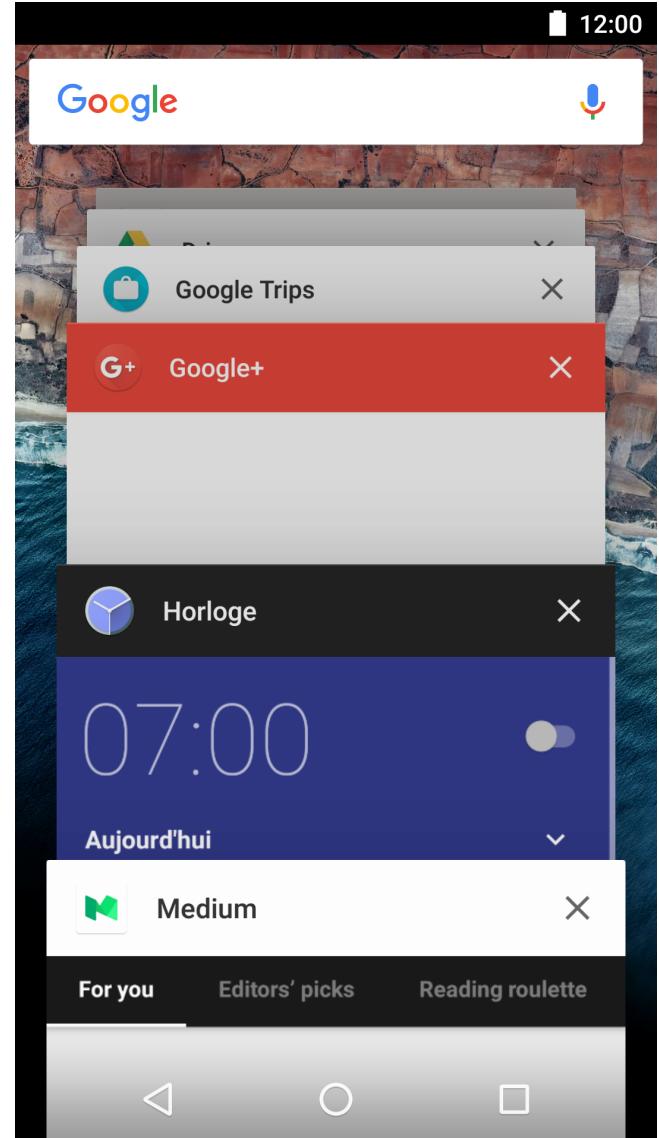


**Applications récentes.**  
**Tap sur Google Allo**

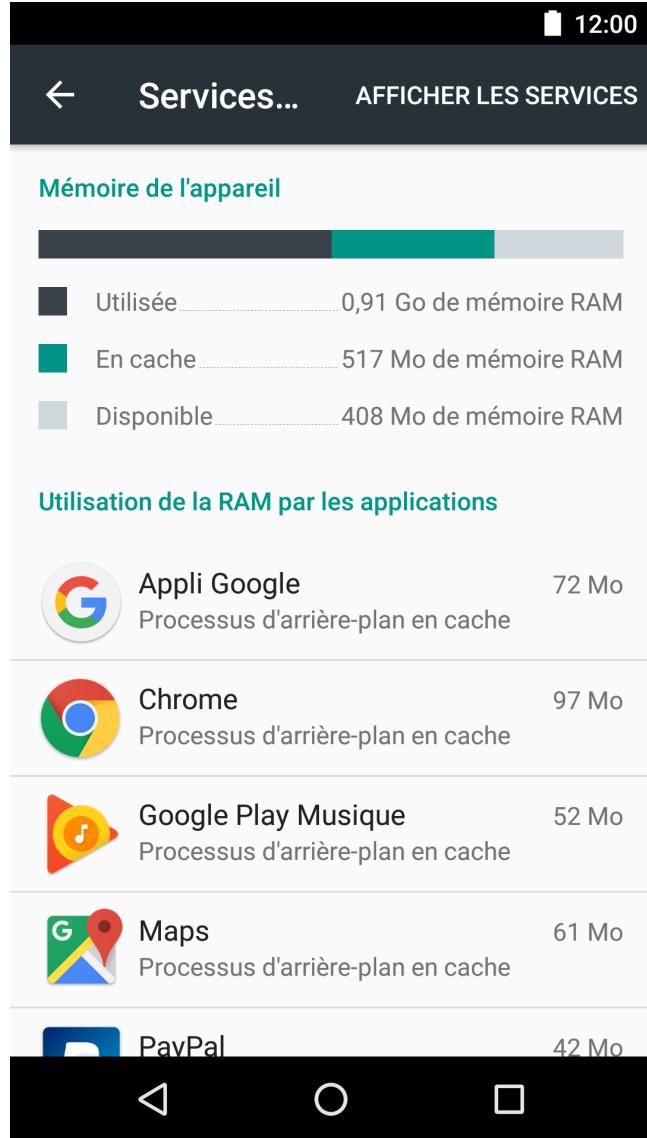




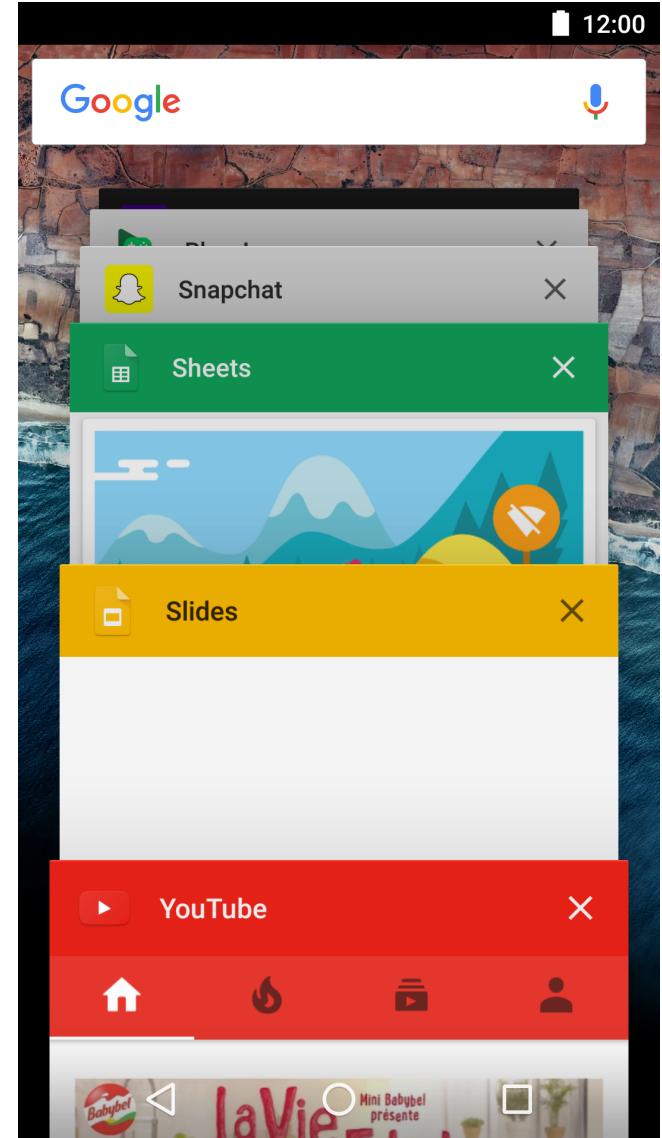
**Mémoire du téléphone presque saturée**



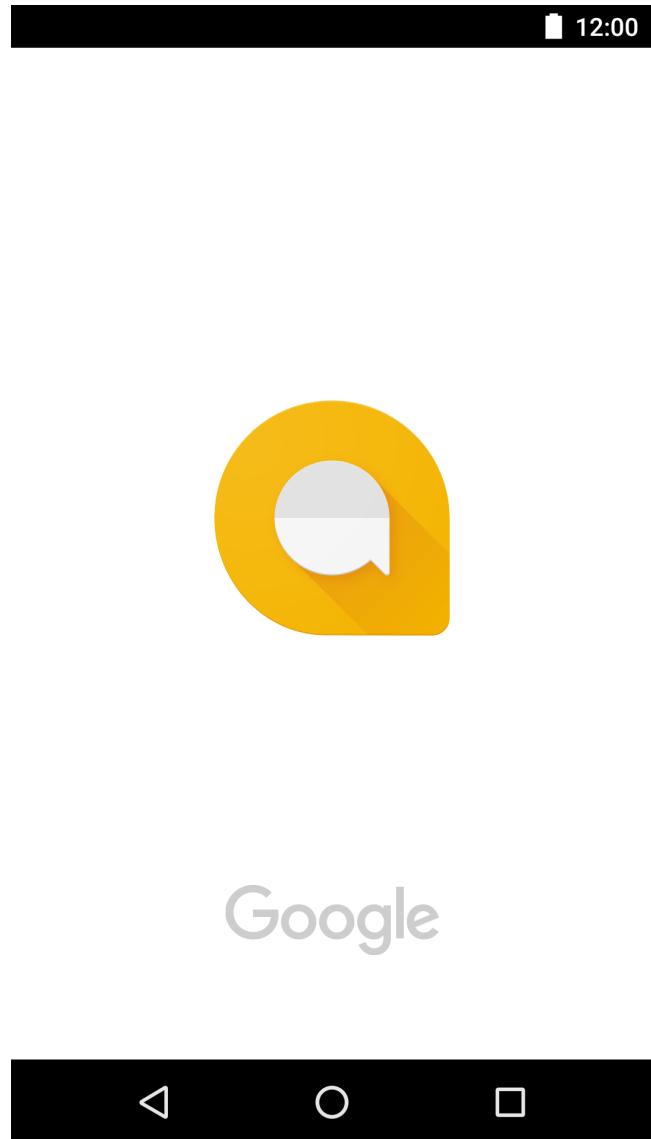
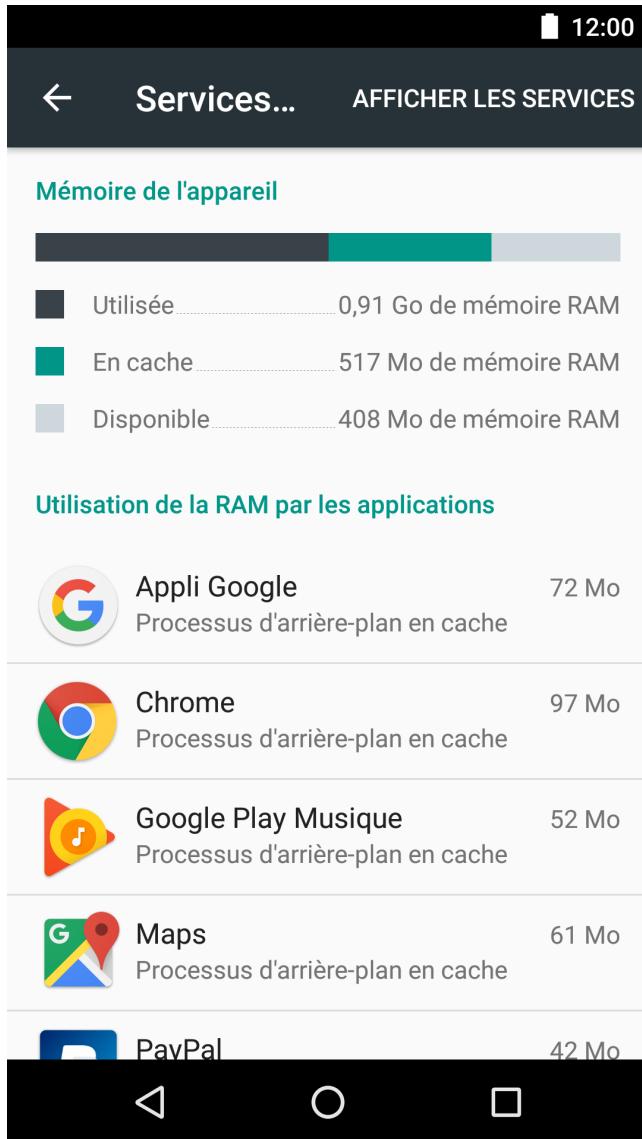
**Applications récentes**



**Mémoire du téléphone.  
Le système a tué des process pour  
libérer de la mémoire.**



**Applications récentes**



Process Google Allo tué par manque de mémoire. Relancement de l'application et recréation de l'ancienne Activity coutante.

# Gestion des process

## 1. Foreground process

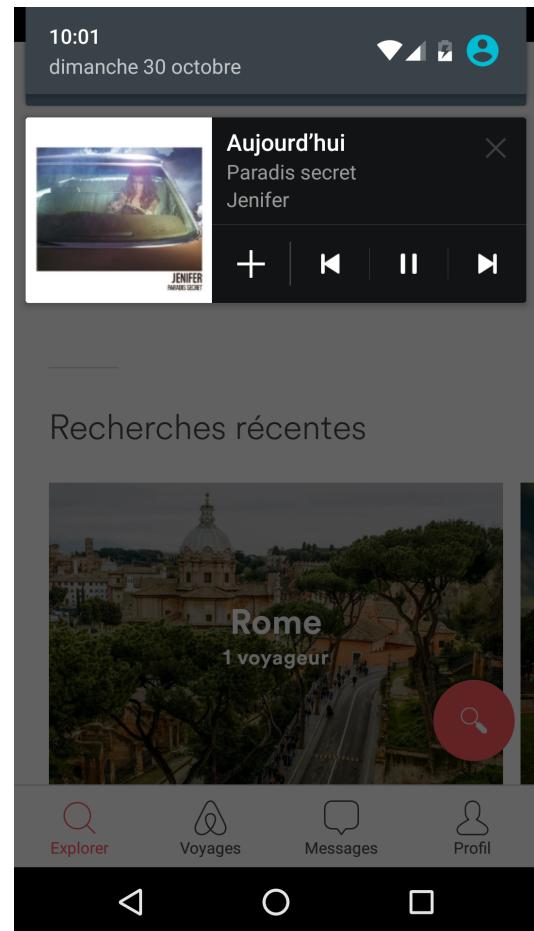
- L'utilisateur interagit avec une Activity de l'application
- Process tué en tout dernier recours



Spotify en premier plan

# Gestion des process

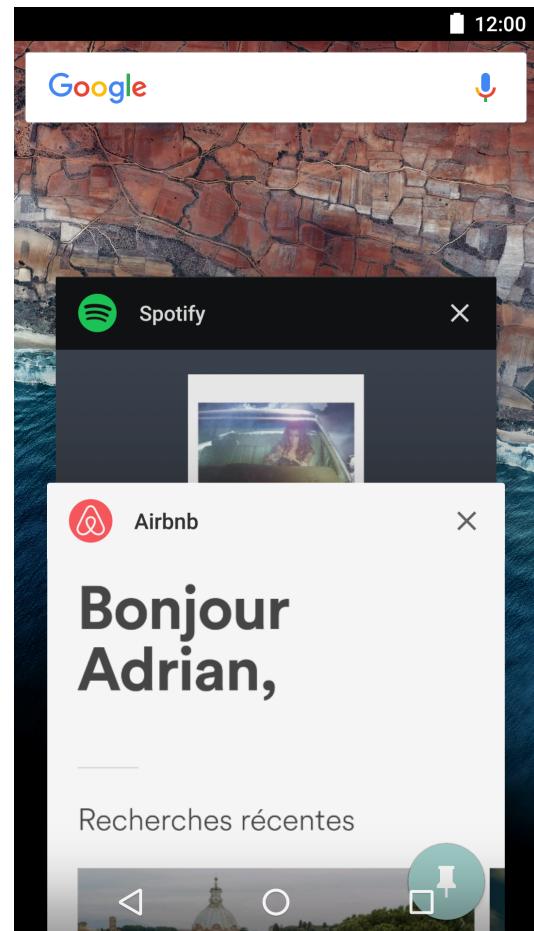
1. Foreground process
2. Service process
  - L'utilisateur n'interagit pas directement avec une Activity de l'application, il l'utilise indirectement
  - Process tué uniquement si mémoire nécessaire pour faire tourner les foreground processes



Spotify en arrière plan avec  
musique qui tourne

# Gestion des process

1. Foreground process
2. Service process
3. Background process
  - L'utilisateur n'intéragit plus du tout avec l'application
  - Process tué si mémoire nécessaire pour faire tourner les foreground processes ou les service processes



Spotify en arrière plan et inactif

# Bonnes pratiques

- Libérer les ressources consommatoires quand application en background
- Ne pas perdre l'avancement de l'utilisateur quand process de l'application tué et Activity courante recréée
- Afficher un contenu toujours récent à l'utilisateur
- Ne pas cracher quand l'utilisateur jongle entre l'application et une autre

# Activity Lifecycle

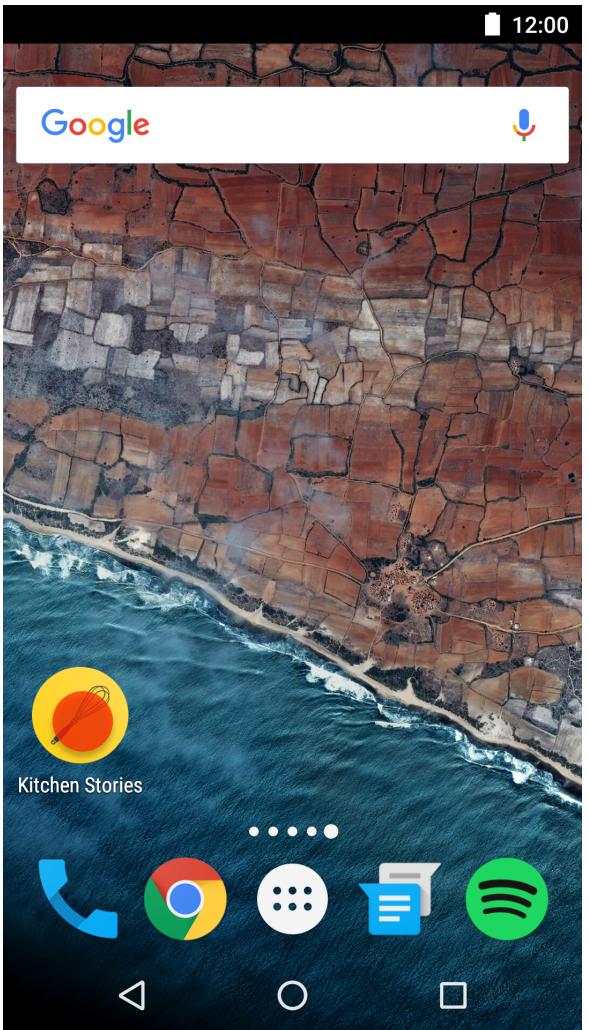
```
public class MyActivity extends Activity {  
    // Appelé à la création de l'Activity  
    @Override  
    protected void onCreate();  
}
```

# Activity Lifecycle

```
public class MyActivity extends Activity {  
  
    @Override  
    protected void onCreate();  
  
    @Override  
    protected void onStart();  
  
    @Override  
    protected void onResume();  
  
    @Override  
    protected void onPause();  
  
    @Override  
    protected void onStop();  
  
    @Override  
    protected void onDestroy();  
}
```

# Activity Lifecycle

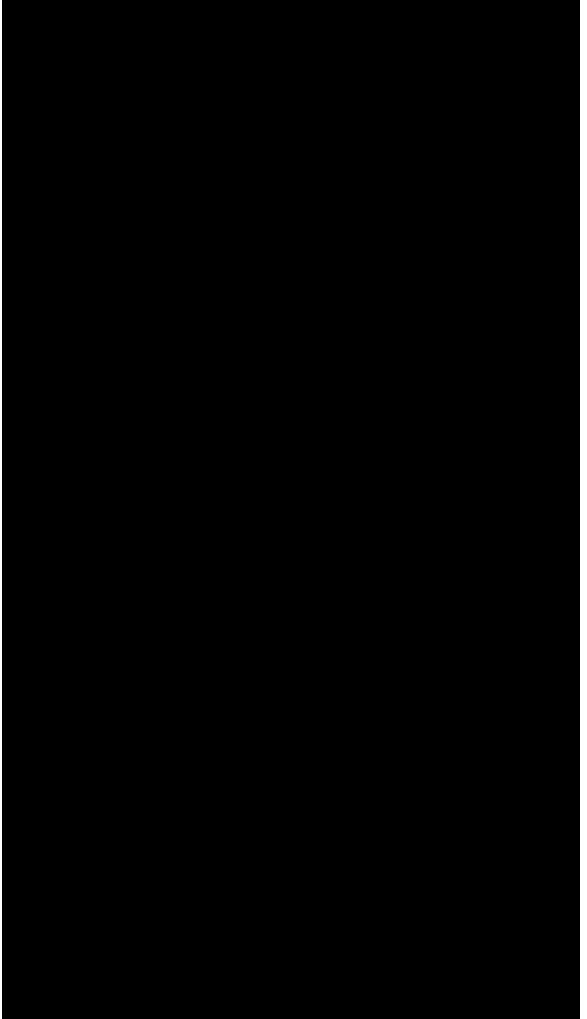
- Enchaînement de méthodes appelées par le système au cours de la vie d'une Activity
- Traduit l'ensemble des états dans lesquels une Activity peut être.



