Programmation Android

III. `android.app.Activity`

Plan

Plan

Activity

Définition

- Une activité (sous classe d'**Activity**) représente un unique écran de l'UI d'une application Android
- Une application contient en général plusieurs activités. Par exemple, une application de courriel : une activité pour la composition, une activité pour la lecture, indépendante l'une de l'autre.
- Ces activités peuvent être lancées par d'autres applications. Par exemple pour partager une image (**gallery app**) via un courriel (**mailer app**)
- Tout l'intérêt d'Android est de faciliter l'utilisation de l'ensemble des composants présents dans le système, et donc notamment des différentes activités fournies par les applications.

Les activités d'une application doivent être déclarées dans le manifest :

Dans AndroidManifest.xml

To declare your activity, open your manifest file and add an <activity> element as a child of the <application> element. For example:

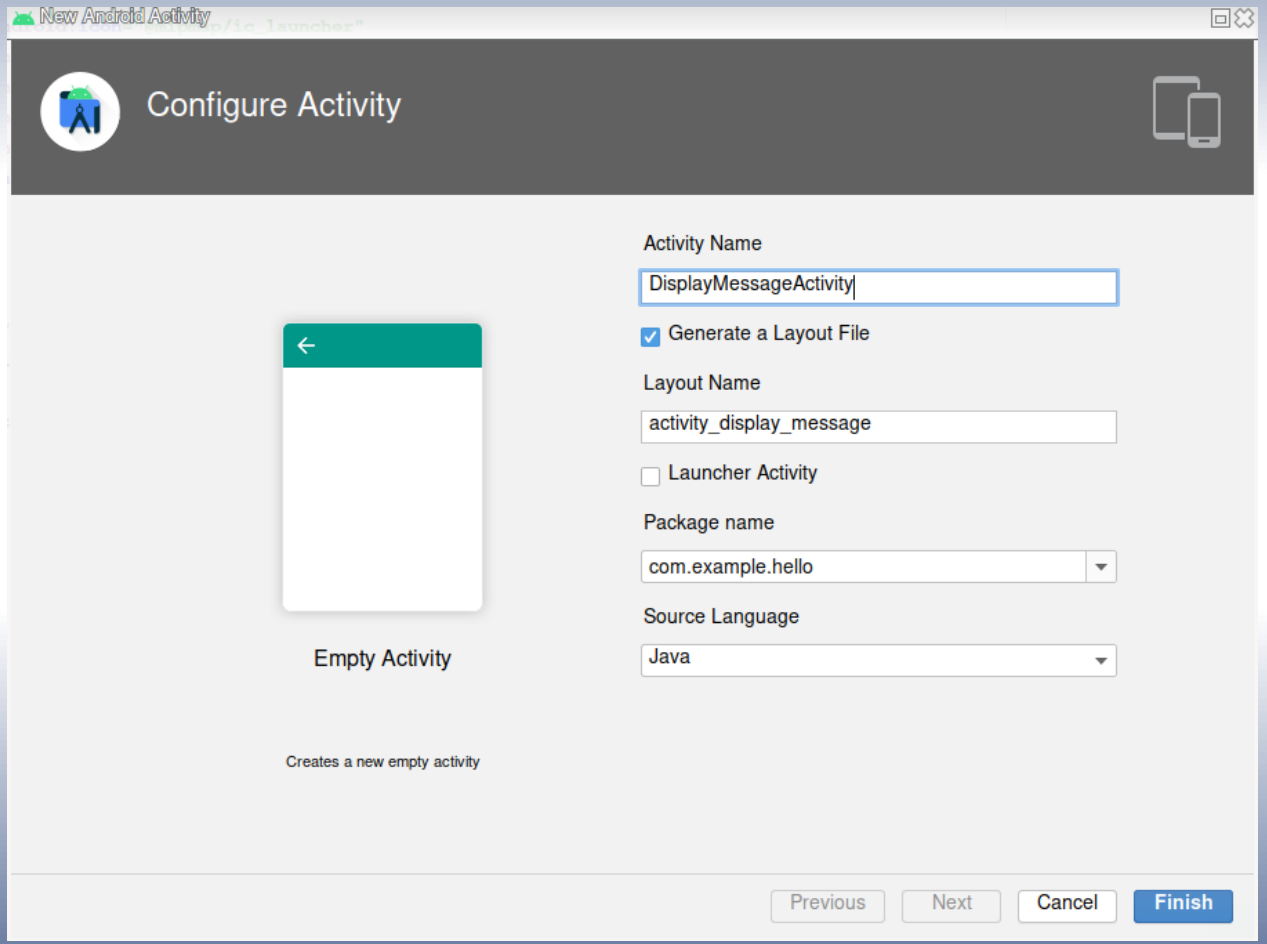
```
<manifest ... >
  <application ... >
      <activity android:name=".ExampleActivity" />
      ...
  </application ... >
  ...
</manifest >
```

