

Mobilitéé: Programmation Android

1

Erick STATNER

Maître de Conférences en Informatique

Université des Antilles

erick.stattner@univ-antilles.fr

www.erickstattner.com

Description de l'enseignement

Objectifs pédagogiques:

- Se familiariser à la Programmation d'applications pour mobile
- Maîtriser les principes autour des applications Android
- Concevoir des applications graphiques sous Android
- Mettre en place la persistance des données

Organisation:

- 30h
- 1 CC + 1 CT

Sommaire

1. Android: Présentation, configuration et principes
- 2. Premières applications Android**
3. Les interfaces
4. Evènements et échanges
5. Persistance des données

Chapitre II.

Premières applications

1. Activité et cycle de vie
2. Création et architecture d'un projet
3. Les ressources
4. Exemple: Hello World
5. Déploiement sur un périphérique physique

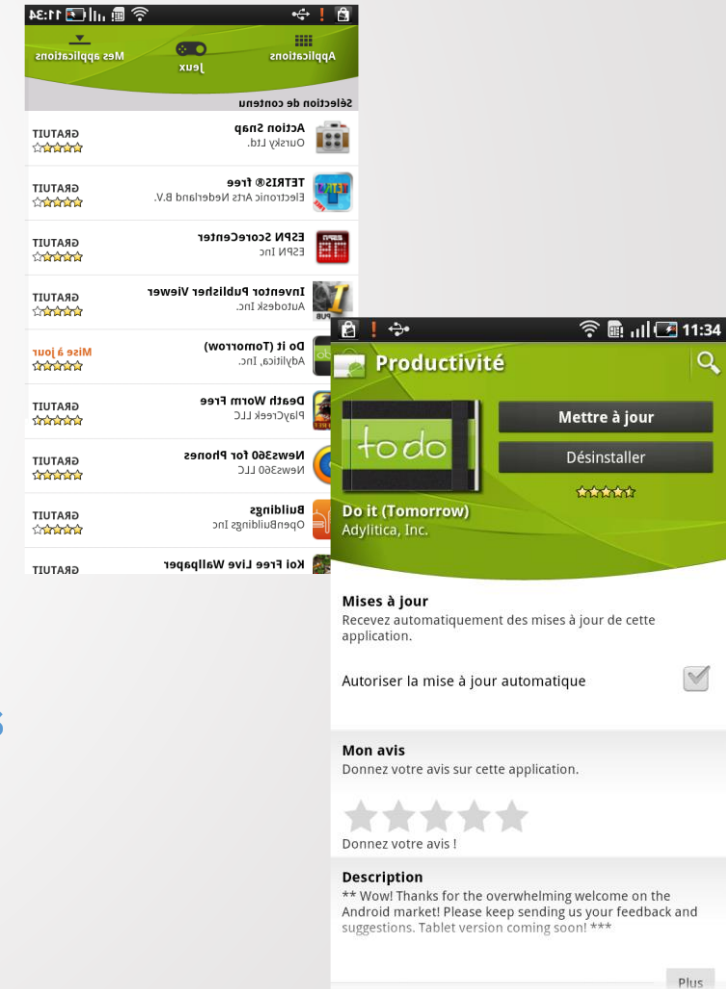
II. Premières applications

Activité et cycle de vie

Activités (**Activity**)

- Composant principal d'une application Android
- Structure l'interface des applications
- Implémentations et interactions des interfaces
- Package **android.app**

Une application Android est un ensemble d'activités qui structurent l'application en différents écrans