**Tap sur l'application  
Kitchen Stories.**

onCreate

Lifecycle de l'Activity  
« Liste de recettes »

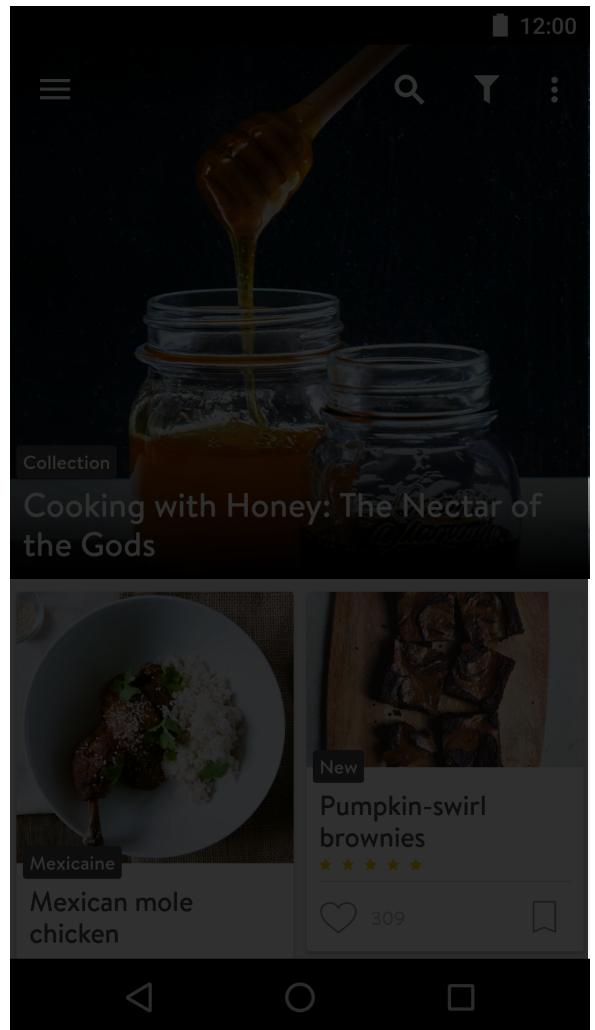


Création de l'Activity principale « Liste de recettes ».  
Activity pas encore visible.

onCreate



onStart

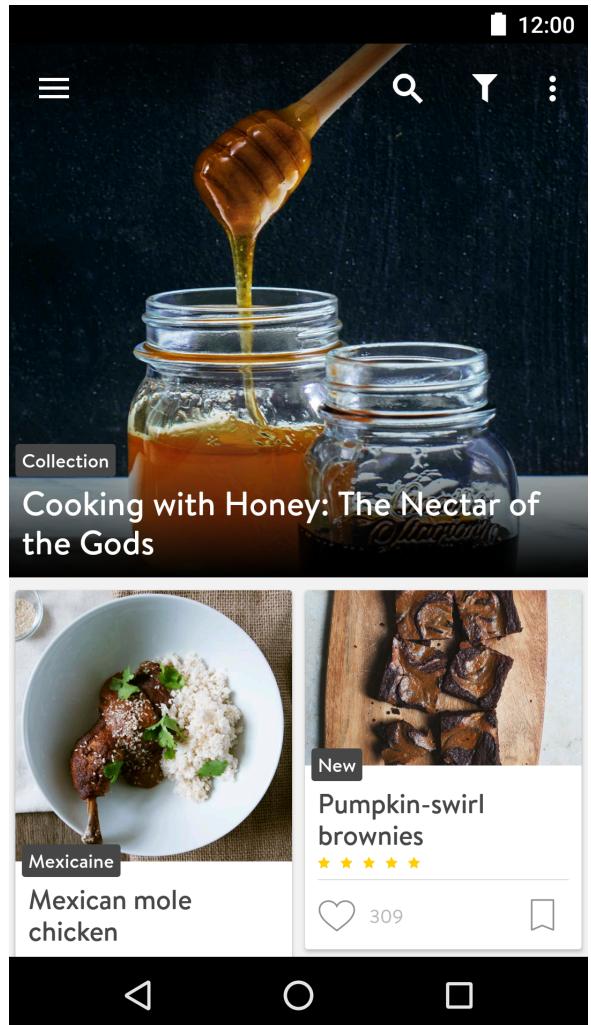


Lifecycle de l'Activity  
« Liste de recettes »

Activity « Liste de  
recettes » crée et en train  
de devenir visible.



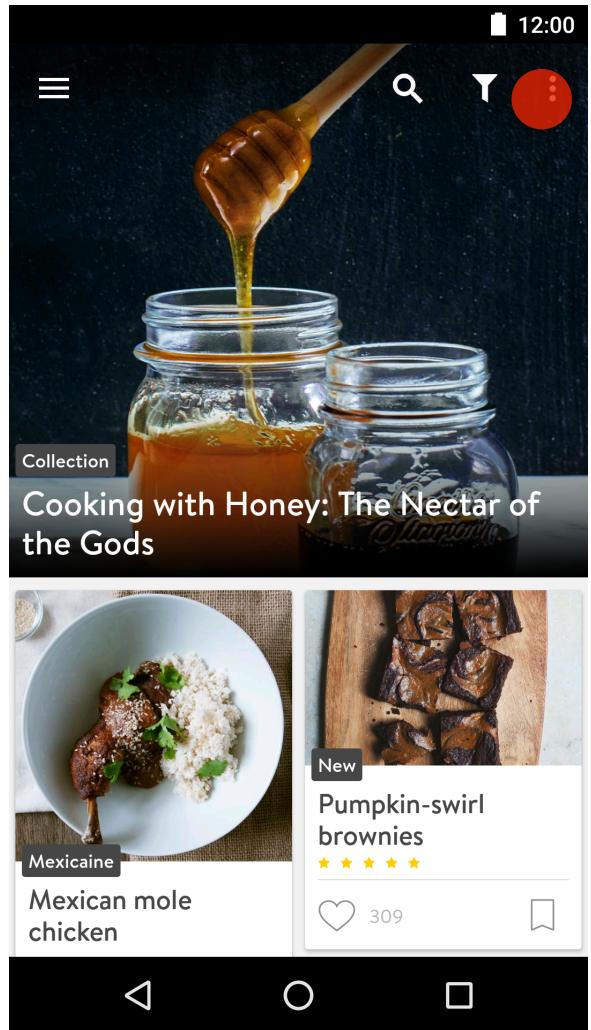
**Lifecycle de l'Activity  
« Liste de recettes »**



**Activity « Liste de  
recettes » visible.  
L'utilisateur peut interagir  
avec.**



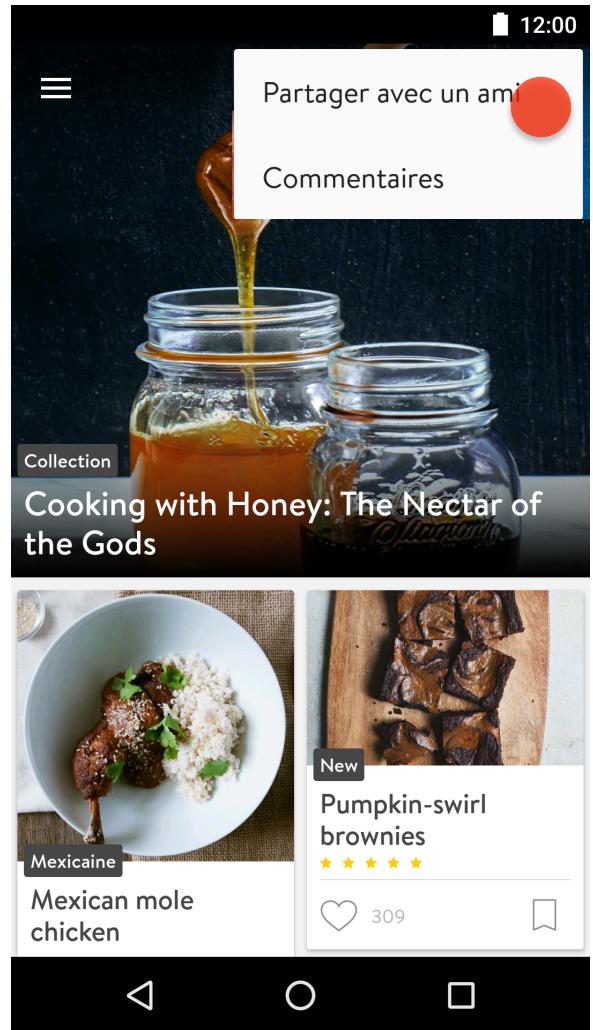
**Lifecycle de l'Activity**  
« Liste de recettes »



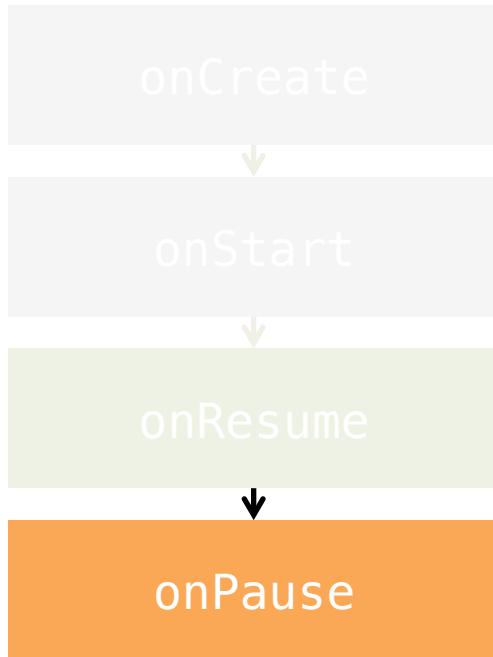
Tap sur les options.



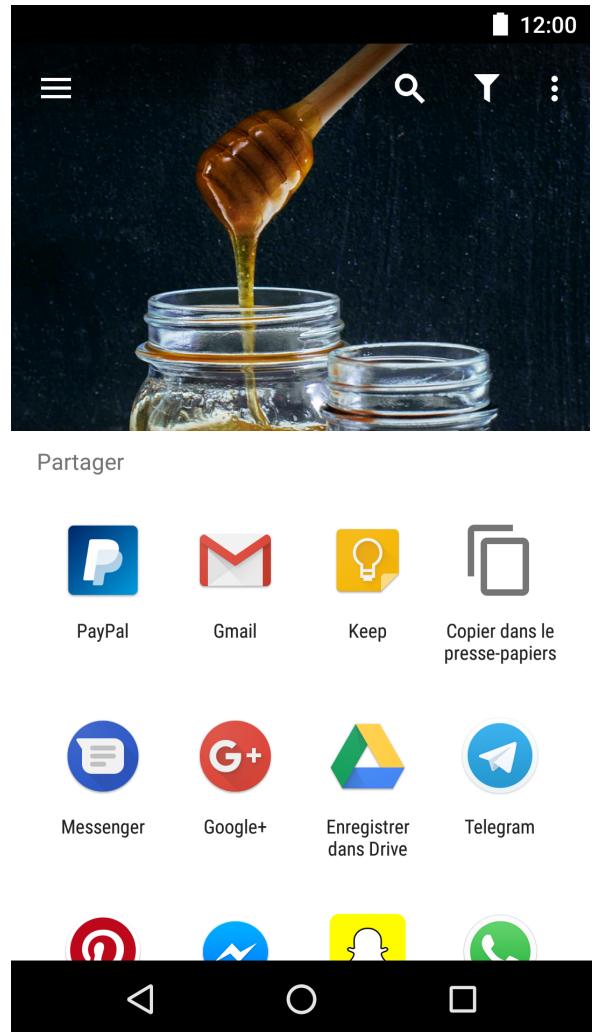
**Lifecycle de l'Activity  
« Liste de recettes »**



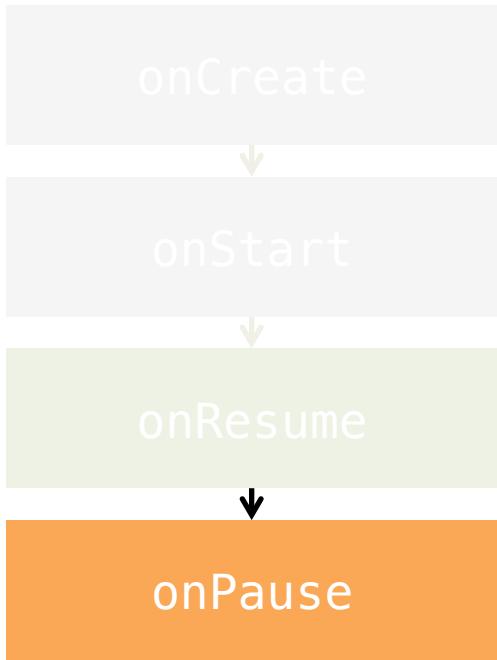
**Tap sur « partager avec un ami ».**



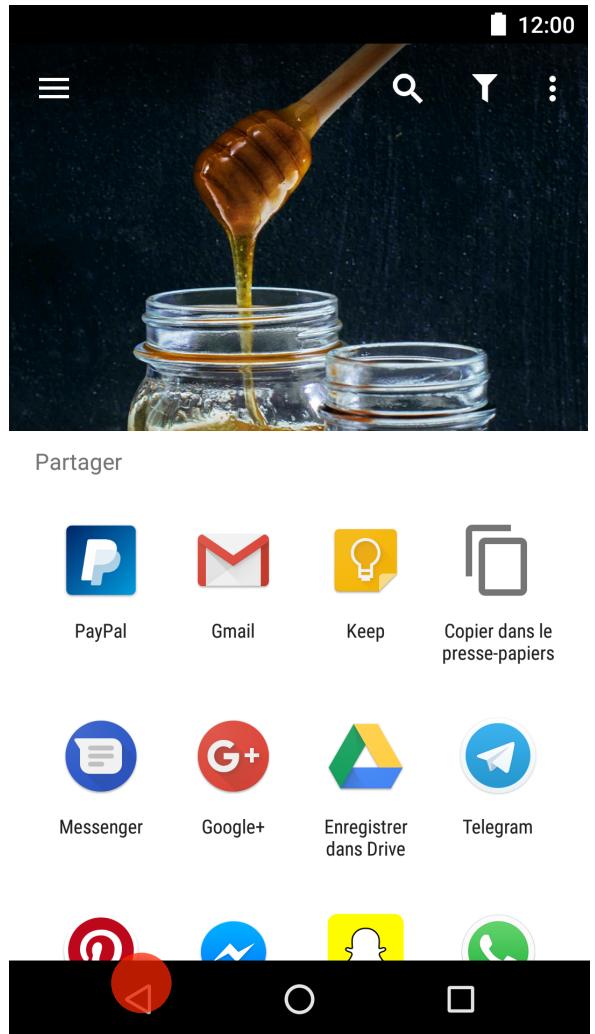
## Lifecycle de l'Activity « Liste de recettes »



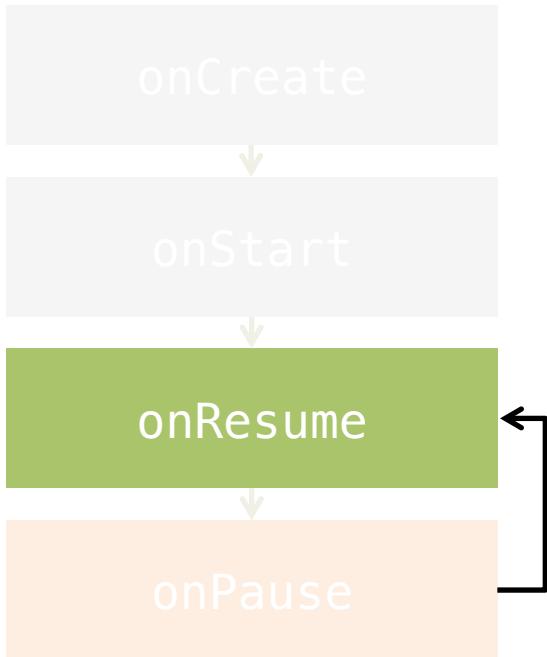
Fenêtre de partage lancée et Activity « Liste de recettes » visible mais plus au premier plan.



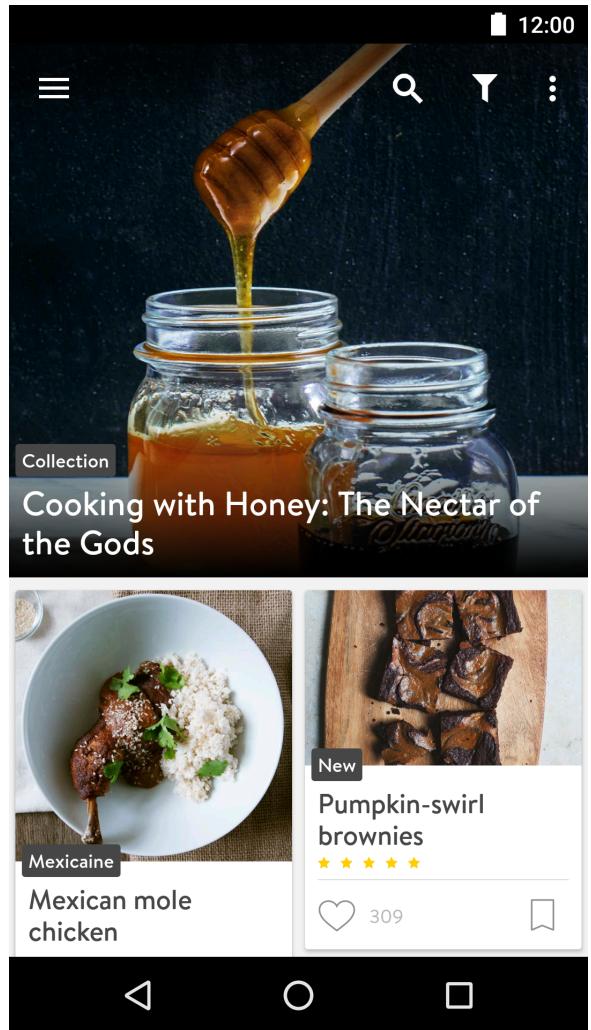
**Lifecycle de l'Activity**  
 « Liste de recettes »



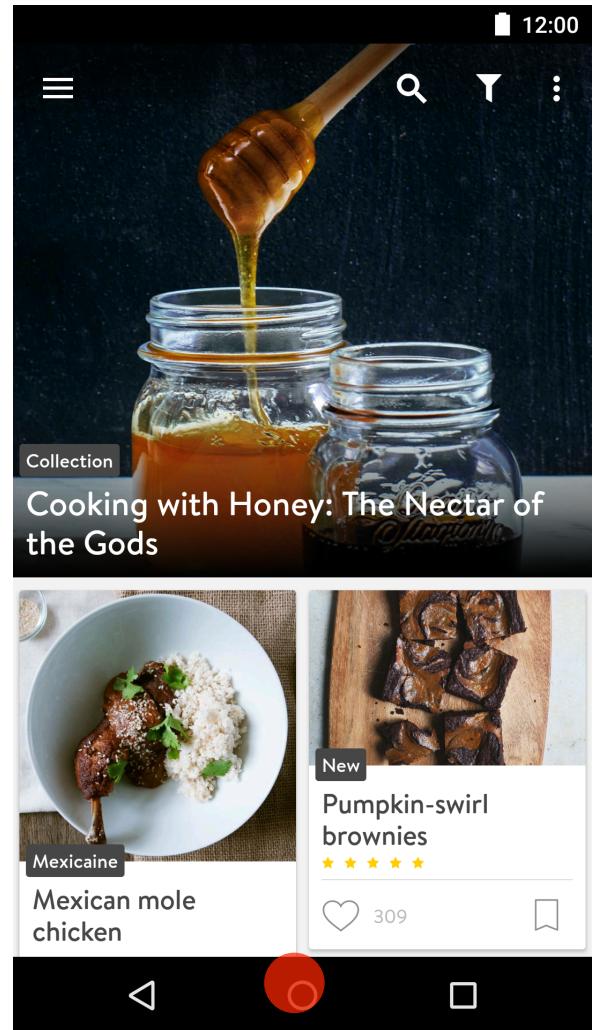
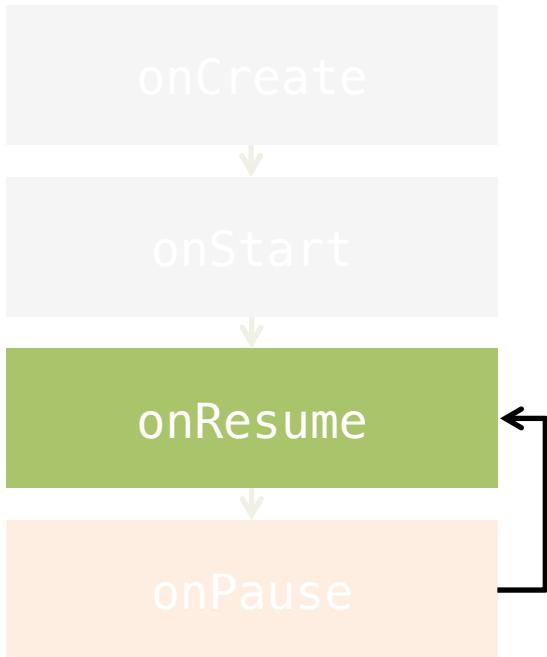
Tap sur BACK.