The only required attribute for this element is android:name, which specifies the class name of the activity. You can also add attributes that define activity characteristics such as label, icon, or UI theme.

Plan

Création d'une activité

Bouton droit sur un package → New → Activity

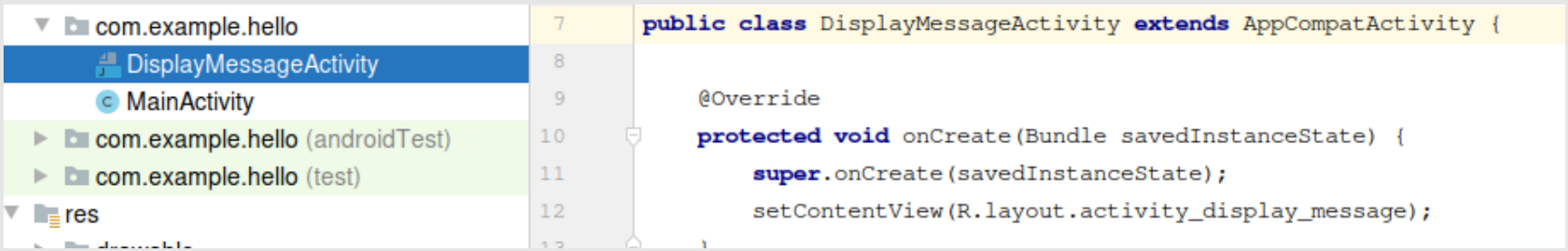


Résultat dans le manifest : création d'une nouvelle balise activity

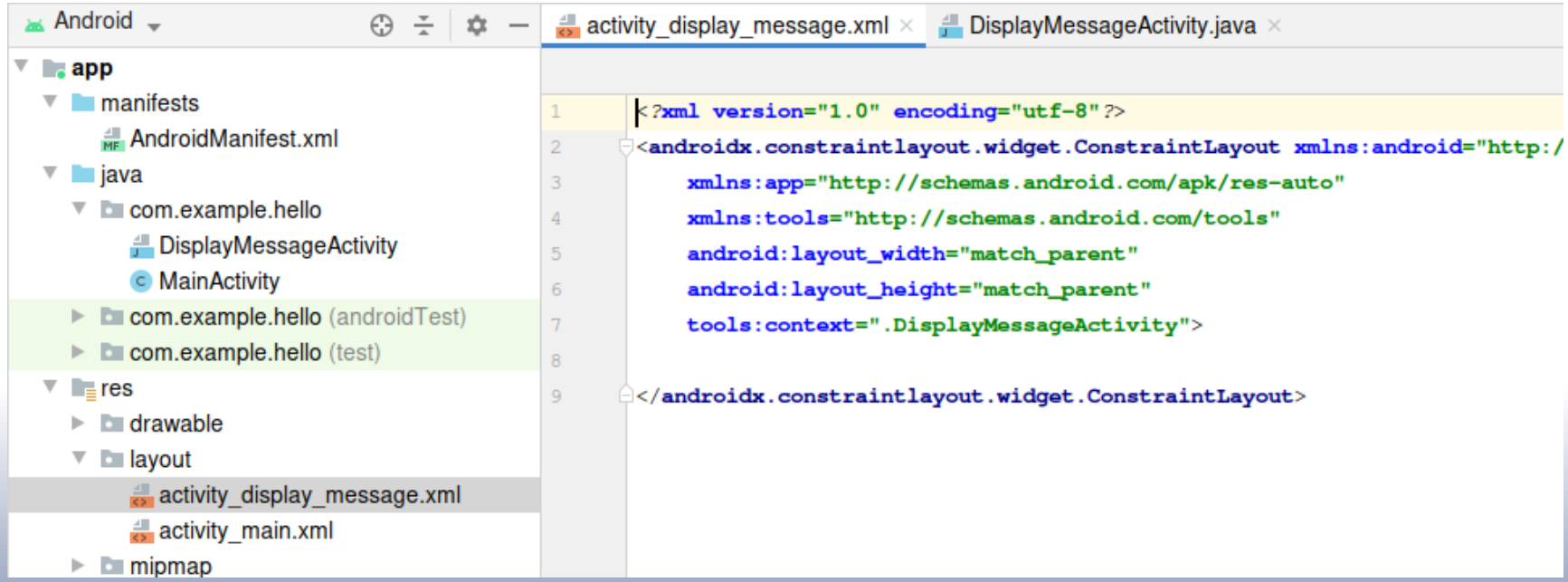
```
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/Theme.Hello">
    <activity android:name=".DisplayMessageActivity"></activity>
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />

            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
</application>
```