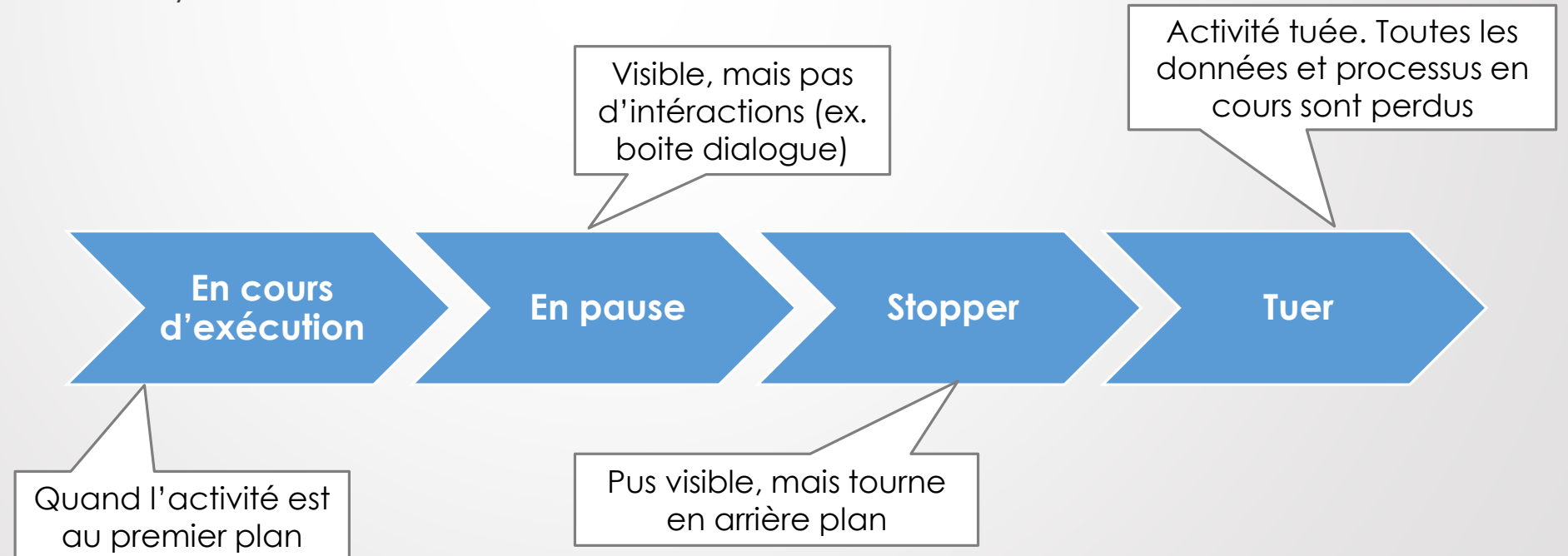


II. Premières applications

Activité et cycle de vie

Etat d'une activité

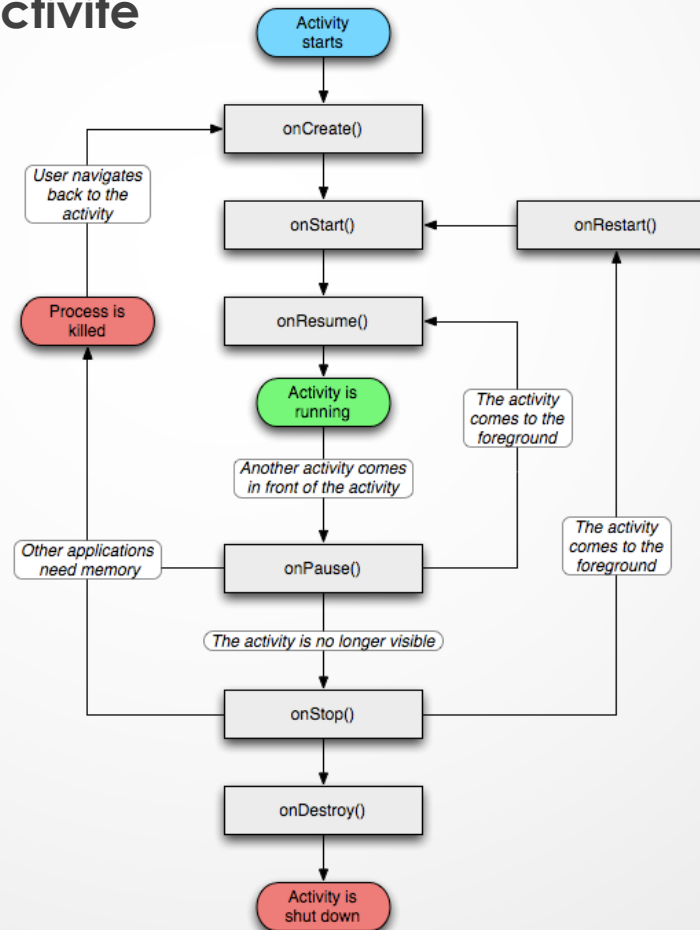
- Chaque application Android s'exécute dans un processus dédié
- Le système peut libérer des ressources si besoin
- Cycle de vie d'une activité