**Lifecycle de l'Activity  
« Liste de recettes »**

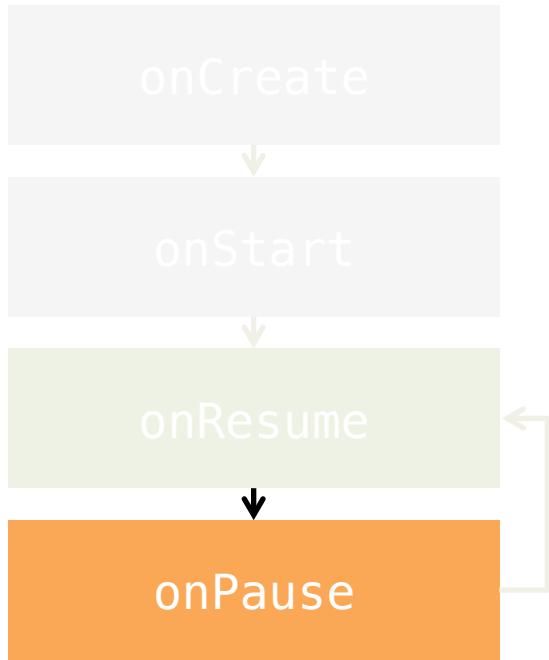


**Retour sur Activity « Liste de recettes ». Activity à nouveau au premier plan.**

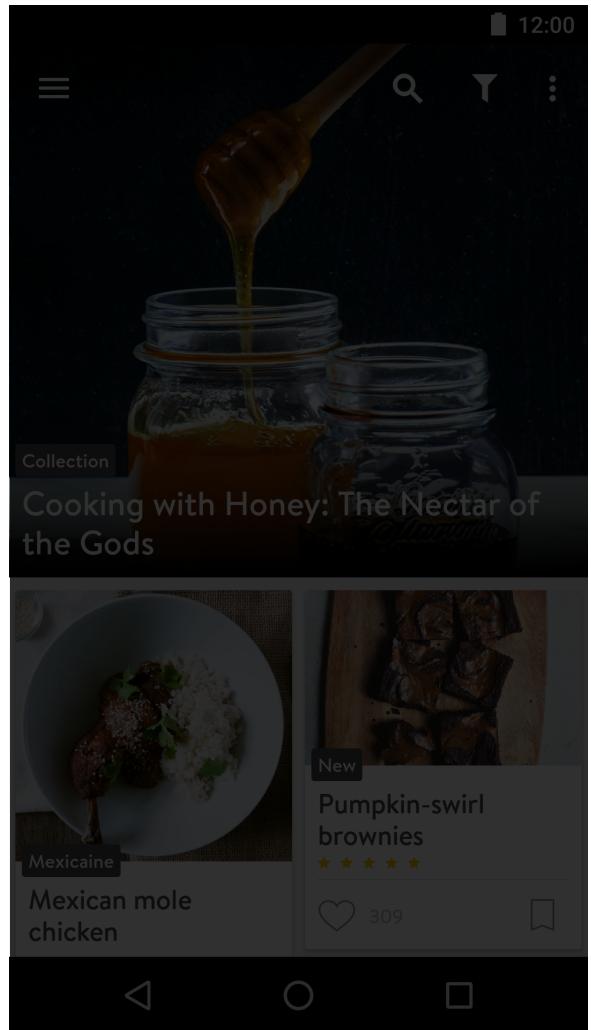


**Lifecycle de l'Activity**  
« Liste de recettes »

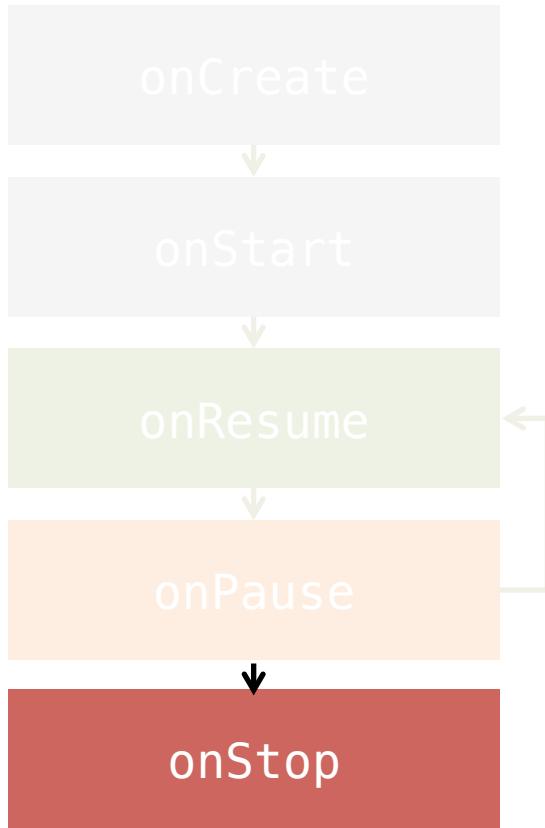
Tap sur HOME.



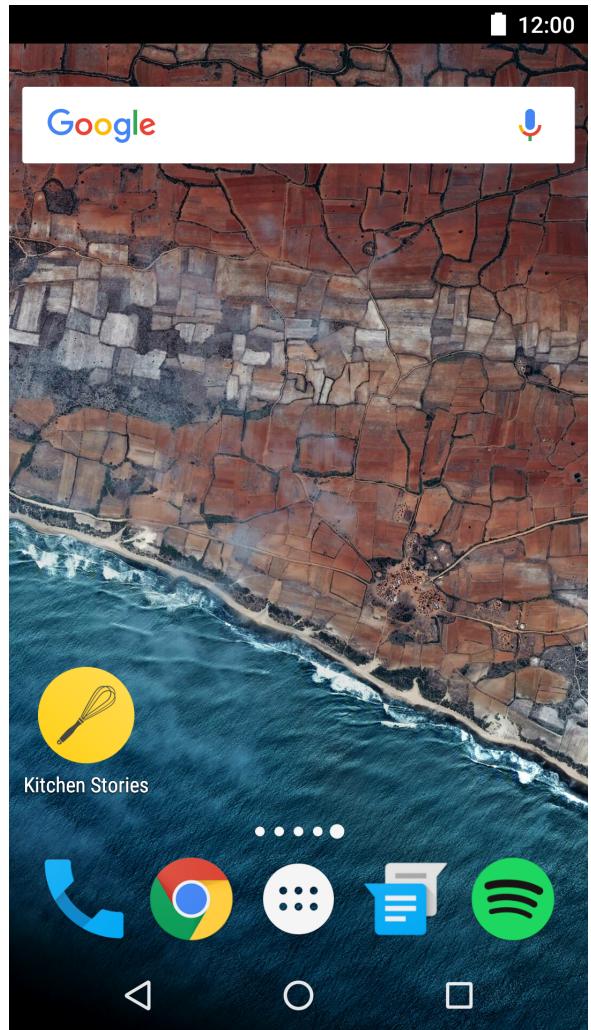
**Lifecycle de l'Activity  
« Liste de recettes »**



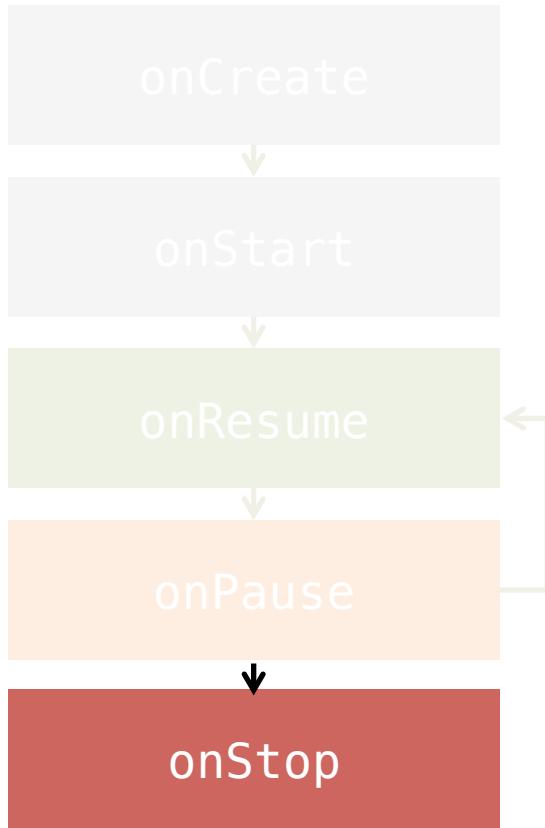
**L'Activity « Liste de Recette » est en train de disparaître.**



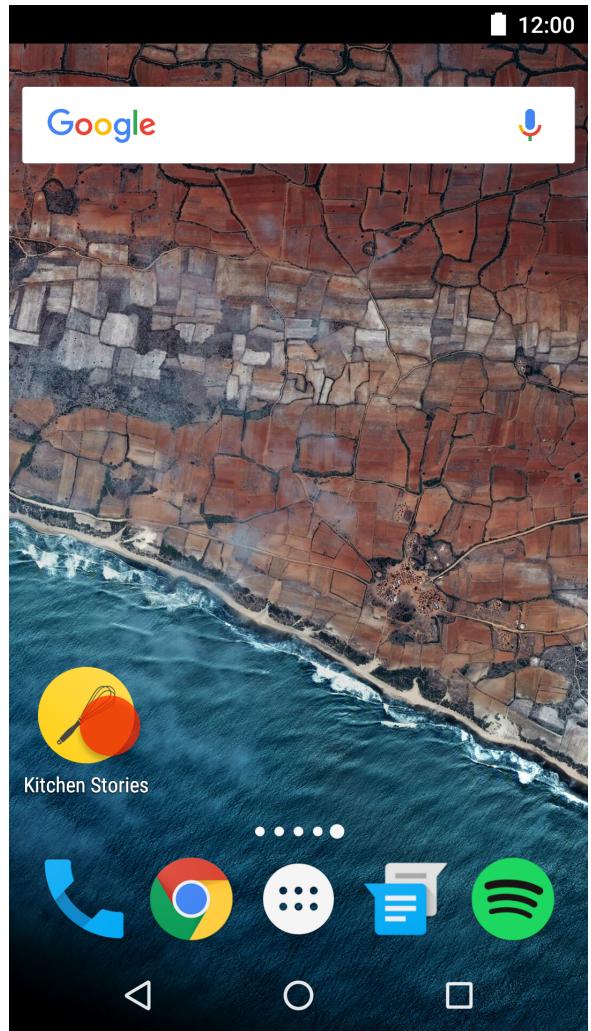
## Lifecycle de l'Activity « Liste de recettes »



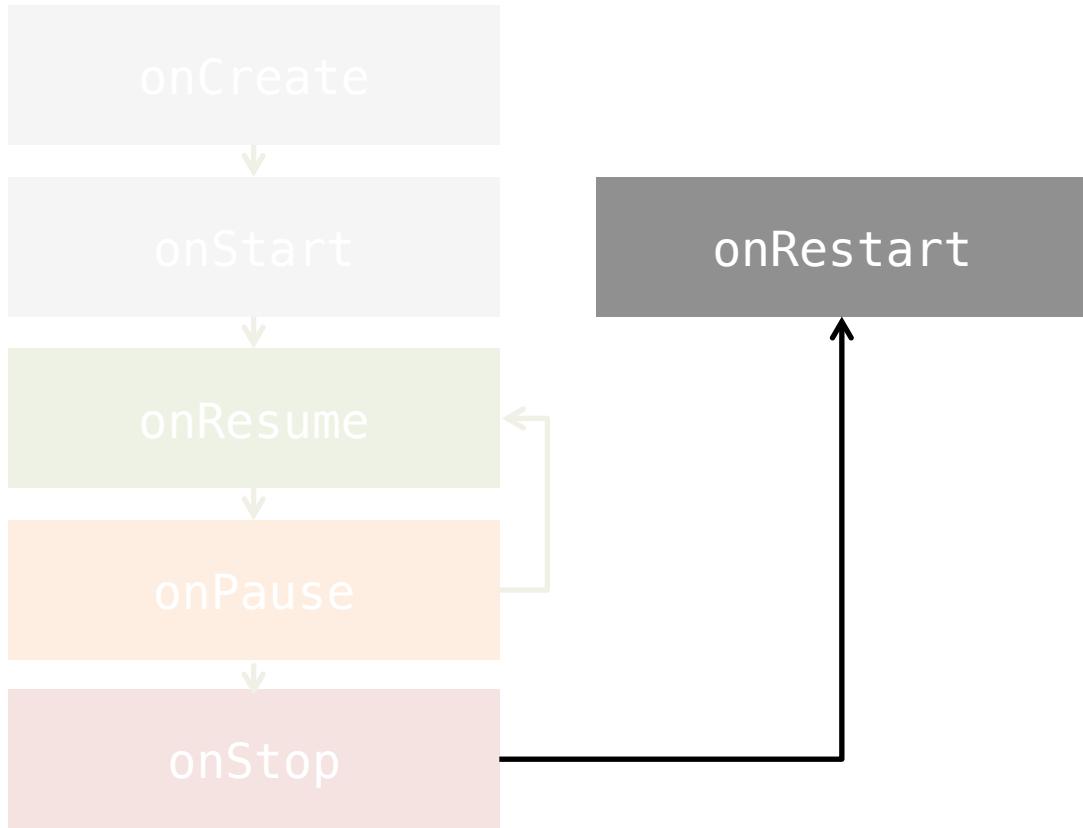
**Retour sur la home. L'Activity « Liste de Recette » complètement cachée. Process de l'application en background.**