Résultat code Java : nouvelle classe

DisplayMessageActivity.java



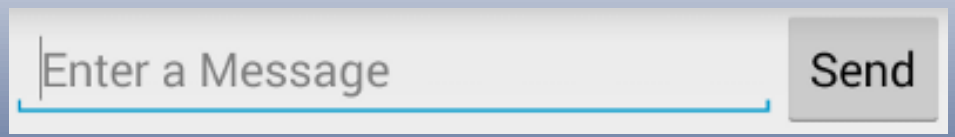
Résultat code XML : création automatique de l'UI de l'activité (layout)



Rappel : activité Main (cours précédent)

activity_main.xml | main.xml | MainActivity.java

```
<LinearLayout 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:orientation="horizontal" >

    <EditText
        android:id="@+id/edit_message"
        android:layout_weight="1"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:hint="@string/edit_message" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/button_send" />

</LinearLayout>
```



Exemple : à partir d'un bouton placé dans l'activité principale

Rappel : gestion clique solution 2

```
public class MainActivity extends ActionBarActivity {

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

        Button button = (Button) findViewById(R.id.button_send);
        button.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                System.out.println("button send");
            }
        });
```

Lancement de l'activité

Méthode `android.app.Activity.startActivity(Intent)`

```
Button button = (Button) findViewById(R.id.button_send);
button.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        System.out.println("button send");
        startactivity
    }
});
```

- startActivity(Intent intent) : void - Activity
- startActivity(Intent intent, Bundle options) : void - Activity
- startActivityForResult(Intent intent, int requestCode) : void - FragmentActivity
- startActivityForResult(Intent intent, int requestCode, Bundle options) : void - Activity
- startActivityFromChild(Activity child, Intent intent, int requestCode) : void - Activity
- startActivityFromChild(Activity child, Intent intent, int requestCode, Bundle options) : void - A
- startActivityFromFragment(Fragment fragment, Intent intent, int requestCode) : void - Activit

Lancement de l'activité

Création d'un Intent *explicite*, utilisation du constructeur `android.content.Intent.Intent(Context, Class<?>)`

```
Button button = (Button) findViewById(R.id.button_send);
button.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        System.out.println("button send");
        startActivity(new intent);
    }
});
```

Intent() - android.content.Intent
Intent(Intent o) - android.content.Intent
Intent(String action) - android.content.Intent
Intent(Context packageContext, Class<?> cls) - android.con
Intent(String action, Uri uri) - android.content.Intent
Intent(String action, Uri uri, Context packageContext, Class
Intent - android.content
IntentFilter() - android.content.IntentFilter
IntentFilter(IntentFilter o) - android.content.IntentFilter
IntentFilter(String action) - android.content.IntentFilter

Create an intent for a specific component. All other fields (action, data, type, class) are null, though they can be modified later with explicit calls. This provides a convenient way to create an intent that is intended to execute a hard-coded class name, rather than relying on the system to find an appropriate class for you; see setComponent for more information on the repercussions of this.

Parameters:

packageContext A Context of the application package implementing this class.

cls The component class that is to be used for the intent.