II. Premières applications

Activité et cycle de vie

Cycle de vie d'une activité

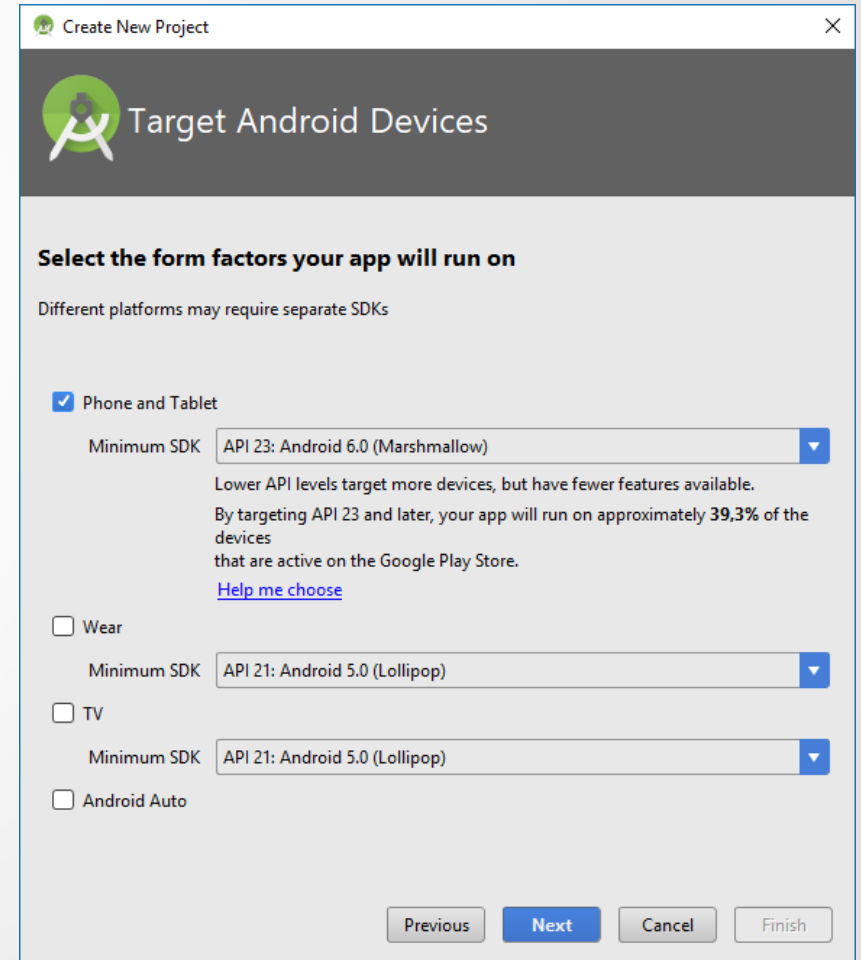


II. Premières applications

Création et architecture d'un projet Android

Création d'un projet

- Nom de l'application
- Id de l'application (package)
- Version d'android utilisée
- Icône de l'application
- Possibilité de créer une première activité

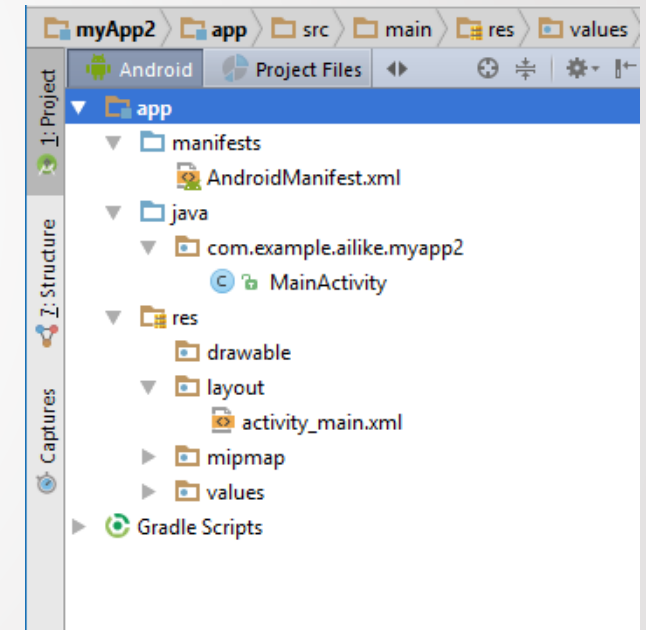


II. Premières applications

Création et architecture d'un projet Android

Architecture d'un projet

- ▶ **java**
sources de l'application
- ▶ **res**
Stocke toutes les ressources utilisées
- ▶ **manifests**
stocke le manifeste (fichier config/description) de l'application
AndroidManifest.xml
- ▶ **Gradle scripts**
stocke les scripts pour l'automatisation de la production