**Lifecycle de l'Activity**  
« Liste de recettes »

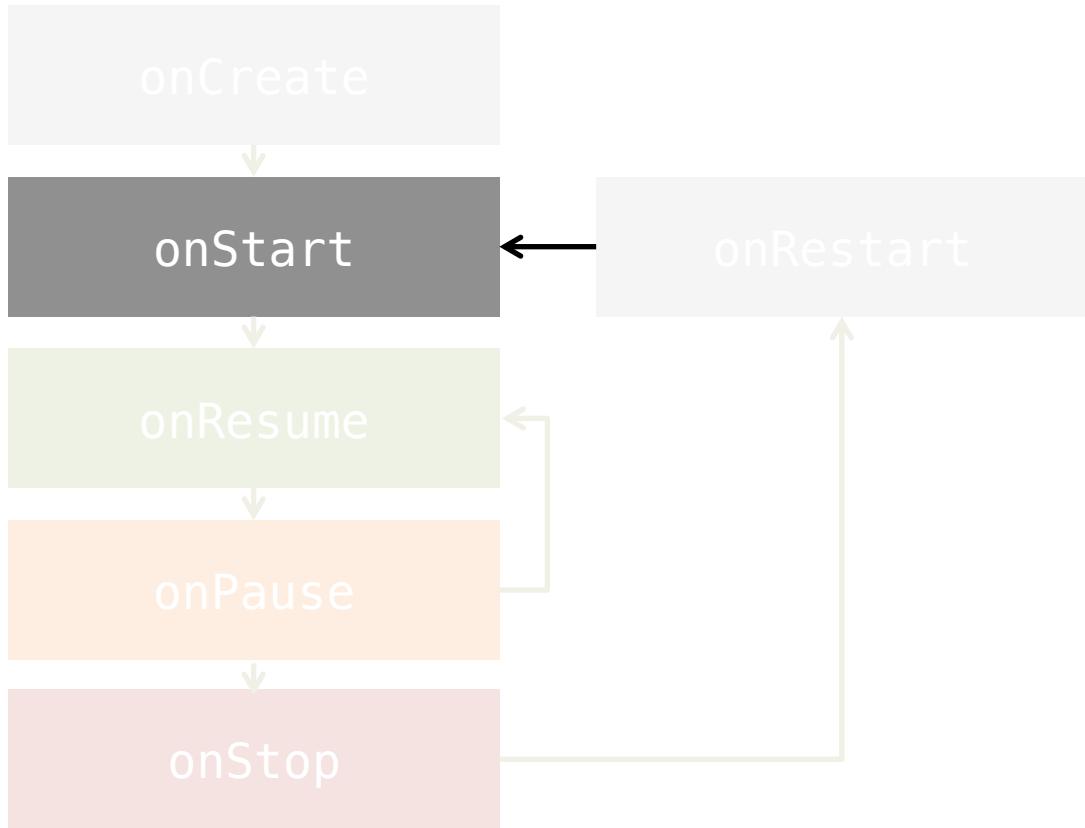


**Tap sur l'application.**

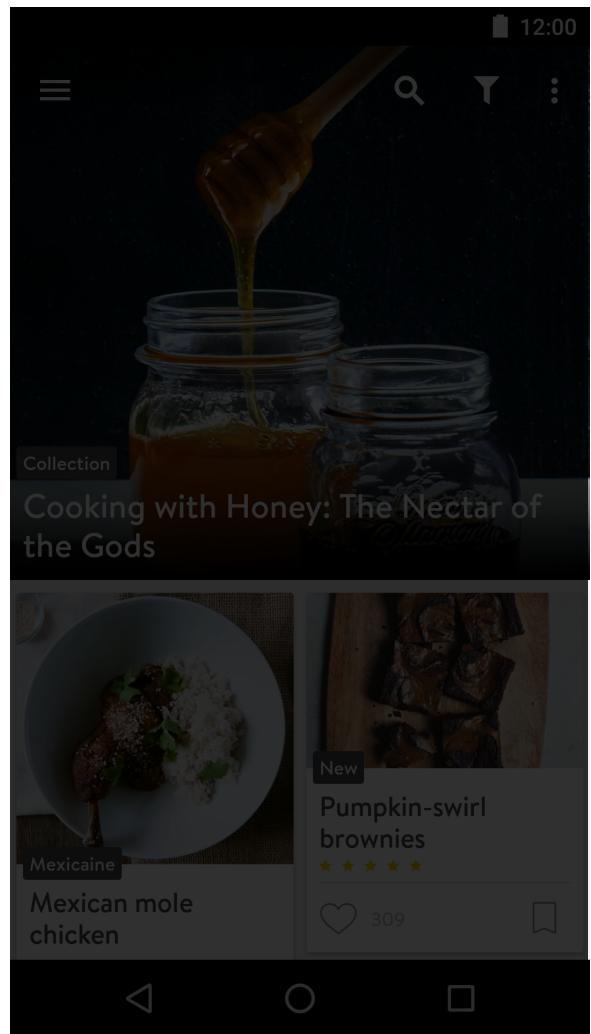


**Lifecycle de l'Activity  
« Liste de recettes »**

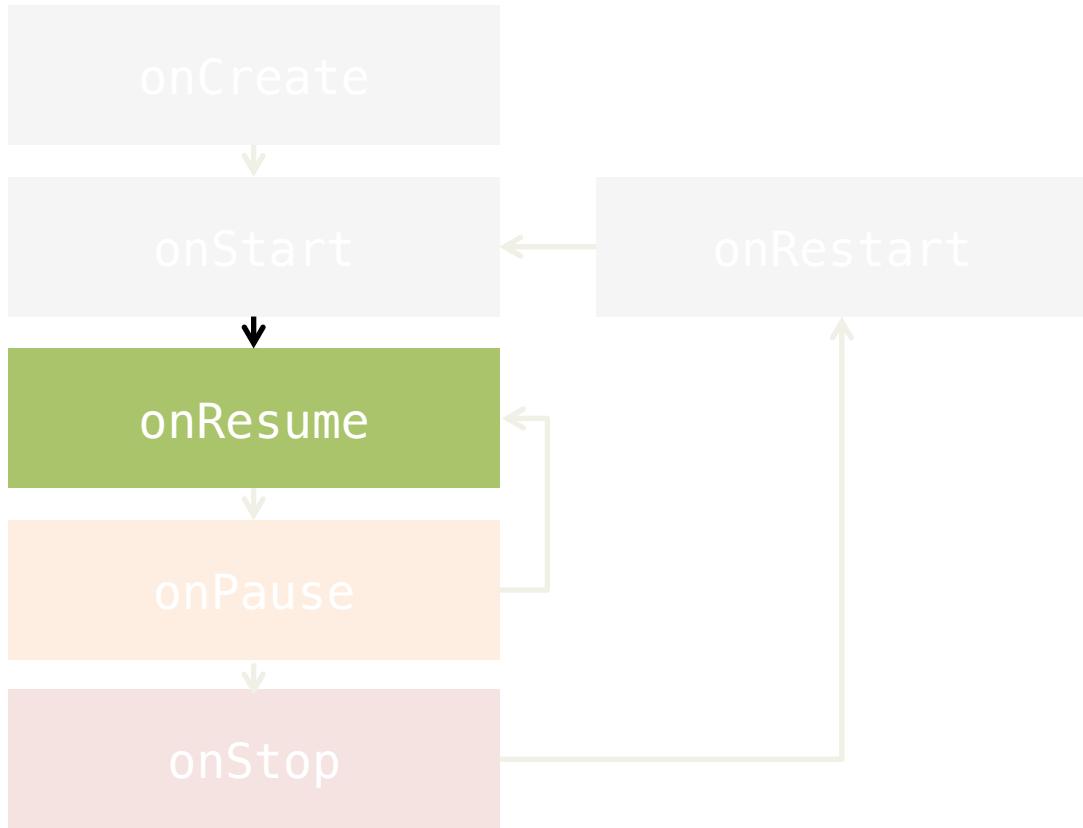
**Relancement de  
l'application. Redémarrage  
de l'Activity « Liste de  
recettes ».**



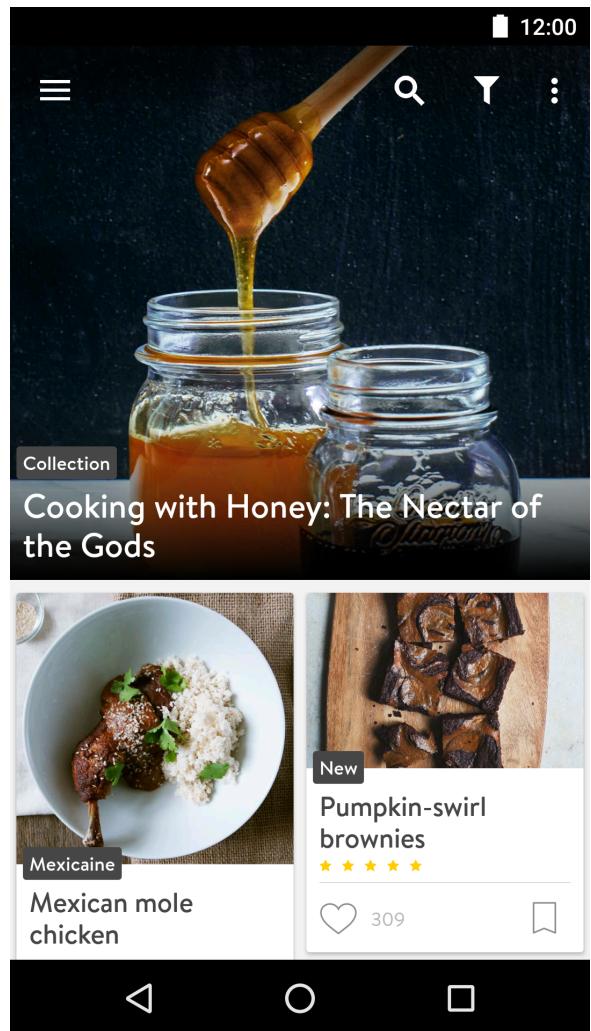
**Lifecycle de l'Activity  
« Liste de recettes »**



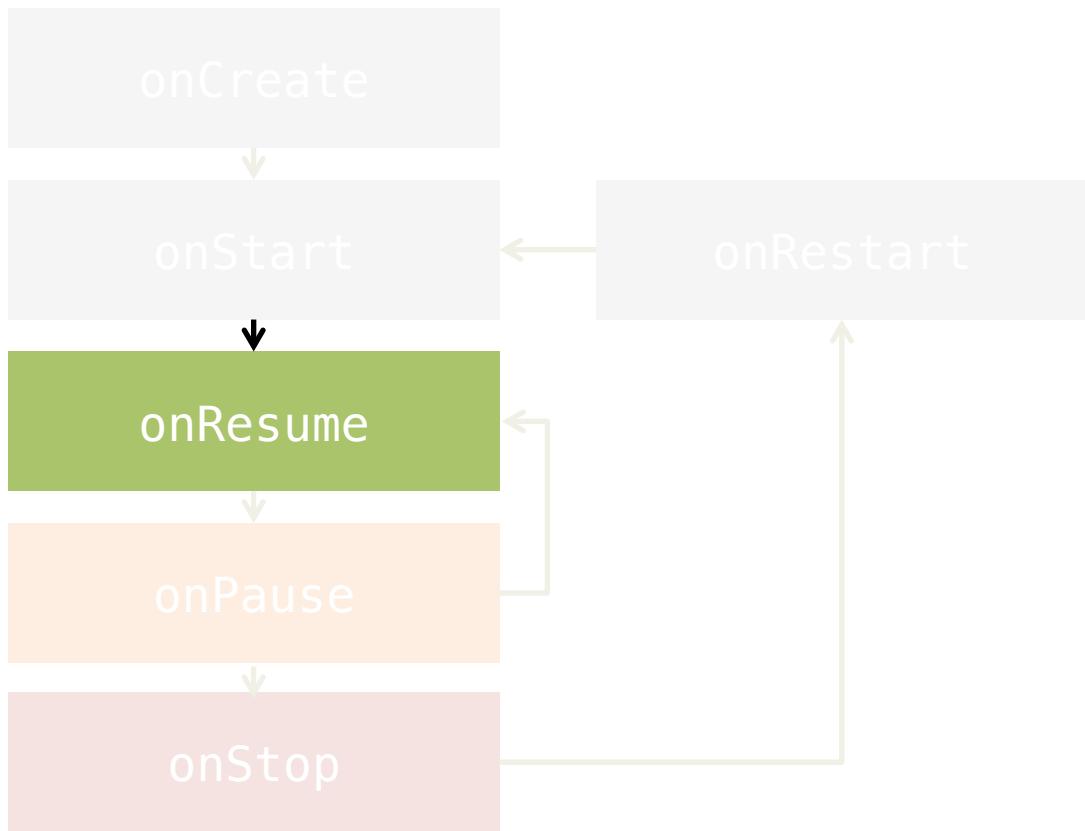
**Activity « Liste de  
Recette » redémarrée et  
en train de devenir visible.**



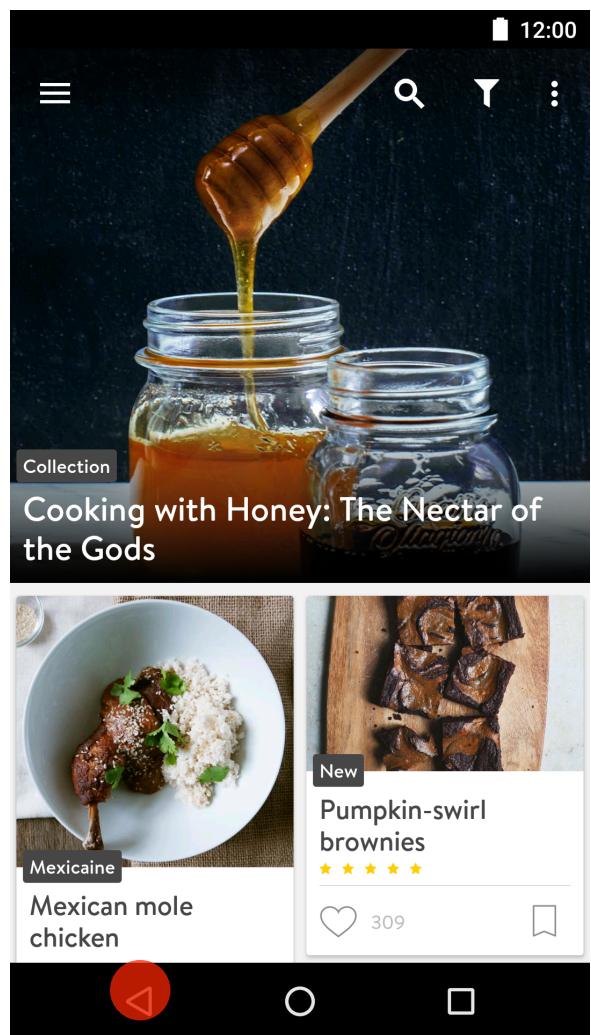
**Lifecycle de l'Activity  
« Liste de recettes »**



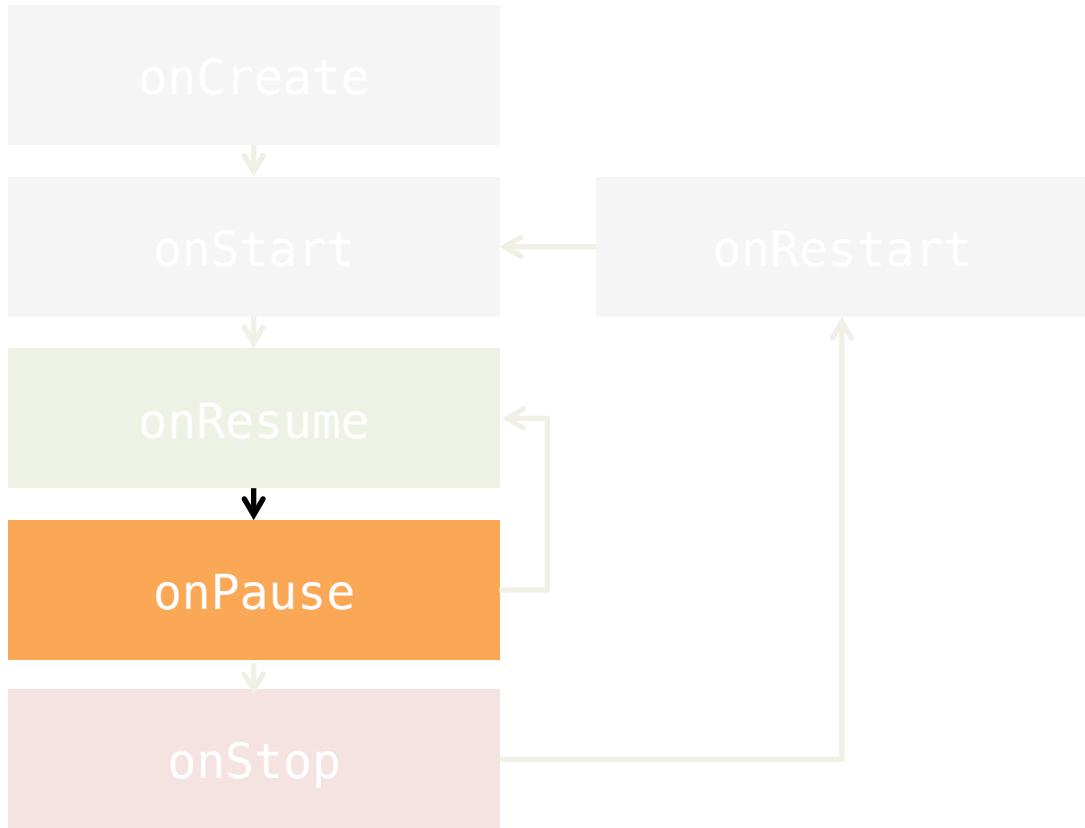
**Activity « Liste de recettes » visible. Process de l'application en foreground.**



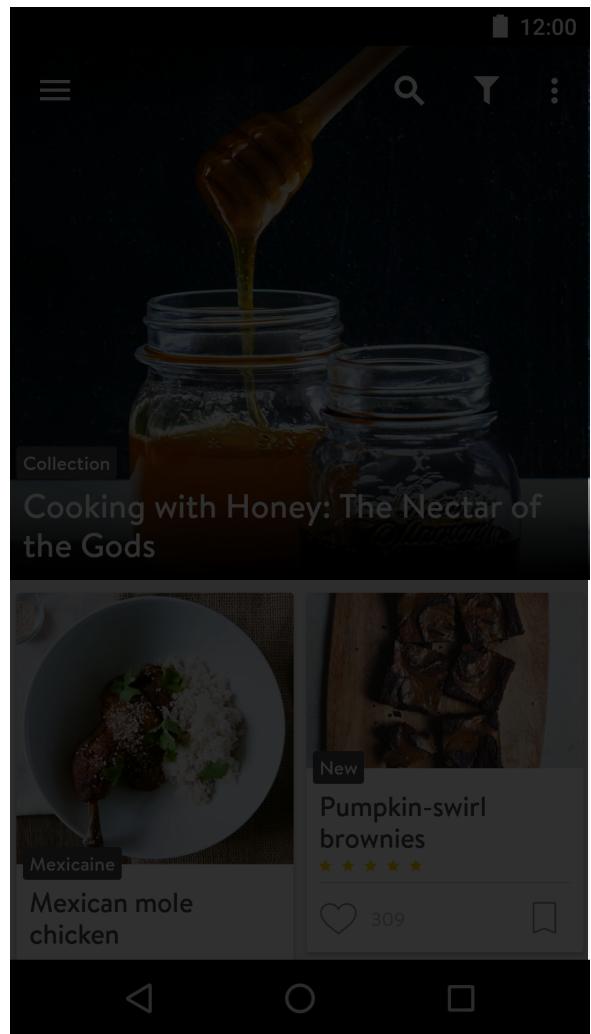
**Lifecycle de l'Activity**  
 « Liste de recettes »



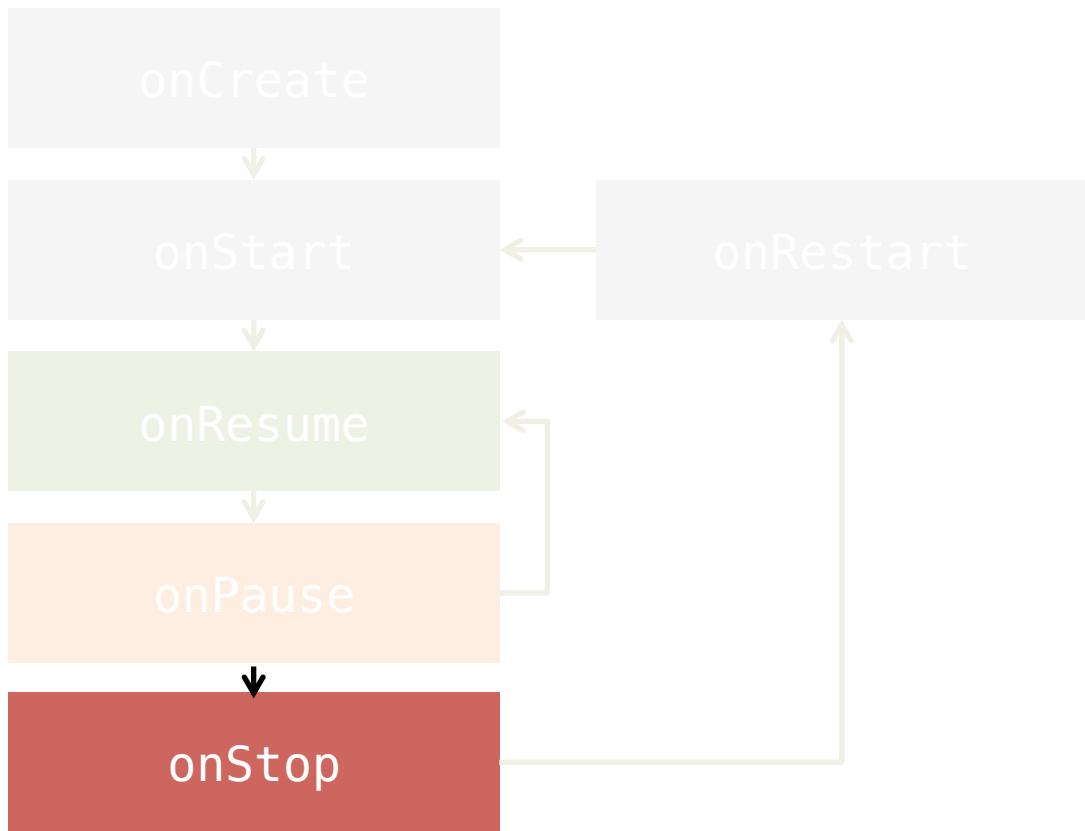
Tap sur BACK.