See Also:

setClass

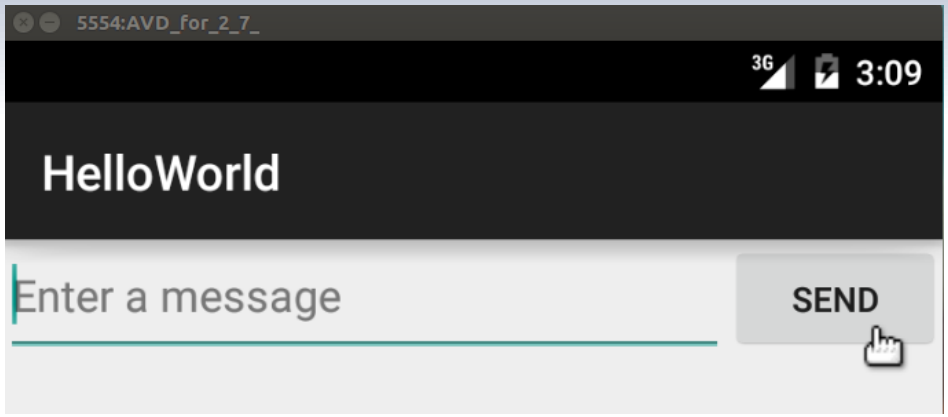
Lancement de l'activité

Création d'un Intent, utilisation du constructeur `android.content.Intent.Intent(Context, Class<?>)`

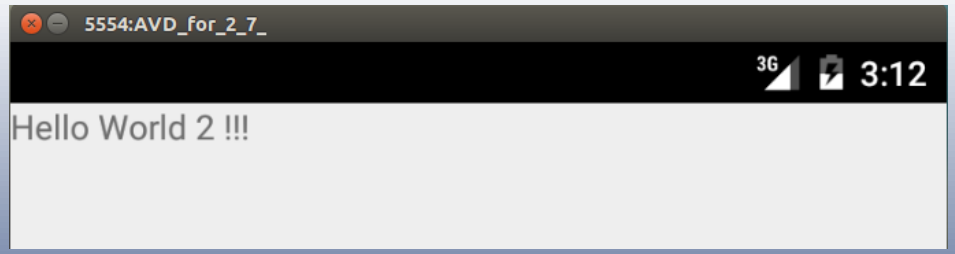
```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    Button button = (Button) findViewById(R.id.button_send);
    button.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            System.out.println("button send");
            startActivity(new Intent(MainActivity.this, DisplayMessageActivity.class));
        }
    });
```

Émetteur classe de l'activité

Résultat : *DisplayMessageActivity* est démarée au clique sur le bouton send



⇒



Plan

À propos de l'objet Intent

Principes d'un Intent

- objet contenant des informations permettant de *faire quelque chose* avec d'autres éléments du système
- Par exemple, faire la demande de **démarrage d'une activité**

android.content.Intent.Intent(Context, Class<?>)

- paramètre 1 : un objet de type `android.content.Context`. `android.app.Activity` est sous classe de contexte
- paramètre 2 : un objet de type `java.lang.Class` : la classe correspondant au composant visé, e.g. une activité

À propos de l'objet Intent

L'activité principale est démarée grâce à un Intent

HelloWorld Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="fr.iutmontp.helloworld"
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="21" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".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>
        <activity android:name=".DisplayMessageActivity"></activity>
    </application>

</manifest>
```

Communication de données via un Intent

android.content.Intent.putExtra(String, String)

```
public class MainActivity extends ActionBarActivity {

    //externalisation : clé pour la valeur de l'intent
    public static final String MESSAGE = "hello3";

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

        Button button = (Button) findViewById(R.id.button_send);
        button.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                System.out.println("button send");
                Intent myIntent = new Intent(MainActivity.this, DisplayMessageActivity.class);
                myIntent.putExtra(MESSAGE, "Hello World 3 !!!!");
                startActivity(myI
            }
        });
```

```
MainActivity
MESSAGE : S
onCreate(Bu
new OnClic
onClick(V
sendMessag
onCreateOpI
sendMessag
onOptionsIte
```