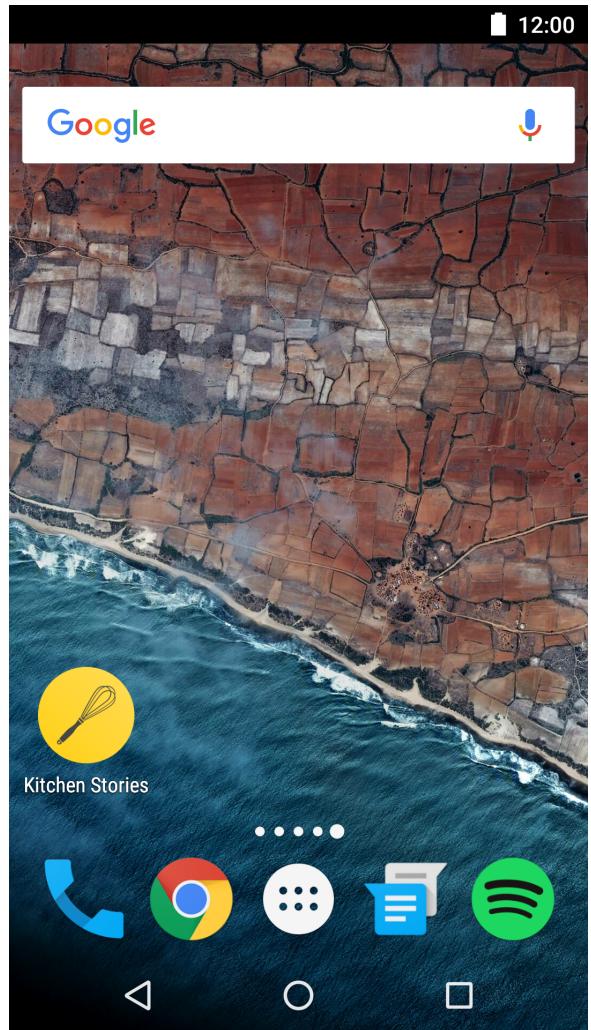
**Lifecycle de l'Activity  
« Liste de recettes »**



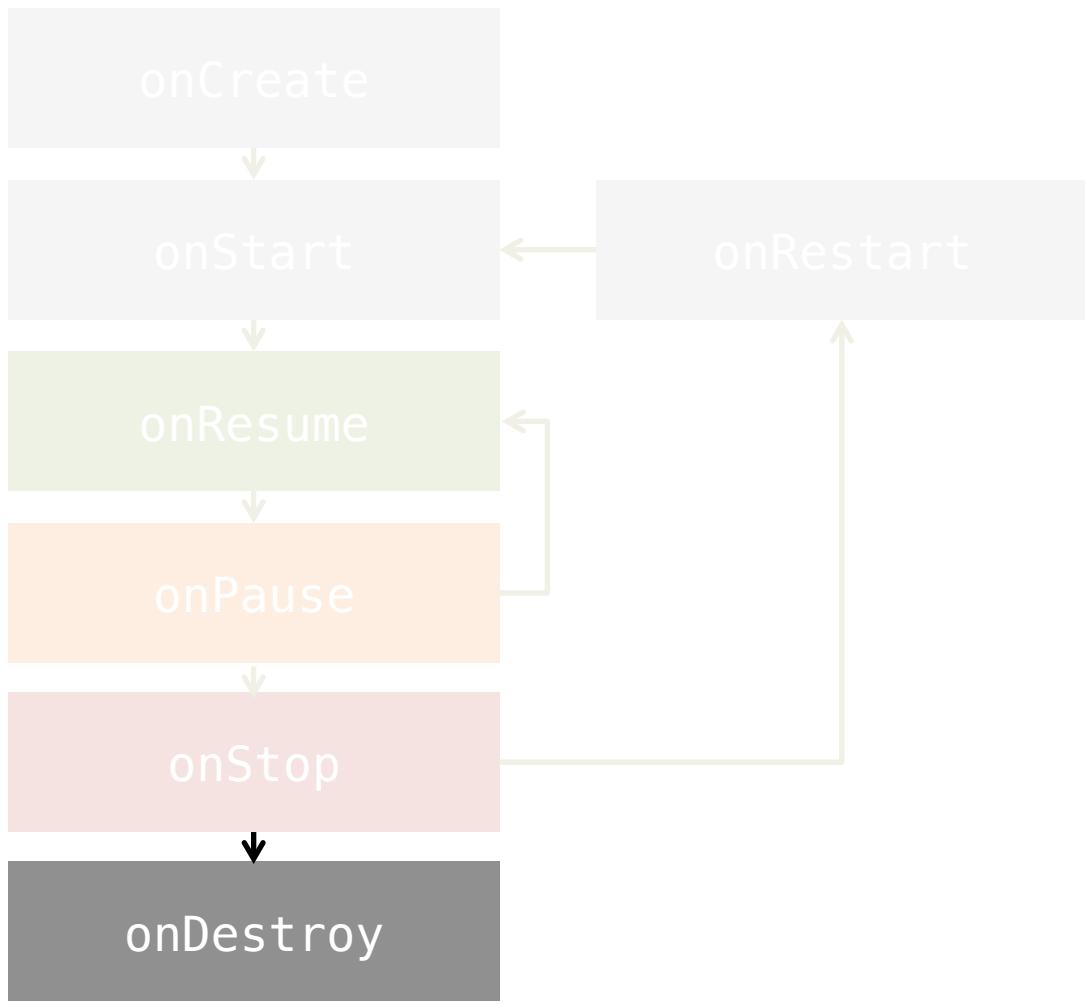
**L'Activity « Liste de  
recettes » est en train de  
disparaître.**



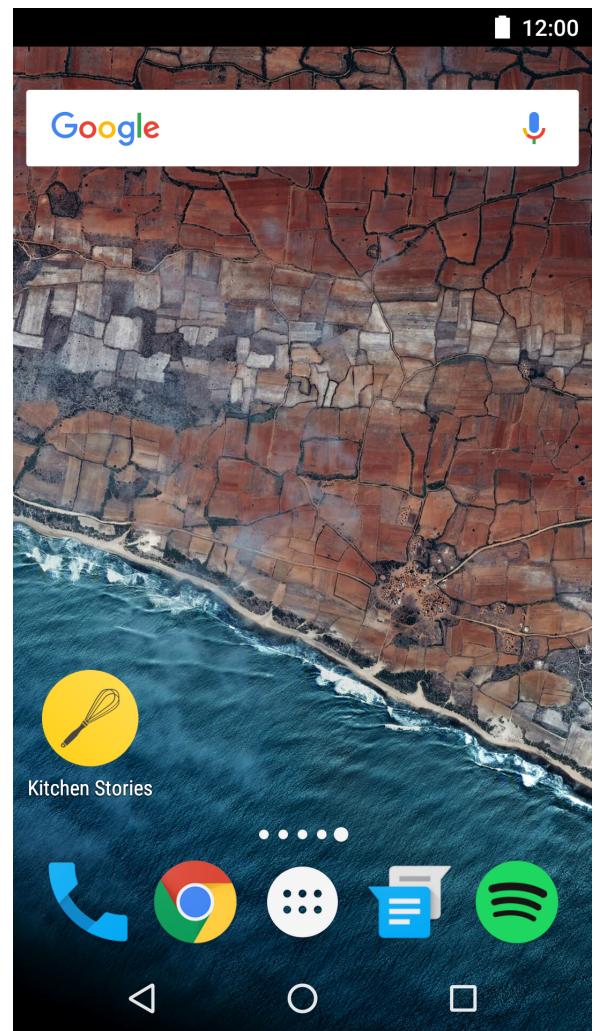
## Lifecycle de l'Activity « Liste de recettes »



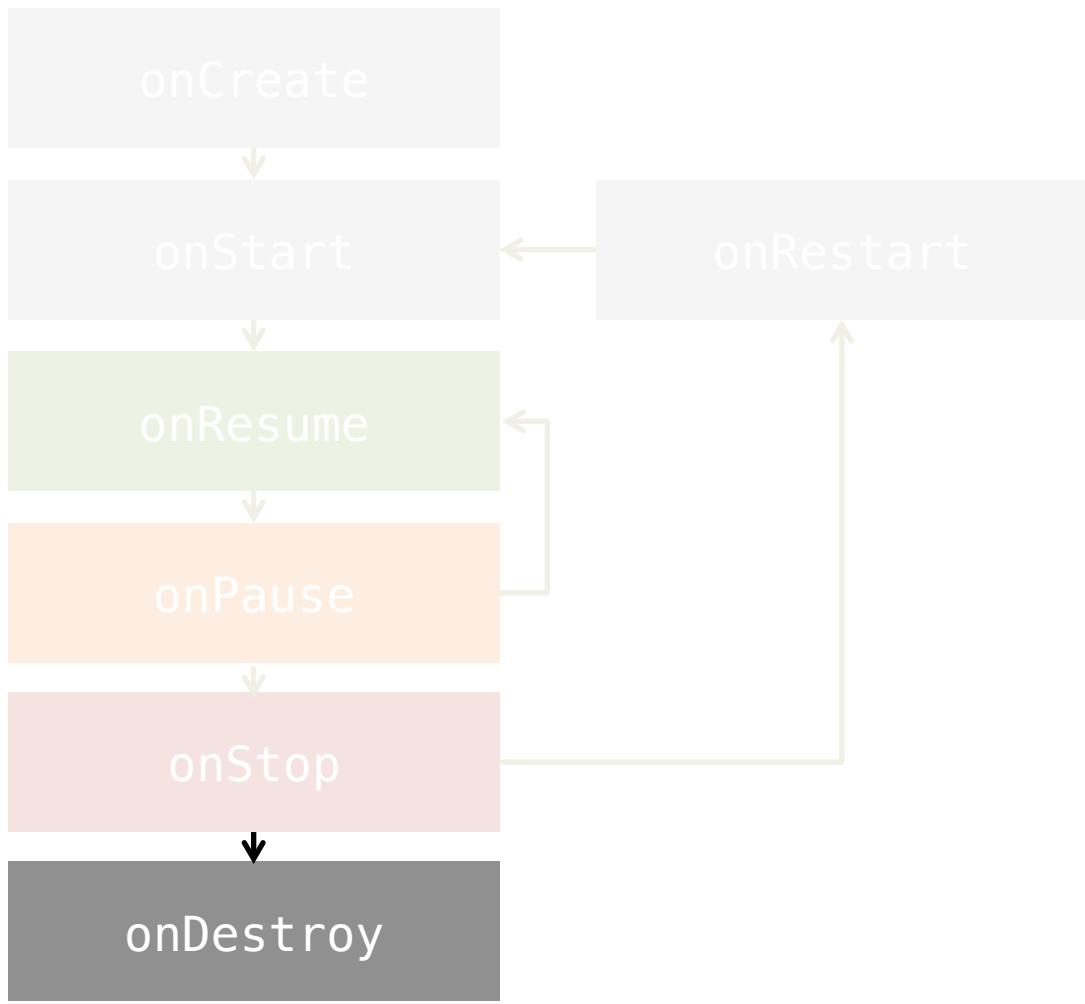
Retour sur la home. L'Activity « Liste de Recette » complètement cachée. Process de l'application en background.



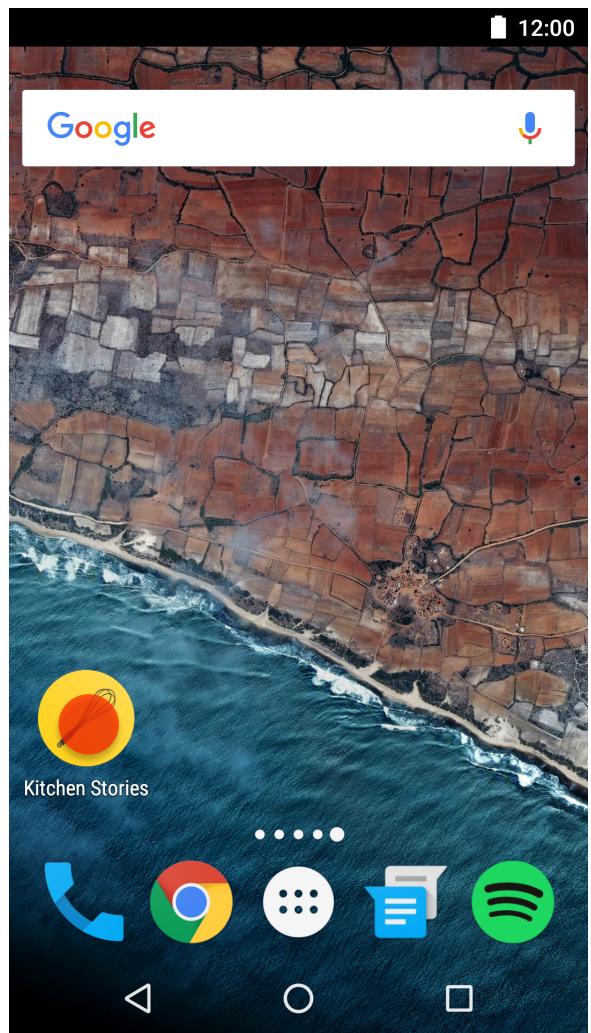
**Lifecycle de l'Activity  
« Liste de recettes »**



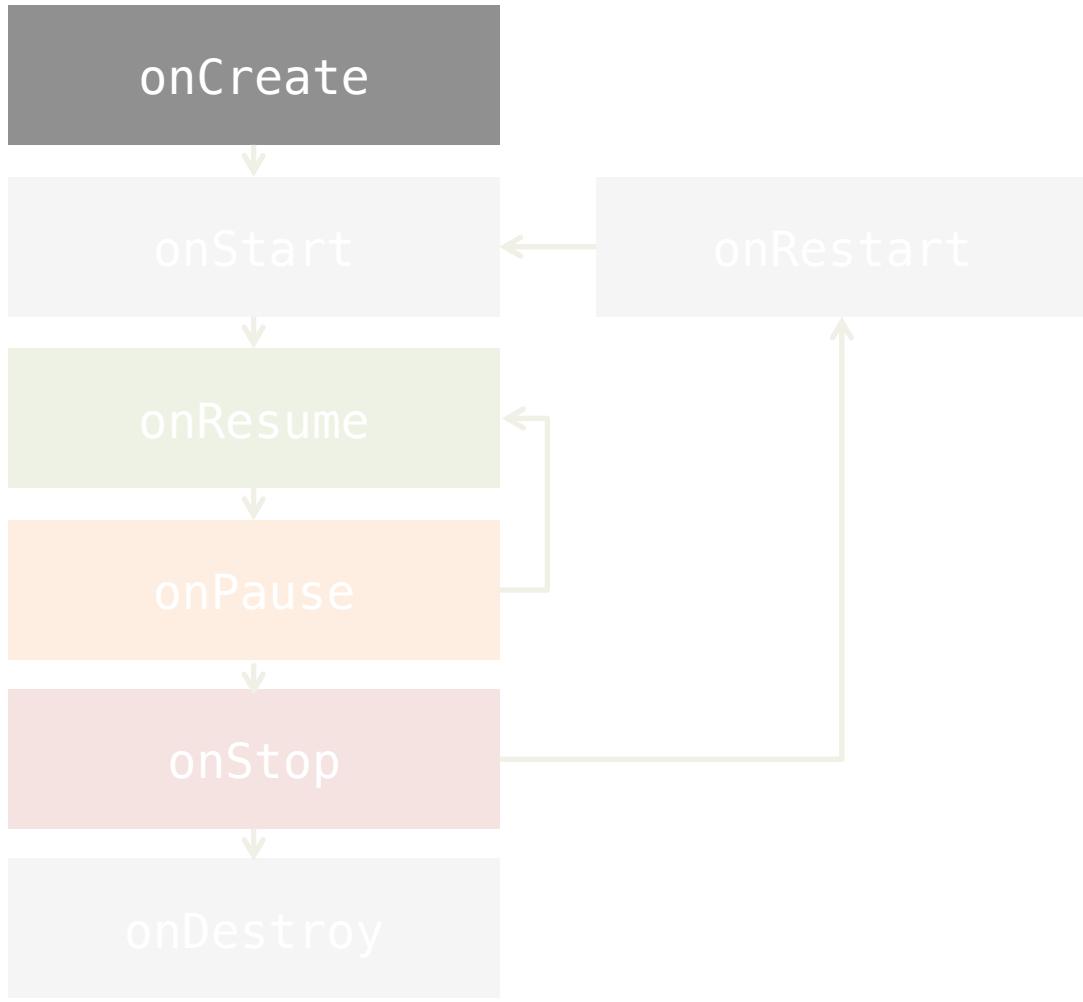
**Activity « Liste de  
recettes » détruite.  
Process tué par le  
système.**



**Lifecycle de l'Activity**  
« Liste de recettes »



**Tap sur l'application  
Kitchen Stories.**



**Lifecycle de l'Activity  
« Liste de recettes »**

**Recréation de l'Activity  
principale « Liste de  
recettes ».**

# Les états d'une Activity

- Etat **resumed** (onResume)
  - Activity en premier plan et l'utilisateur peut interagir avec
  - Process de l'application en foreground

# Les états d'une Activity

- Etat **resumed** (onResume)
- Etat **paused** (onPause)
  - Etat rare où l'Activity est partiellement visible.  
L'utilisateur ne peut pas interagir avec.
  - L'Activity ne peut pas exécuter de code

# Les états d'une Activity

- Etat **resumed** (onResume)
- Etat **paused** (onPause)
- Etat **stopped** (onStop)
  - Activity invisible à l'utilisateur
  - L'Activity ne peut pas exécuter de code
  - Process de l'application en background

## onCreate()

### Appelé quand ?

- A la création de l'Activity
- Lors d'un changement d'orientation d'écran

### A override pour quoi ?

- Mise en place de l'interface
- Initialisations qui ont besoin d'être faites une seule fois dans la vie de l'Activity

## onDestroy()

### Appelé quand ?

- Quand l'utilisateur appuie sur BACK
- Quand processus de l'application tué
- Pas appelée si crash de l'application

### A override pour quoi ?

- Arrêter les éventuels traitements en background
- Libérer les différentes ressources initialisées dans onCreate()

## onStart()

### Appelé quand ?

- A chaque fois que l'Activity revient au premier plan

## onStop()

### Appelé quand ?

- A chaque fois que l'Activity part en arrière plan

### A override pour quoi ?

- Initialisations de choses qui peuvent être arrêtées quand l'application part en background

### A override pour quoi ?

- Libérer les différentes ressources initialisées dans onStart()
- Faire des sauvegardes dans la base de données

## onResume( )

### Appelé quand ?

- A chaque fois que l'Activity est au premier plan

### A override pour quoi ?

- Mise à jour de l'interface
- Initialisations de choses qui peuvent être arrêtées quand l'Activity est partiellement cachée
- Effectuer des tâches légères car potentiellement appelée souvent

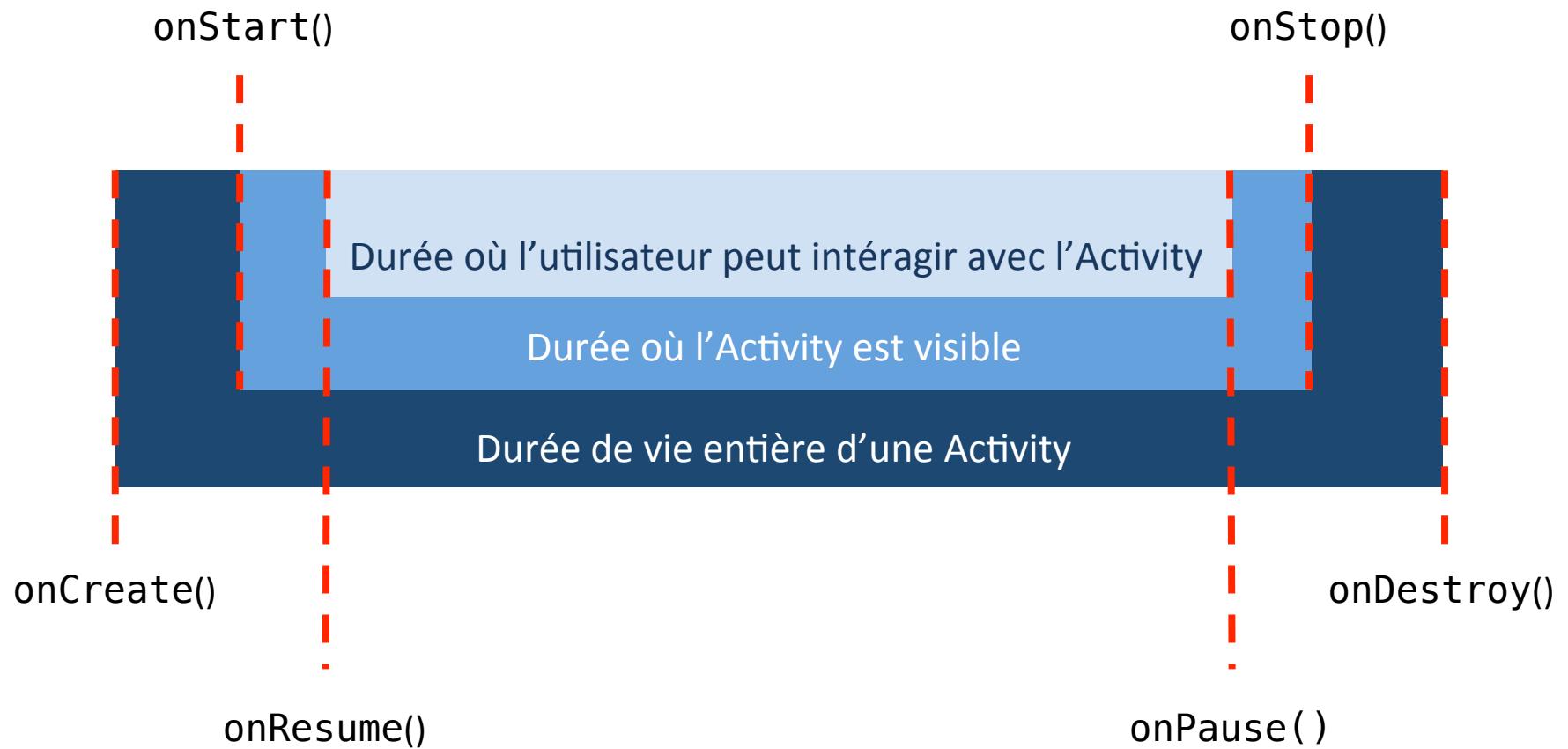
## onPause( )

### Appelé quand ?

- A chaque fois que l'Activity perd le focus

### A override pour quoi ?

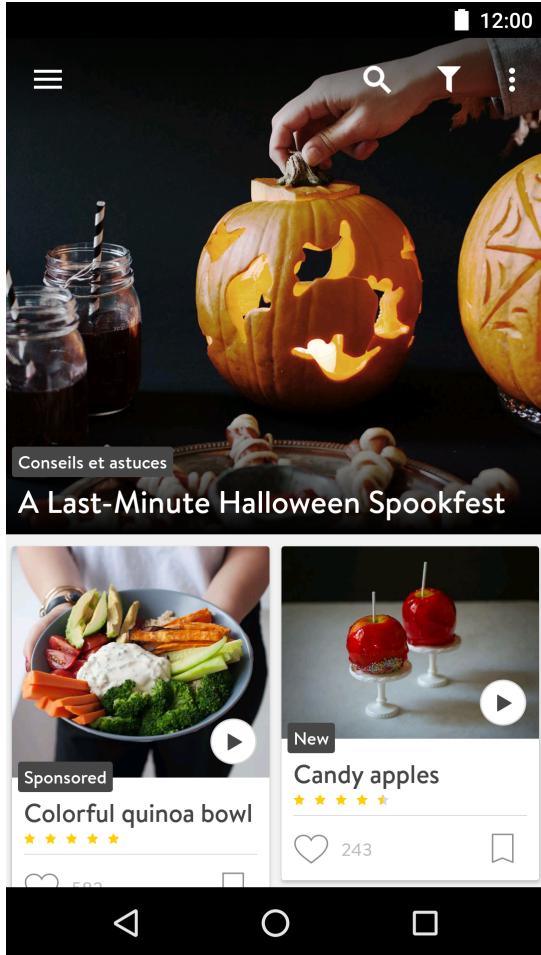
- Libérer les différentes ressources initialisées dans onResume()
- Effectuer des tâches légères car potentiellement appelée souvent



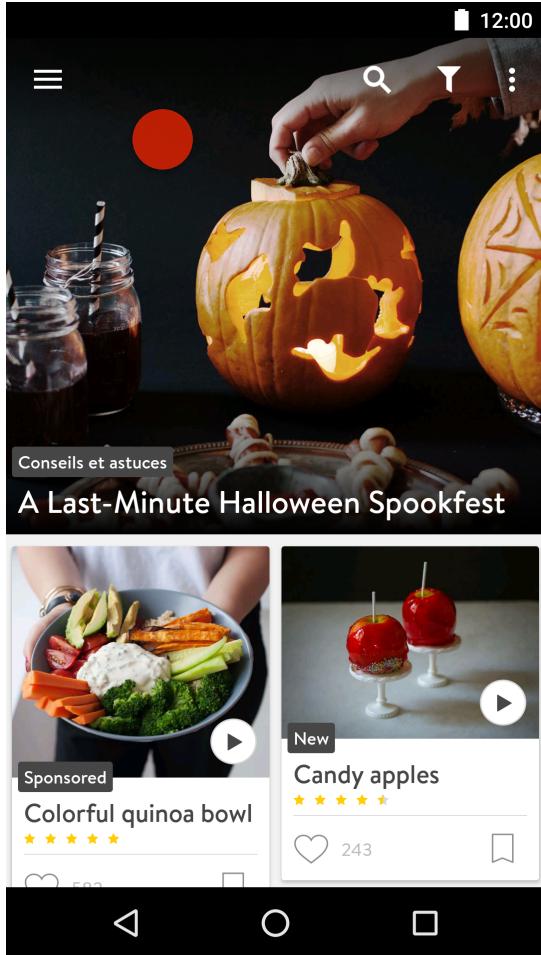
# Activity Lifecycle : bonnes pratiques

- Si initialisation dans **onCreate()**, nettoyage dans **onDestroy()**
- Si initialisation dans **onStart()**, nettoyage dans **onStop()**
- Si initialisation dans **onResume()**, nettoyage dans **onPause()**

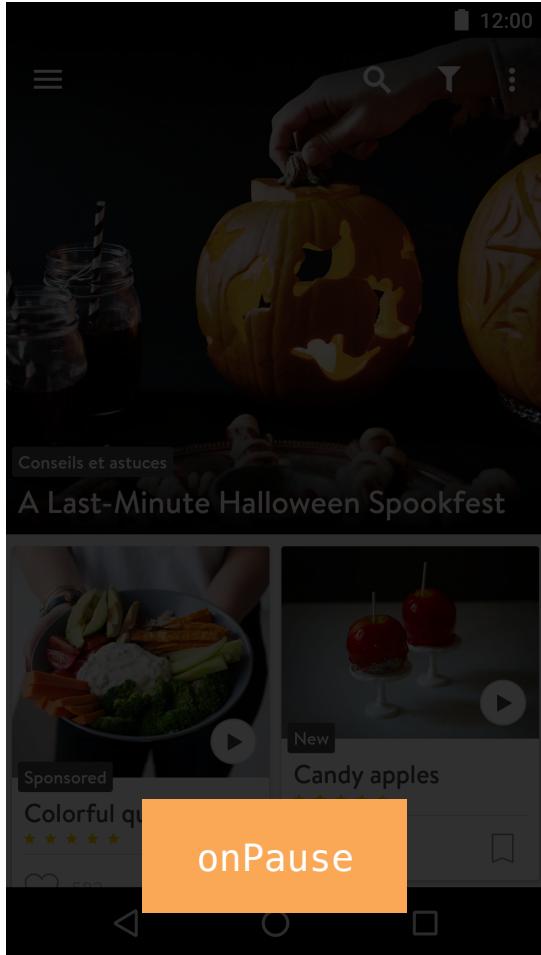
→ Toujours respecter les paires



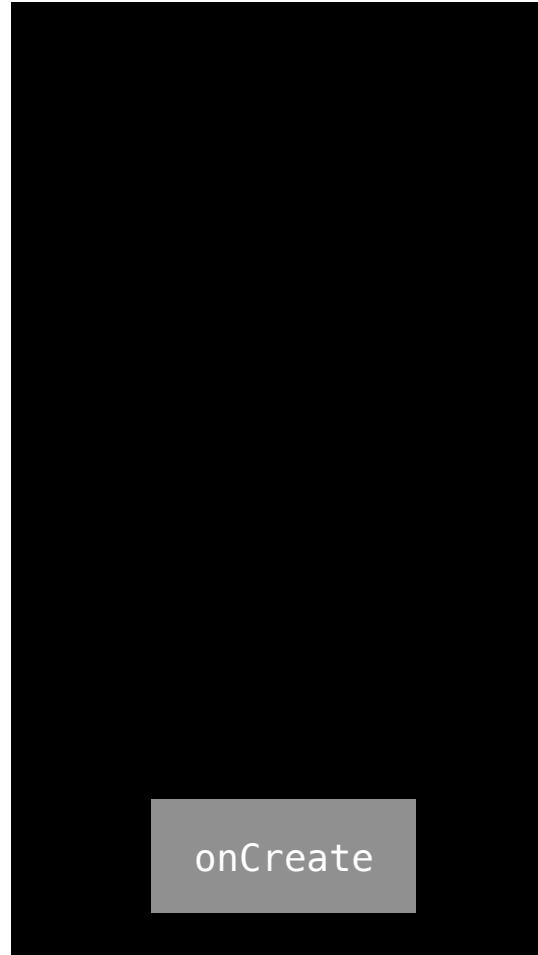
Activity principale « Liste de recettes ».



Tap sur image en en-tête



Activity suit son lifecycle



En parallèle création de  
l'Activity « Liste de recettes  
Halloween »

# Backstack



Activity suit son lifecycle



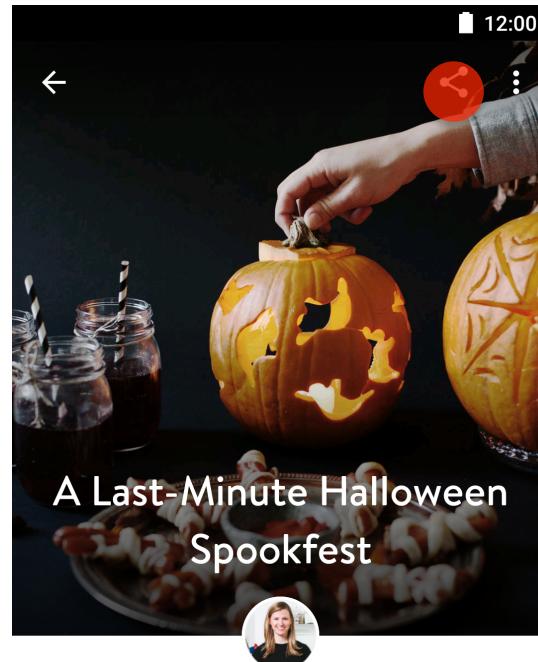
Activity suit son lifecycle

# Backstack

Activity « Liste de recettes »



Activity suit son lifecycle



Don't let t onResume eak up on

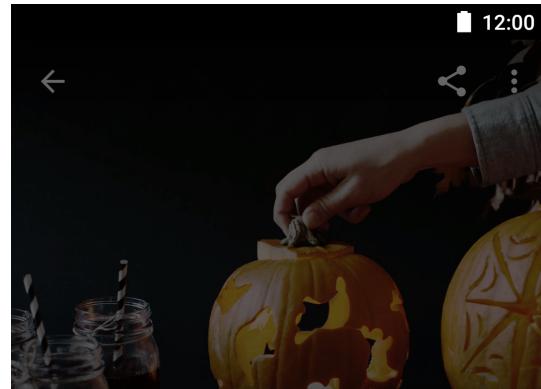
onResume



Tap sur partager la page

# Backstack

Activity « Liste de recettes »