```
putExtra(String name, Serializable value) : Intent - Intent
putExtra(String name, short value) : Intent - Intent
putExtra(String name, short[] value) : Intent - Intent
putExtra(String name, String value) : Intent - Intent
putExtra(String name, String[] value) : Intent - Intent
putExtras(Bundle extras) : Intent - Intent
putExtras(Intent src) : Intent - Intent
```

Add extended data to the intent. The name must include a package prefix, for example the app com.android.contacts would use names like "com.android.contacts.ShowAll".

Parameters:

name The name of the extra data, with package prefix.

value The String data value.

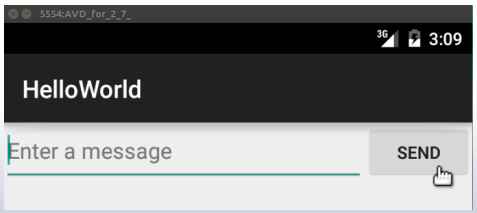
Returns:

Returns the same Intent object, for chaining multiple calls into a single statement.

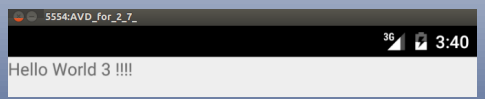
See Also:

Affichage des données transférées dans *DisplayMessageActivity*

```
public class DisplayMessageActivity extends Activity {

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

        TextView textView = (TextView) findViewById(R.id.textView1);
        String message = getIntent().getExtras().getString(MainActivity.MESSAGE);
        //résultat identique :
        message = getIntent().getStringExtra(MainActivity.MESSAGE);
        textView.setText(message);
    }
}
```



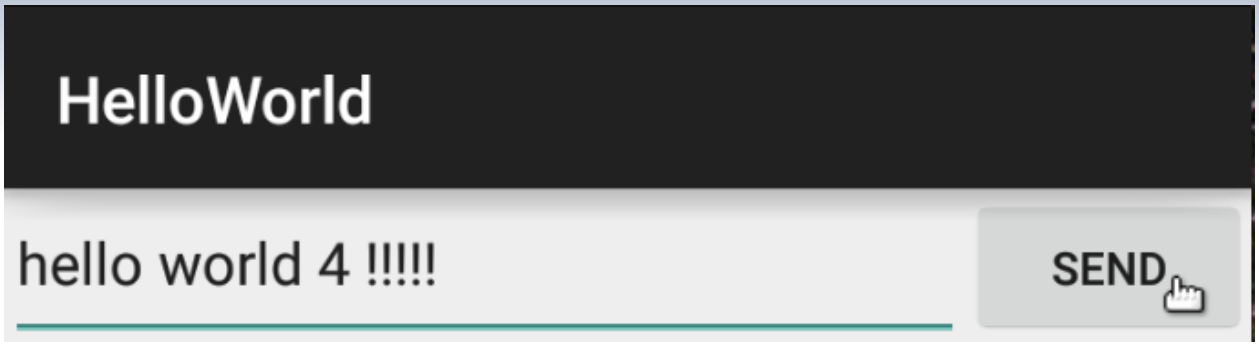
⇒



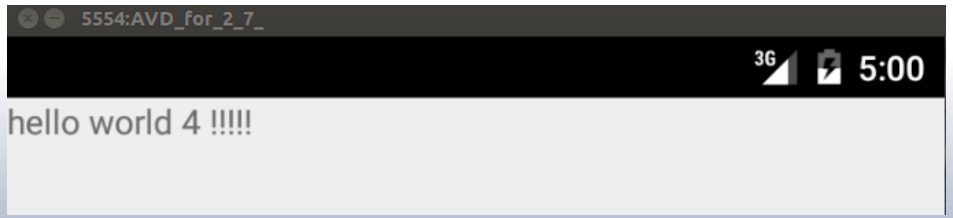
Transfert des données de la vue

```
public class MainActivity extends ActionBarActivity {

    //externalisation : clé pour la valeur de l'intent
    public static final String MESSAGE = "hello3";

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

        final TextView textView = (TextView) findViewById(R.id.edit_message);

        Button button = (Button) findViewById(R.id.button_send);
        button.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                System.out.println("button send");
                Intent myIntent = new Intent(MainActivity.this, DisplayMessageActivity.class);
                myIntent.putExtra(MESSAGE, textView.getText().toString());
                startActivity(myIntent);
            }
        });
```

Transfert des données de la vue



$\Rightarrow$



Création d'une vue dans le code Java : nouvelle activité AfficheMessage

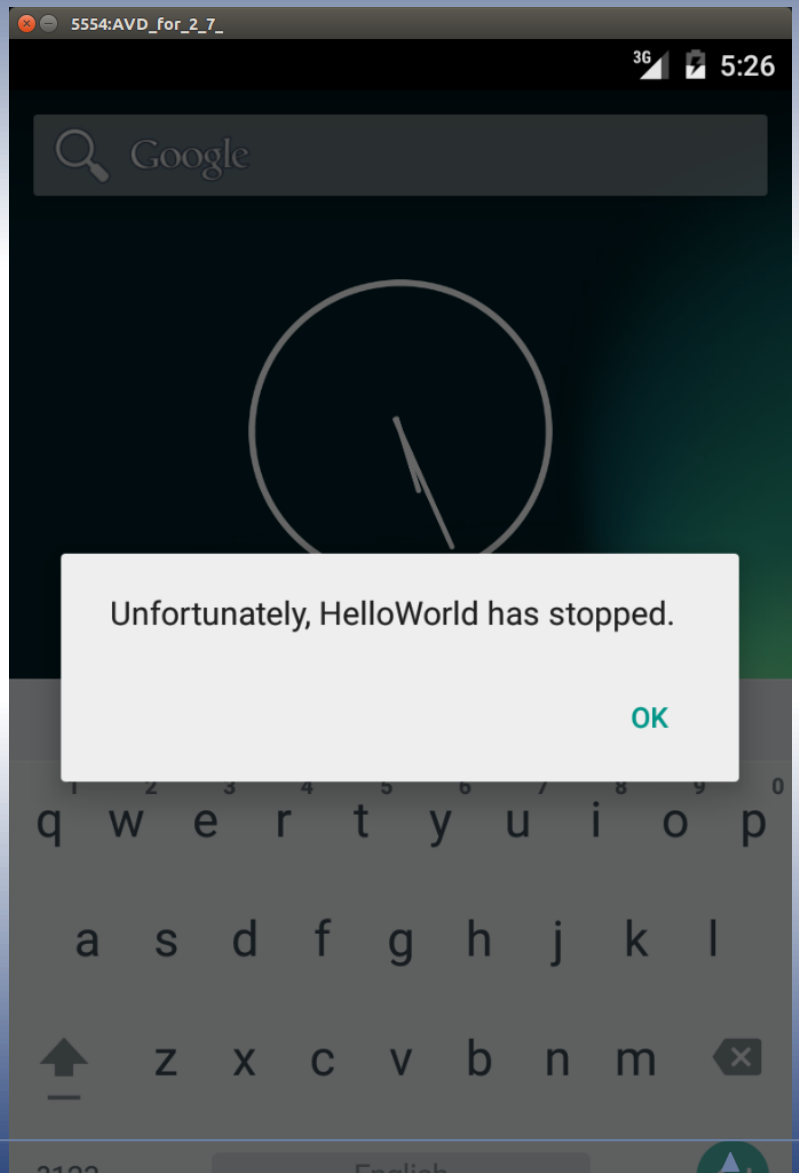
```
public class AfficheMessage extends Activity {

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

        // Get the message from the intent
        Intent intent = getIntent();
        String message = intent.getStringExtra(MainActivity.MESSAGE);

        // Create the text view
        TextView textView = new TextView(this);
        textView.setTextSize(40);
        textView.setText(message);

        // Set the text view as the activity layout
        setContentView(textView);

    }
}
```

Transfert des données à AfficheMessage

Modification de MainActivity.java