Partager



PayPal



Gmail



Keep



Google+



Copier dans le  
presse-papiers



Hangouts



Messenger



Enregistrer  
dans Drive

onStop

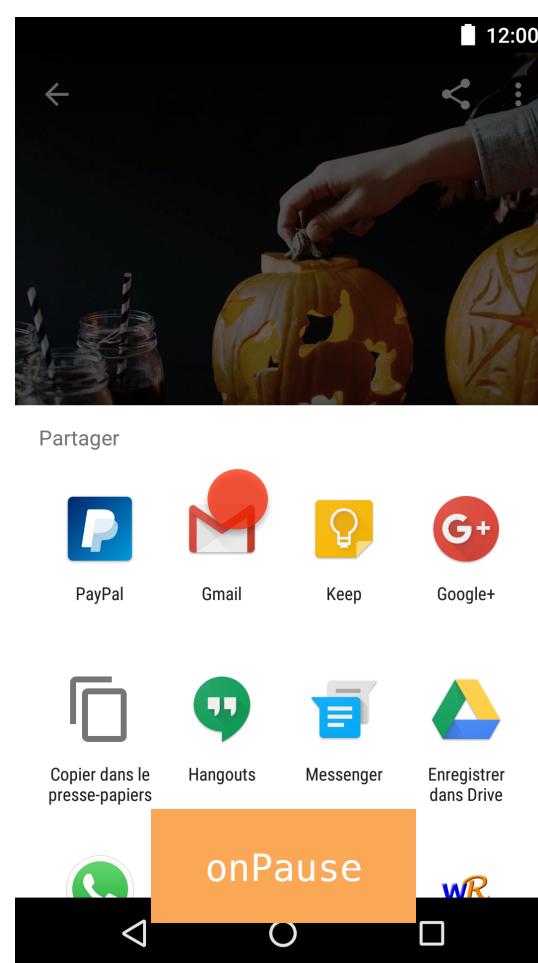
onPause

wR



Activity suit son lifecycle

# Backstack



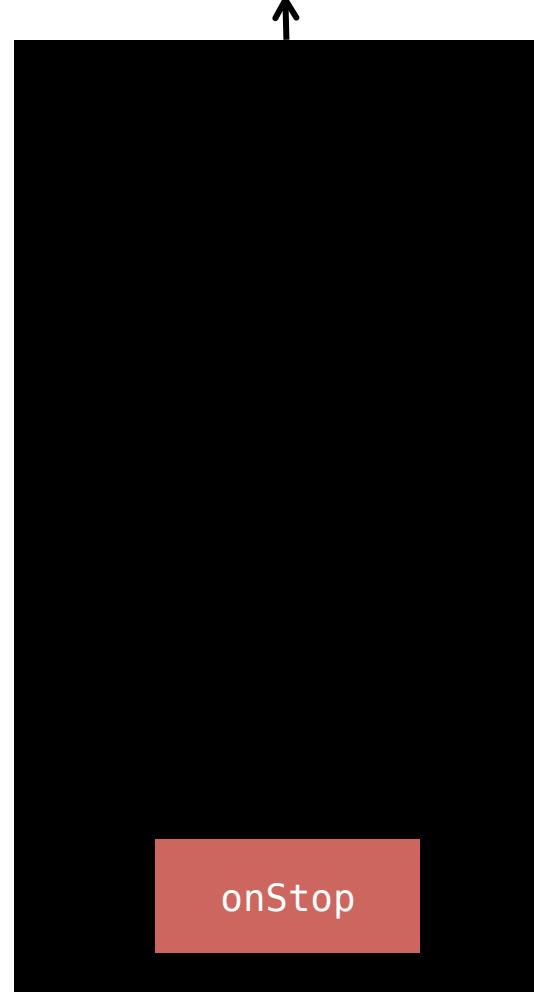
Activity suit son lifecycle

Tap sur Gmail

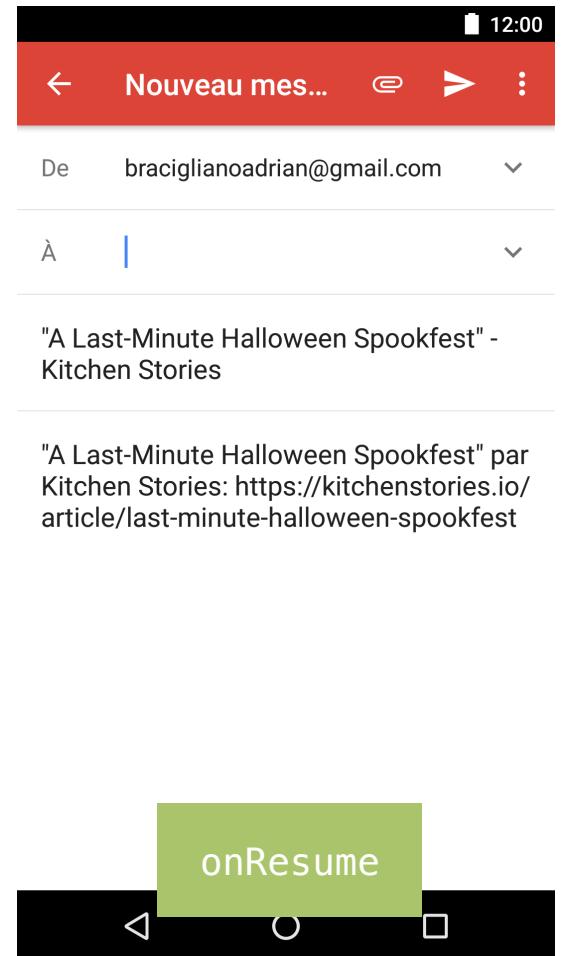
# Backstack



Activity suit son lifecycle



Activity suit son lifecycle

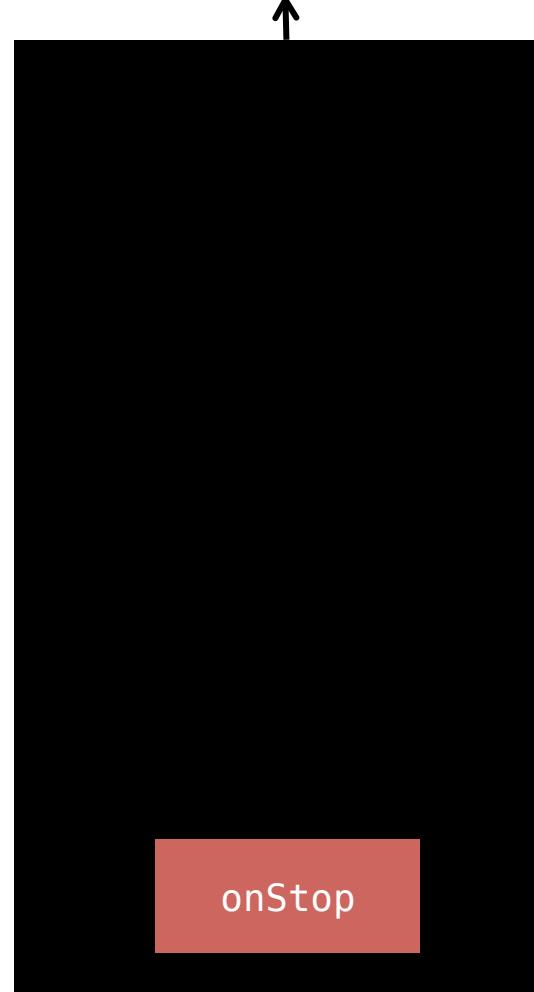


Activity de Gmail suit son lifecycle

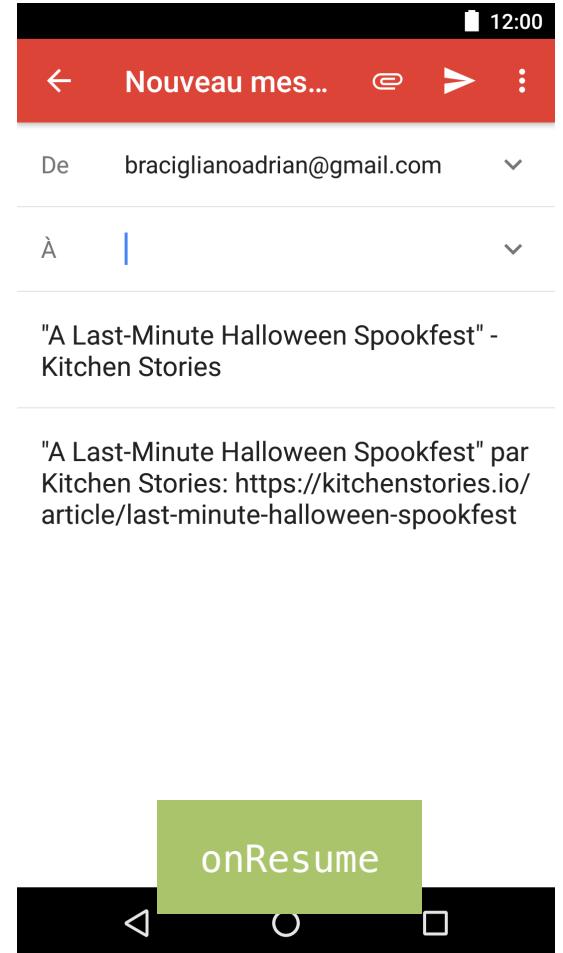
# Backstack



Activity suit son lifecycle



Activity suit son lifecycle

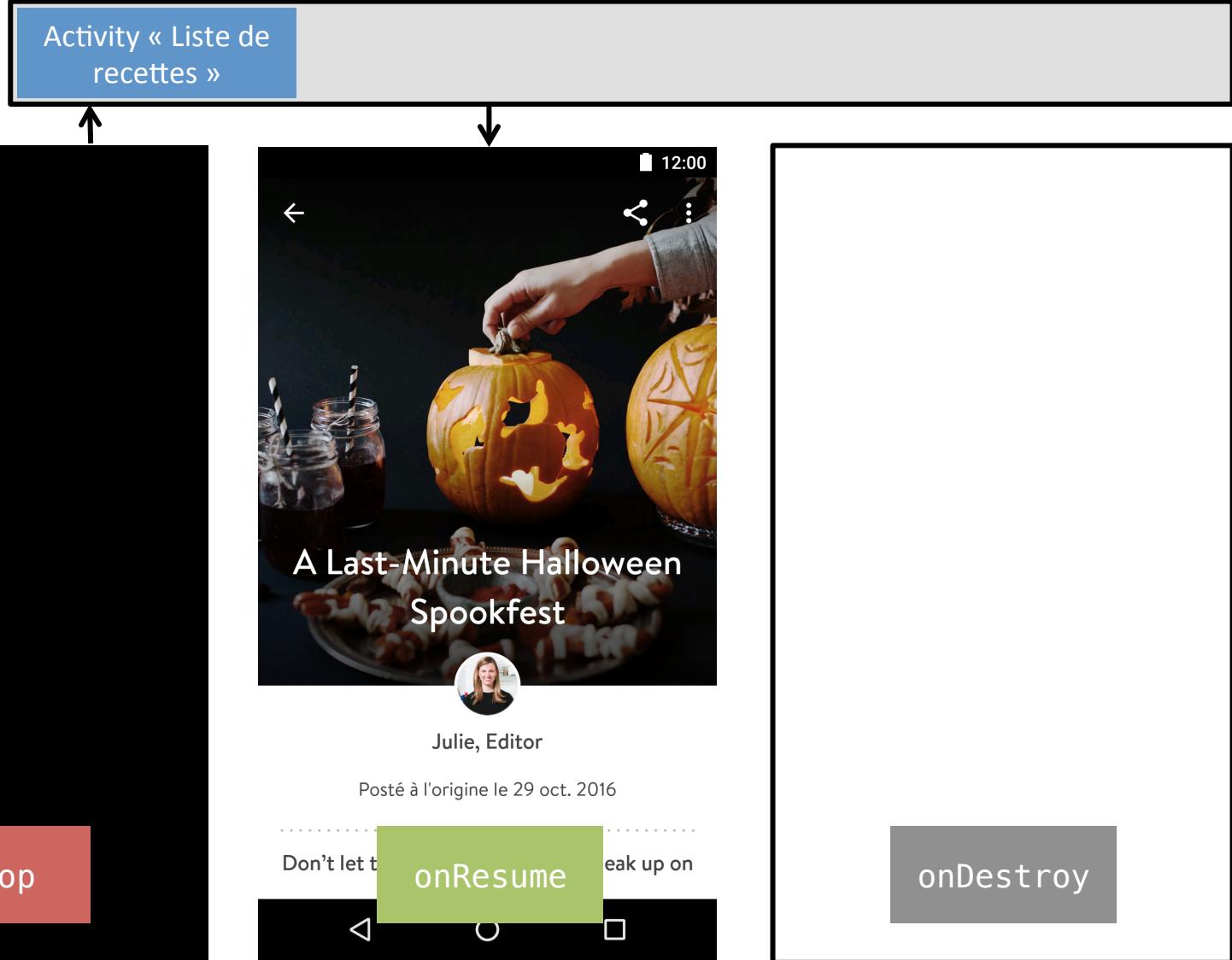


Appuie sur BACK

# Backstack



Activity suit son lifecycle



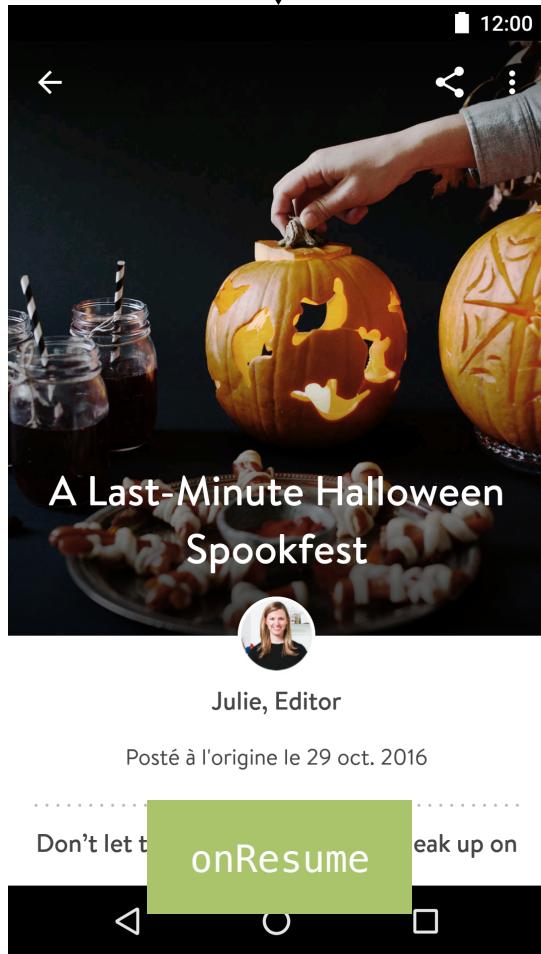
Activity suit son lifecycle

# Backstack

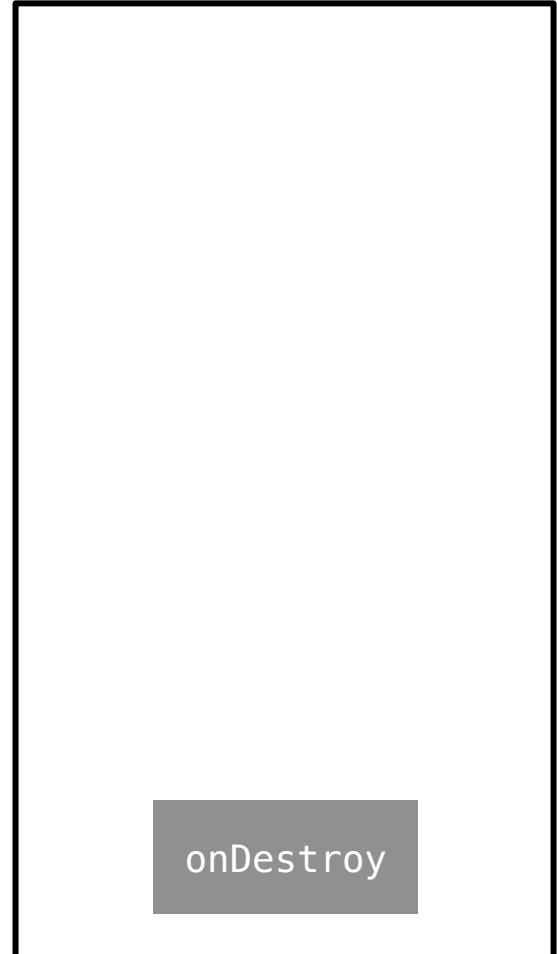
Activity « Liste de recettes »



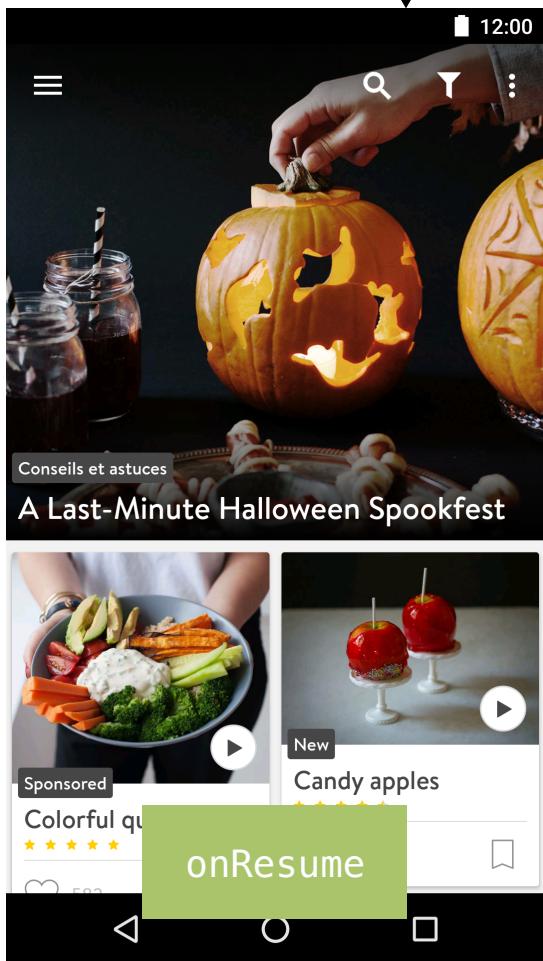
Activity suit son lifecycle



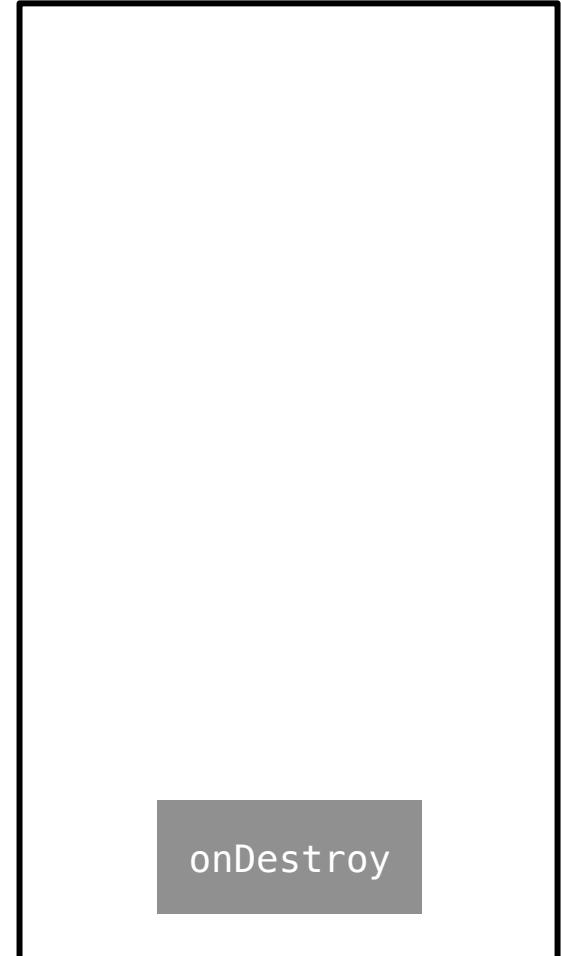
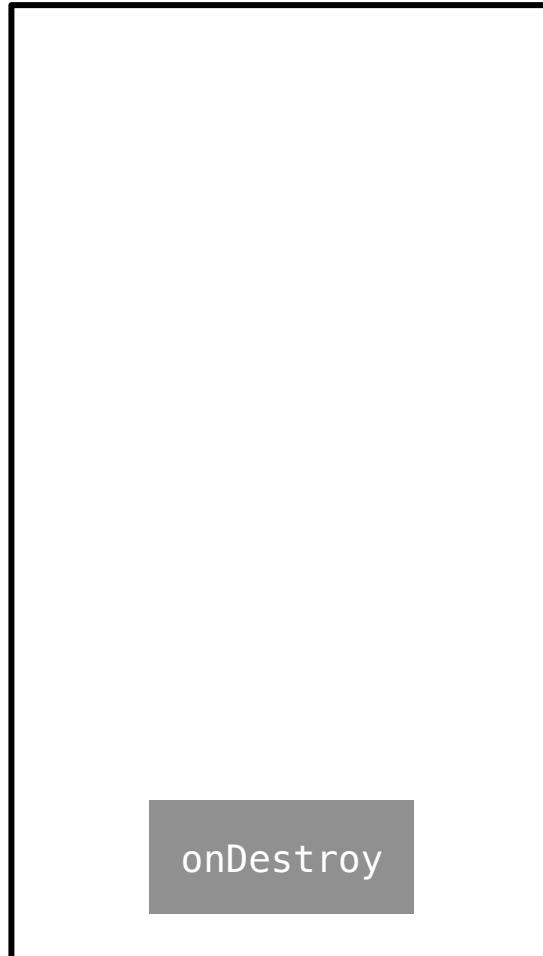
Appuie sur BACK



# Backstack



Activity suit son lifecycle



# Lancer une autre de nos Activities

```
public class MyActivity extends Activity {  
  
    @Override  
    protected void onCreate(Bundle b) {  
        super.onCreate(b);  
  
        ...  
        Intent intent = new Intent(this, MyActivity2.class);  
        startActivity(intent);  
    }  
}
```

# Lancer une autre de nos Activities

```
public class MyActivity extends Activity {  
  
    @Override  
    protected void onCreate(Bundle b) {  
        super.onCreate(b);  
  
        ...  
        Intent intent = new Intent(this, MyActivity2.class);  
        startActivity(intent);  
    }  
}
```

# Lancer une autre de nos Activities

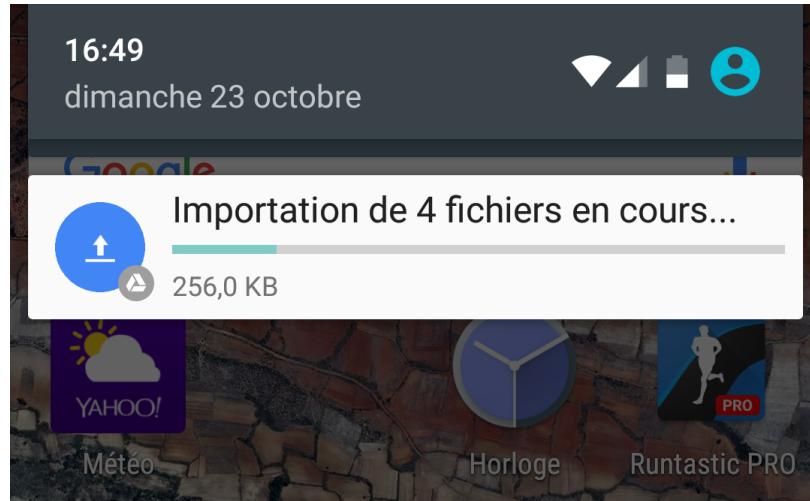
```
public class MyActivity extends Activity {  
  
    @Override  
    protected void onCreate(Bundle b) {  
        super.onCreate(b);  
  
        ...  
        Intent intent = new Intent(this, MyActivity2.class);  
        startActivity(intent);  
    }  
}
```

# Intent

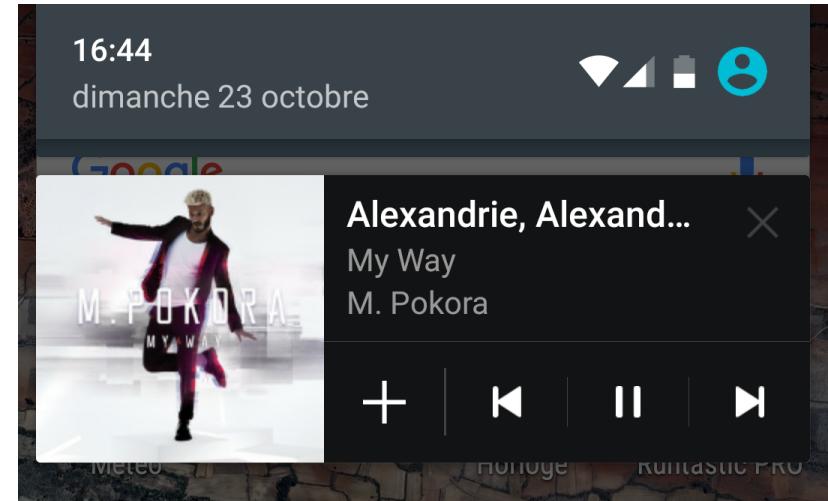
- Permet de lancer les principaux composants Android
  - Activities

# Intent

- Permet de lancer les principaux composants Android
  - Activities
  - Services : Composant sans interface graphique qui réalise des opérations en background



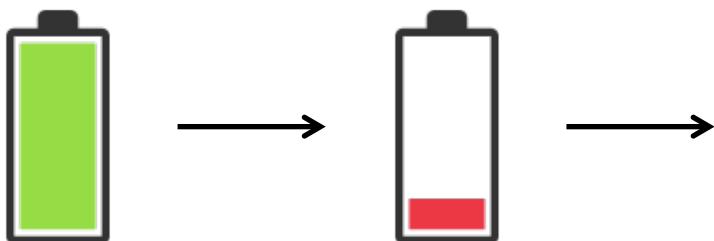
Service Google Drive importation de fichier



Service Spotify lecture de musique

# Intent

- Permet de lancer les principaux composants Android
  - Activities
  - Services
  - Broadcast : Message envoyé par le système ou une application et qui peut être réceptionné par n'importe quelle app



Lancement du broadcast  
`Intent.ACTION_BATTERY_LOW`

Envoi d'un broadcast par le système quand batterie presque vide

# Explicit intent

- Pour le lancement d'un composant dont on connaît exactement le nom

```
Intent intent = new Intent(this, MyActivity2.class);
```

Lancement de l'Activity MyActivity2

# Explicit intent

- Pour le lancement d'un composant dont on connaît exactement le nom

```
Intent intent = new Intent(this, MyActivity2.class);
intent.putExtra("key1", 1);
intent.putExtra("key2", true);
intent.putExtra("key3", my_object);
```

Lancement de l'Activity MyActivity2 en passant des données

# Explicit intent

```
public class MyActivity2 extends Activity {  
  
    @Override  
    protected void onCreate(Bundle b) {  
        super.onCreate(b);  
  
        Bundle bundle = getIntent().getExtras();  
        int key1 = bundle.getInt("key1");  
        boolean key2 = bundle.getBoolean("key2");  
        MyObj key3 = (MyObj) bundle.getSerializable("key3");  
    }  
}
```

Récupération des données envoyées à MyActivity2

# AndroidManifest.xml

```
<manifest package="com.ensiie" ...>  
    <application ...>  
        <activity android:name=".MyActivity2"/>  
    </application>  
</manifest>
```

Déclaration de l'Activity MyActivity2

# Implicit intent

- Pour une action qu'on souhaite réaliser et que notre application ne permet pas de faire
- Se défini par :
  - Action : L'action à réaliser
  - Data : La donnée sur laquelle réaliser l'action ou le type de donnée
  - Extras: Données supplémentaires pouvant être utilisées pour réaliser l'action

# Implicit intent

- Pour une action qu'on souhaite réaliser et que notre application ne permet pas de faire

```
Intent intent = new Intent();
intent.setAction(Intent.ACTION_SEND);
intent.putExtra(Intent.EXTRA_TEXT, "message");
intent.setType("text/plain");
```

Demande au système de lancer une application permettant d'envoyer le texte  
« message »

# Implicit intent

- Pour une action qu'on souhaite réaliser et que notre application ne permet pas de faire

```
Intent intent = new Intent();
intent.setAction(Intent.ACTION_DIAL);
intent.setData(Uri.parse("tel:" + phoneNumber));
```

Demande au système de lancer une application pour appeler un numéro de téléphone

# AndroidManifest.xml

```
<manifest package="com.ensiie" ...>  
  
    <application ...>  
        ...  
        <activity android:name="com.ensiie.MyActivity">  
            <intent-filter>  
                <action android:name="android.intent.action.SEND" />  
                <!-- Indique au système que cette Activity peut être lancée  
                    avec un implicit intent -->  
                <category android:name="android.intent.category.DEFAULT" />  
                <data android:mimeType="text/plain" />  
            </intent-filter>  
        </activity>  
    </application>  
  
</manifest>
```

Déclaration d'une Activity capable d'être lancée à partir d'un implicit intent contenant  
un message à envoyer

# Implicit intent



Articles à lire

 BFMTV.COM · Il y a 17 heures

Strasbourg: une voiture retournée sur les voies du tramway



Un véhicule termine sur le toit à Strasbourg, au milieu des rails du tramway, après une collision. (Pho...)

→ Plus d'articles

◀ ○ □

**Application Google Now**

# Implicit intent



Articles à lire

BFMTV.COM · Il y a 17 heures

Strasbourg: une voiture retournée sur les voies du tramway

Un véhicule termine sur le toit à Strasbourg, au milieu des rails du tramway, après une collision. (Pho...)