```
Button button = (Button) findViewById(R.id.button_send);
button.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        System.out.println("button send");
        Intent myIntent = new Intent(MainActivity.this, AfficheMessage.class);
        myIntent.putExtra(MESSAGE, textView.getText().toString());
        startActivity(myIntent);
    }
});
```

Problème ! ! !


5554:AVD_for_2_7_
3G
5:26
Google
Unfortunately, HelloWorld has stopped.
OK
q w e r t y u i o p
a s d f g h j k l
z x c v b n m

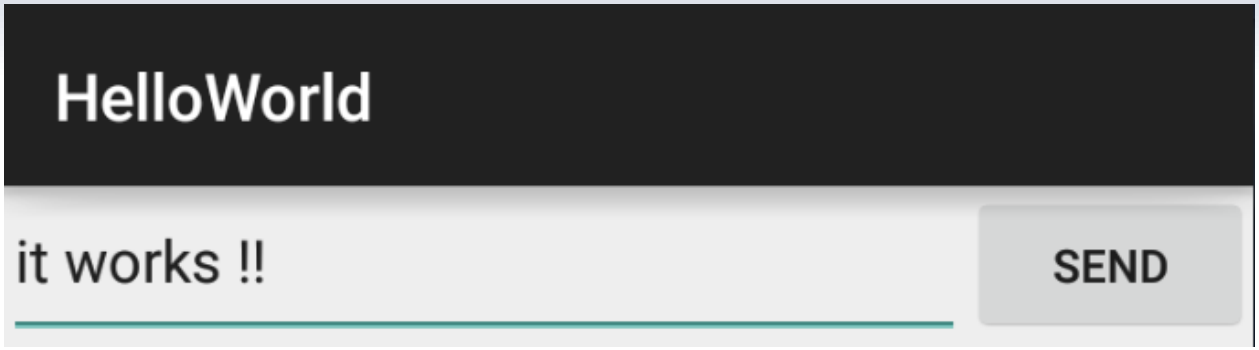
Problème → LogCat en mode debug

Tag	Text
AndroidRuntime	Shutting down VM
AndroidRuntime	FATAL EXCEPTION: main
AndroidRuntime	Process: fr.iutmontp.helloworld, PID: 23949
AndroidRuntime	android.content.ActivityNotFoundException: Unable to find explicit act ⏎ ivity class {fr.iutmontp.helloworld/fr.iutmontp.helloworld.AfficheMess ⏎ age}; have you declared this activity in your AndroidManifest.xml?
AndroidRuntime	at android.app.Instrumentation.checkStartActivityResult(Instrumentati ⏎ on.java:1761)
AndroidRuntime	at android.app.Instrumentation.execStartActivity(Instrumentation.java ⏎ :1485)
AndroidRuntime	at android.app.Activity.startActivityForResult(Activity.java:3736)
AndroidRuntime	at android.app.Activity.startActivityForResult(Activity.java:3697)
AndroidRuntime	at android.support.v4.app.FragmentActivity.startActivityForResult(Fra ⏎ gmentActivity.java:817)
AndroidRuntime	at android.app.Activity.startActivity(Activity.java:4007)
AndroidRuntime	at android.app.Activity.startActivity(Activity.java:3975)

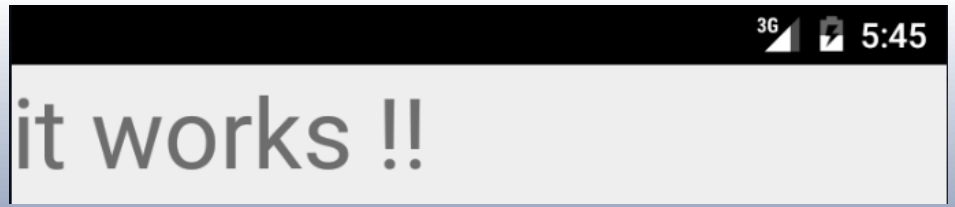
Modification AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="fr.iutmontp.helloworld"
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="21" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".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>
        <activity android:name=".DisplayMessageActivity"></activity>
        <activity android:name=".AfficheMessage"></activity>
        </application>

</manifest>
```