→ Plus d'articles

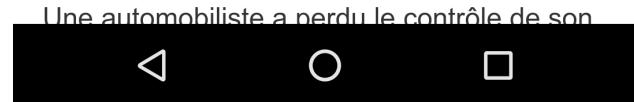
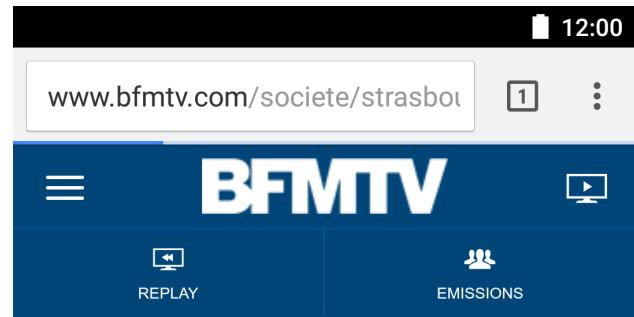
◀ ○ □

Tap sur l'article

# Implicit intent

Le téléphone possède uniquement l'application Google Chrome d'installée sur son téléphone :

1. Envoi d'un implicit intent par Google Now pour ouvrir une url dans un navigateur web
2. Le système recherche les apps avec un intent-filter éligible aux critères de l'intent
3. Une seule app trouvée, le système appelle la méthode `onCreate()` de l'Activity avec l'intent-filter éligible

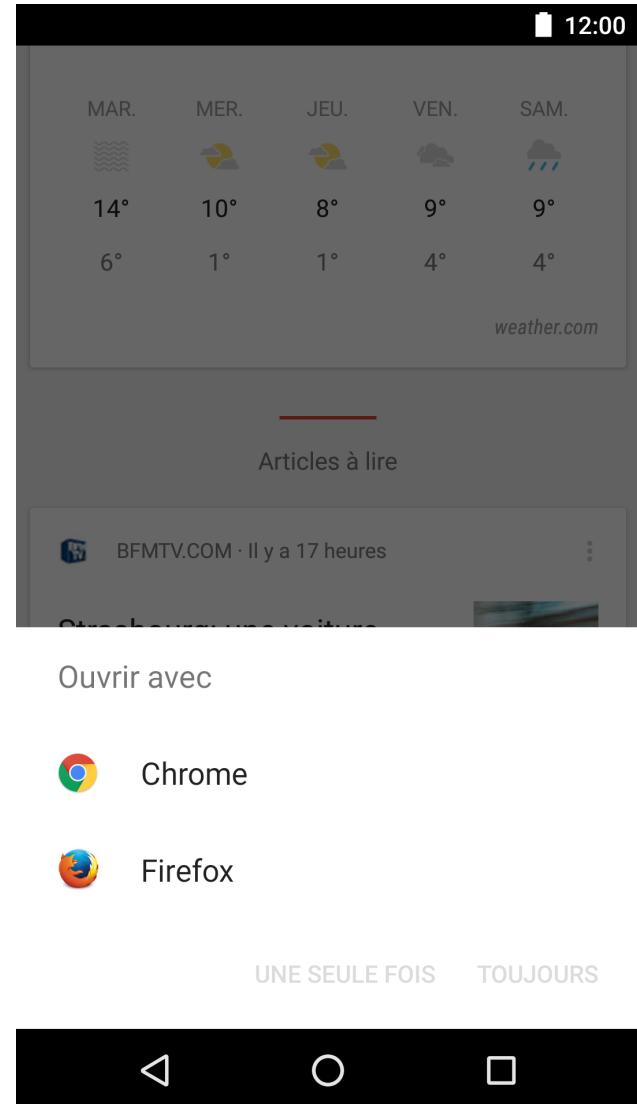


L'article s'ouvre dans Google Chrome

# Implicit intent

Le téléphone possède l'application Google Chrome et Firefox :

1. Envoi d'un implicit intent par Google Now pour ouvrir une url dans un navigateur web
2. Le système recherche les apps avec un intent-filter éligible aux critères de l'intent
3. Plusieurs apps trouvées, le système demande à l'utilisateur de choisir quelle application lancer



Fenêtre de choix affichée à l'utilisateur

# Implicit intent

```
public class MyActivity extends Activity {  
  
    @Override  
    protected void onCreate(Bundle b) {  
        super.onCreate(b);  
  
        Intent intent = new Intent();  
        intent.setAction(Intent.ACTION_VIEW);  
        intent.setDataAndType(Uri.parse(url), "text/html");  
  
        if (intent.resolveActivity(getApplicationContext()) != null) {  
            startActivity(intent);  
        } else {  
            Toast.makeText(this, "Aucun navigateur web installé  
sur le téléphone", Toast.LENGTH_SHORT).show();  
        }  
    }  
}
```

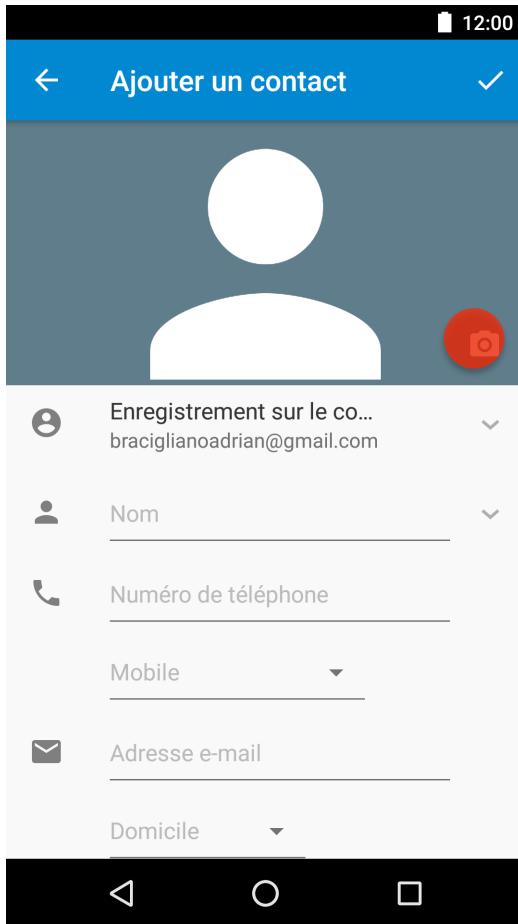
Test si une app peut gérer l'implicit intent

# Implicit intent

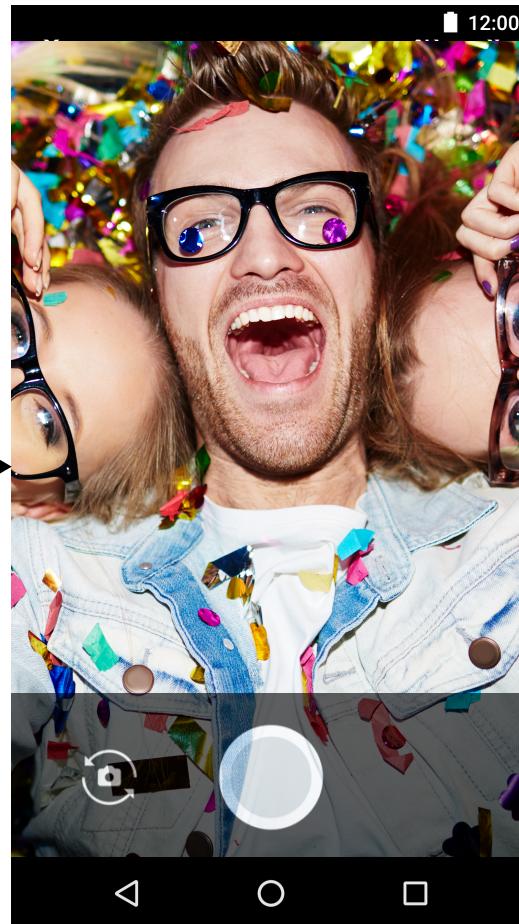
```
public class MyActivity extends Activity {  
  
    @Override  
    protected void onCreate(Bundle b) {  
        super.onCreate(b);  
  
        Intent intent = new Intent();  
        intent.setAction(Intent.ACTION_VIEW);  
        intent.setDataAndType(Uri.parse(url), "text/html");  
  
        if (intent.resolveActivity(getApplicationContext()) != null) {  
            startActivity(intent);  
        } else {  
            Toast.makeText(this, "Aucun navigateur web installé  
sur le téléphone", Toast.LENGTH_SHORT).show();  
        }  
    }  
}
```

Gestion du cas où aucune app ne peut gérer l'implicit intent

# Lancer une Activity et attendre une réponse

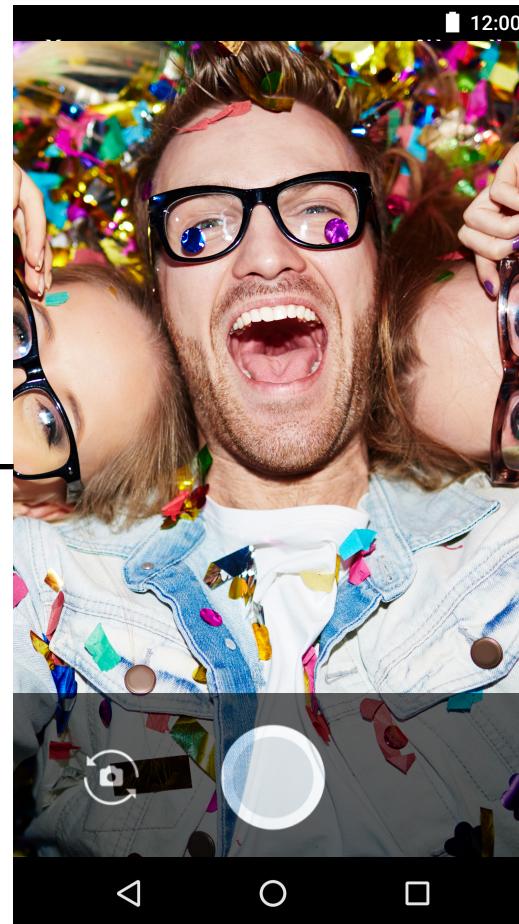
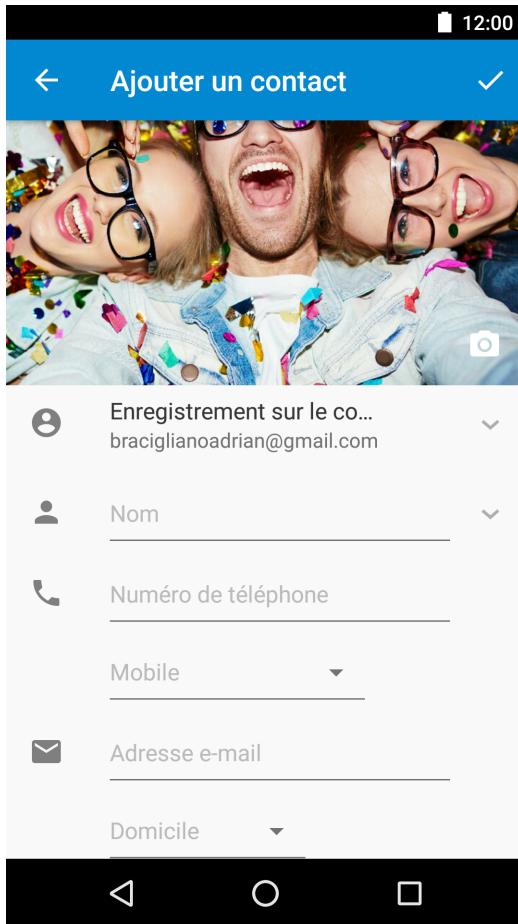


**Application Contact.**  
Tap sur icône photo



**Lancement implicit intent et ouverture application appareil photo**

# Lancer une Activity et attendre une réponse



**Retour sur application  
Contact avec photo prise.**

# Lancer une Activity et attendre une réponse

```
public class MyActivity extends Activity {  
    static final int REQUEST_PHOTO = 1;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
  
        Intent intent = new Intent(this, PhotosActivity);  
        startActivityForResult(intent, REQUEST_PHOTO);  
    }  
  
    @Override  
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
        if (requestCode == REQUEST_PHOTO && resultCode == RESULT_OK) {  
            Bundle bundle = data.getExtras();  
            Photo photo = (Serializable) bundle.getSerializable("photo");  
        }  
    }  
}
```

Utilisation de la méthode `startActivityForResult()` pour lancer une Activity et attendre la réponse

# Lancer une Activity et attendre une réponse

```
public class MyActivity extends Activity {  
    static final int REQUEST_PHOTO = 1;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
  
        Intent intent = new Intent(this, PhotosActivity);  
        startActivityForResult(intent, REQUEST_PHOTO);  
    }  
  
    @Override  
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
        if (requestCode == REQUEST_PHOTO && resultCode == RESULT_OK) {  
            Bundle bundle = data.getExtras();  
            Photo photo = (Serializable) bundle.getSerializable("photo");  
        }  
    }  
}
```

Passage d'un token pour identifier le retour de l'Activity PhotosActivity

# Lancer une Activity et attendre une réponse

```
public class MyActivity extends Activity {  
    static final int REQUEST_PHOTO = 1;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
  
        Intent intent = new Intent(this, PhotosActivity);  
        startActivityForResult(intent, REQUEST_PHOTO);  
    }  
  
    @Override  
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
        if (requestCode == REQUEST_PHOTO && resultCode == RESULT_OK) {  
            Bundle bundle = data.getExtras();  
            Photo photo = (Serializable) bundle.getSerializable("photo");  
        }  
    }  
}
```

Override de la méthode `onActivityResult()` pour récupération de la réponse de l'Activity `PhotosActivity`

# Lancer une Activity et attendre une réponse

```
public class MyActivity extends Activity {  
    static final int REQUEST_PHOTO = 1;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
  
        Intent intent = new Intent(this, PhotosActivity);  
        startActivityForResult(intent, REQUEST_PHOTO);  
    }  
  
    @Override  
    protected void onActivityResult(int requestCode, int resultCode, Intent  
data) {  
        if (requestCode == REQUEST_PHOTO && resultCode == RESULT_OK) {  
            Bundle bundle = data.getExtras();  
            Photo photo = (Serializable) bundle.getSerializable("photo");  
        }  
    }  
}
```

Récupération de la photo choisie par l'utilisateur

# Lancer une Activity et attendre une réponse

```
public class PhotosActivity extends Activity {  
  
    @Override  
    protected void onCreate(Bundle b) {  
        super.onCreate(b);  
  
        Intent intent = new Intent();  
        intent.putExtra("photo", photo);  
        setResult(RESULT_OK, intent);  
        finish();  
    }  
}
```

Passage de la photo choisie par l'utilisateur dans la réponse de l'Activity

# Lancer une Activity et attendre une réponse

```
public class PhotosActivity extends Activity {  
  
    @Override  
    protected void onCreate(Bundle b) {  
        super.onCreate(b);  
  
        Intent intent = new Intent();  
        intent.putExtra("photo", photo);  
        setResult(RESULT_OK, intent);  
        finish();  
    }  
}
```

**Fermeture de l'Activity**

# Fragment

- Partie modulaire de l'interface d'une Activity
- Possède son propre lifecycle lié à celui de l'Activity
- Utilisé pour créer des interfaces plus dynamiques et flexibles

## All inboxes



Anissa Lee  
origamifan26@gmail.com

C



All inboxes 3

Primary 1

Social 1 new

Promotions 2 new

Updates 1 new

### All labels

Starred 1

Important 7



Tim Greer 2:52 PM  
Favorite Christmas movie?  
Putting together a list on my top christmas... ☆



Kelsey Green 2:49 PM  
Softball game Tuesday night  
I think we're going up against team "St. El... ☆



Merrill Anovick 2:37 PM  
**Wedding plus one**  
Hey, do you know if Aileen is still available... ☆



Lindsay Carter, Anissa Lee 1:57 PM  
Dinner on Sunday?  
Perfect! I'll put us down for 7:30pm.... ☆



Keri Andersen Jan 22  
**Snacks**  
What should I bring tonight snack wise? I t... ☆



Michael Potts Jan 22  
**Ski Trip**  
Hey, are you still in for next week's ski trip?... ☆



Brett, me 2 8/29/2014  
Visiting the San Diego Zoo  
Yes! Swing by to see the monkeys! On R...



Favorite Christmas movie?

Tim Greer  
to me  
2:52 PM [View details](#)

Putting together a list on my top christmas mo  
outspoken expert I was hoping you could weigl  
guidance.



Reply



Reply all



Application Gmail sur tablette

## All inboxes



Anissa Lee  
origamifan26@gmail.com

C



All inboxes 3

Primary 1

Social Fragment 1 1 new

Promotions 2 new

Updates 1 new

All labels

Starred 1

Important 7



Tim Greer  
Favorite Christmas movie?  
Putting together a list on my top christmas... ☆



Kelsey Green  
Softball game Tuesday night  
I think we're going up against team "St. El..." ☆



M Merrill Anovick  
Wedding plus one  
Hey, do you know if Aileen is still available... ☆



Lindsay Carter, Anissa Lee 3  
Dinner on Sunday?  
Perfect! I'll put it down for 7:30pm.... Inbox ☆



Keri Andersen  
Snacks  
What should I bring tonight snack wise? I t... ☆



Michael Potts  
Ski Trip  
Hey, are you still in for next week's ski trip?... ☆



Brett, me 2  
Visiting the San Diego Zoo  
Yes! Swing by to see the monkeys! On R...



Favorite Christmas movie? Inbox



Tim Greer  
to me  
2:52 PM View details

Putting together a list on my top christmas mo  
outspoken expert I was hoping you could weigh  
guidance.

Fragment 3 Reply Reply all



Application Gmail sur tablette



## Primary



### Social

Google+

1 new



### Promotions

Zagat, Google Offers

2 new



### Updates

Google Play

1 new



Andy Brown

1:45 PM

Bring Your Parents to Work Day!

Hey! What do you think about a...

Work



Keri Anderson

1:39 PM

Picture from last Saturday

Check out the new friend we made, Me...



Regis, Peter, Rachel 3

Sep 29

Board game night?

Sunday works! If you can get Dex...

Fun



Aruna Knight

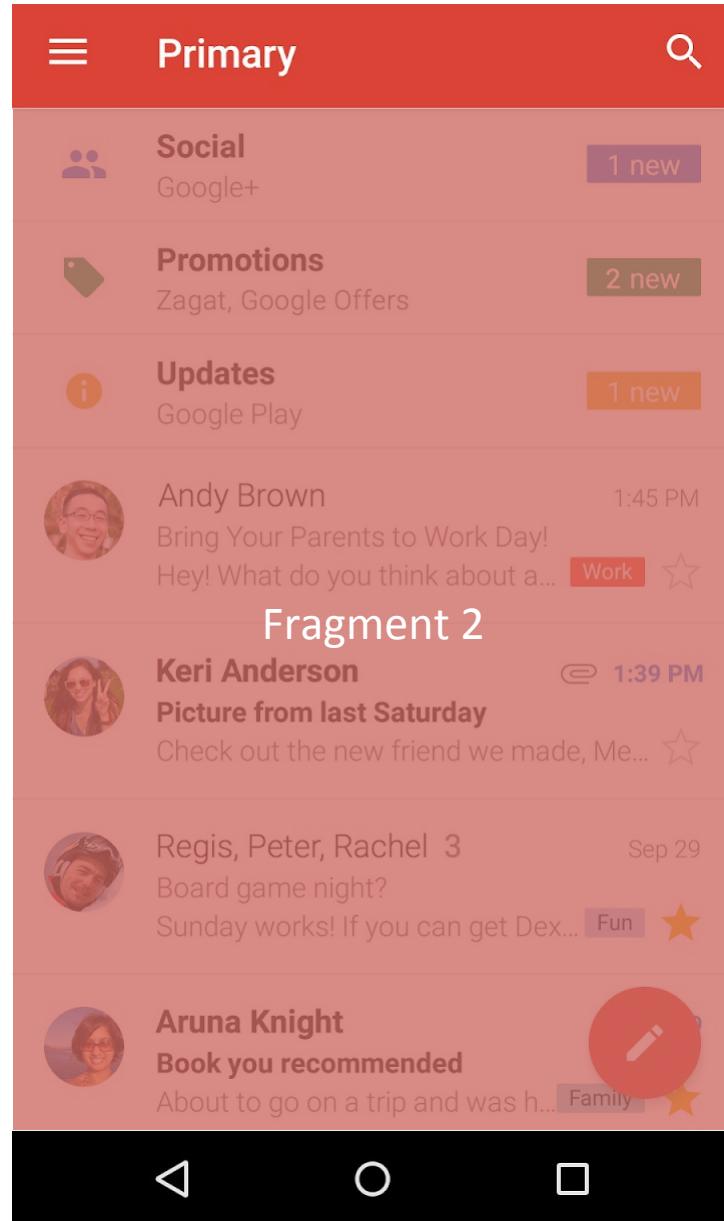
Book you recommended

About to go on a trip and was h...

Family



Application Gmail sur smartphone



Application Gmail sur smartphone