Transfert des données à AfficheMessage



$\Rightarrow$



Plan

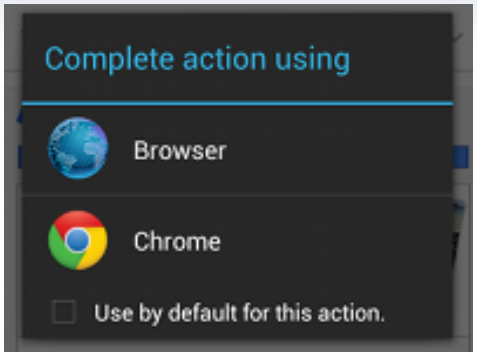
Intent implicite

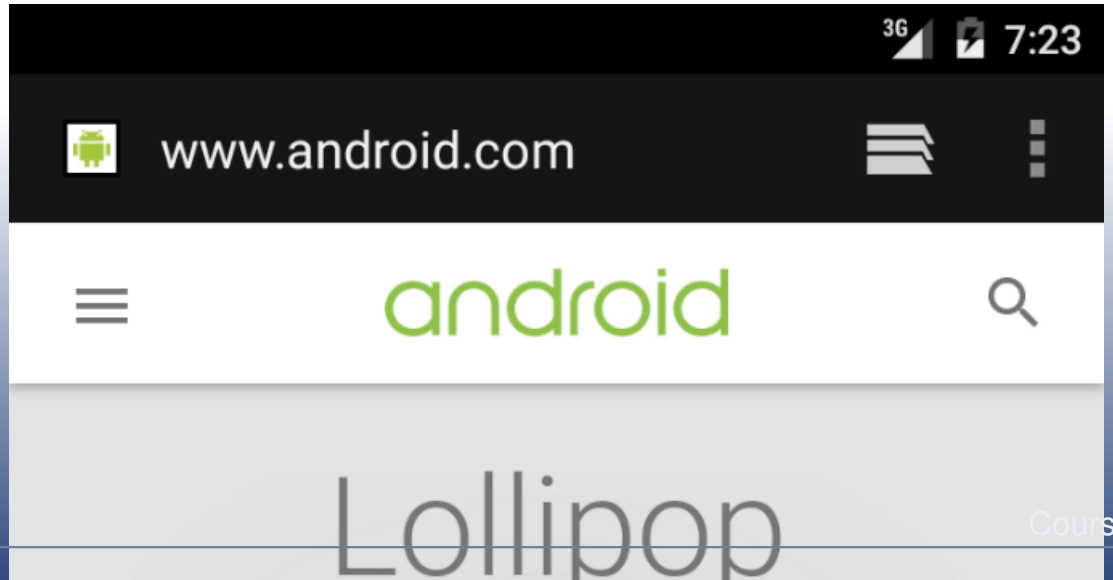
Objectif d'un Intent implicite

- Permet de demander au système de réaliser une action particulière sans viser une Activité spécifique
- Lors de l'utilisation, Android cherche parmi les activités qui se sont enregistrées comme capable de gérer cette demande (manifest)
- Si plusieurs activitées sont trouvées, il est automatiquement demandé à l'utilisateur de choisir ("*ouvrir avec*")
- Exemples :
  - affichage d'une page web
  - composition d'un mail

Exemple : ActionView (générique)

```
Uri webpage = Uri.parse("http://www.android.com");
Intent webIntent = new Intent(Intent.ACTION_VIEW, webpage);
startActivity(webIntent);
```





Précaution d'emploi

Vérification de la disponibilité d'un composant adéquat

- Il est préférable de vérifier que le système est capable de gérer l'action demandée
- si aucun composant ne peut gérer une demande, l'application initiatrice plante !

```
Uri webpage = Uri.parse("http://www.android.com");
Intent webIntent = new Intent(Intent.ACTION_VIEW, webpage);

PackageManager packageManager = getPackageManager();
List<ResolveInfo> activities = packageManager.queryIntentActivities(intent, 0);
boolean isIntentSafe = activities.size() > 0;

if (isIntentSafe) {
    startActivity(webIntent);
}
```

Enregistrer une activité comme capable de gérer un Intent

Nouvelle activité : AfficheURL

```
public class AfficheURL extends Activity {

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

        // Get the intent that started this activity
        Intent intent = getIntent();
        Uri url = intent.getData();

        // Create the text view
        TextView textView = new TextView(this);
        textView.setTextSize(40);
        textView.setText(url.toString());

        // Set the text view as the activity layout
        setContentView(textView);
```

Définition d'un intent-filter dans le manifest

```
<activity android:name=".AfficheURL" >
    <intent-filter>
        <action android:name="android.intent.action.VIEW" />
        <category android:name="android.intent.category.DEFAULT" />
        <data android:scheme="http" />
    </intent-filter>
</activity>
```

intent-filter

- **action** : type d'action gérée par l'activité (e.g. Action.SEND, Action.VIEW)
- **data** : le type de donnée que l'activité peut gérer (e.g. android :mimeType, android :scheme)
- **category** : permet de spécifier plus avant le type de l'activité (par défaut : DEFAULT)

Résumé global

- **Activity**
  - définition : **manifest** Android
  - création / implémentation : extends **android.app.Activity**
  - lancement : **android.app.Activity.startActivity(Intent)**
- **Intent**
  - **explicite** : lancement d'une activité spécifique
  - **implicite** : demande de lancement d'un service
  - communication entre activités : **android.content.Intent.putExtra(K,V)**

Ce cours reprend largement les tutoriaux en ligne proposés par Google :

▸ Android developers