II. Premières applications

Les ressources

Le dossier `res`

- Stocke toutes les ressources utilisées dans l'application
- Paramétrable selon
 - la résolution,
 - l'orientation de l'écran,
 - la langue,
 - la version d'android,
 - etc.
- Automatiquement précompilés
- Ajoutées automatiquement au fichier `R.java`
- Accessible partout dans le code

II. Premières applications

Les ressources

Le dossier res

- **Drawable**

- Les images de l'application

- **Layout**

- Les vues de l'application

- **Values**

- Valeurs utiles dans l'application

- Strings: chaîne de caractères
- Colors: les couleurs utilisés dans l'application
- Style: le thème
- Dimensions
- Etc.

II. Premières applications

Les ressources

Paramétrage des ressources

- ▶ Possibilité de paramétrer chaque ressource du dossier **res** en précisant les ressources à utiliser selon
 - ▶ Langue
 - ▶ Taille de l'écran
 - ▶ Orientation de l'écran
 - ▶ Version d'android
- ▶ Une combinaison de paramètres

II. Premières applications

Les ressources

Paramétrage des ressources

- ▶ Exemples
 - ▶ `drawable-ldpi`: images utilisées pour les écrans à faibles résolutions
 - ▶ `drawable-hdpi`: images utilisées pour les écrans à très hautes résolutions
 - ▶ `drawable`: images utilisées par défaut

- ▶ `values-fr`: valeurs utilisées pour les périphériques en français
- ▶ `values-es`: valeurs utilisées pour les périphériques en espagnol
- ▶ `values`: valeurs utilisées par défaut

II. Premières applications

Les ressources

Pour la gestion des **drawable**

	Low density (120), <i>ldpi</i>	Medium density (160), <i>mdpi</i>	High density (240), <i>hdpi</i>	Extra high density (320), <i>xhdpi</i>
Small screen	QVGA (240x320)		480x640	/drawable
Normal screen	WQVGA400 (240x400) WQVGA432 (240x432)	HVGA (320x480)	WVGA800 (480x800) WVGA854 (480x854) 600x1024	640x960 /drawable-normal-hdpi
Large screen	WVGA800** (480x800) WVGA854** (480x854)	WVGA800* (480x800) WVGA854* (480x854) 600x1024		/drawable-xlarge
Extra Large screen	1024x600	WXGA (1280x800)[†] 1024x768 1280x768	1536x1152 1920x1152 1920x1200	2048x1536 2560x1536 2560x1600

II. Premières applications

Les ressources

Création de deux chaînes de caractère

- Dossier: **res > values > strings.xml**
Utilisé si l'écran est en mode portrait

```
<resources>  
  <string name="message">Hello world: je suis en mode portait</string>  
</resources>
```

- Dossier: **res > values > strings-land.xml**
Utilisé si l'écran est en mode paysage (*landscape*)

```
<resources>  
  <string name="message">Hello world: je suis en mode paysage</string>  
</resources>
```

II. Premières applications

Les ressources

Les ressources

- ▶ Une fois la ressource créée, elle est précompilée et accessible partout
 - ▶ Dans un fichier JAVA
exemple:
`R.strings.message`
OU
`R.drawable.monImage`
 - ▶ Dans un fichier XML
exemple:
`@string/message`
OU
`@drawable/monImage`

II. Premières applications

Exemple: Hello World

Application Hello World

1. Création de la vue (**Layout**): *activity-main.xml*



II. Premières applications

Exemple: Hello World

Application Hello World

1. Création de la vue (**Layout**): *activity-main.xml*
Les ressources sont accessibles pour construire la vue



```
<?xml version="1.0" encoding="utf-8" ?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/message"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>
```

II. Premières applications

Exemple: Hello World

Application Hello World

2. Création de l'activité
 - ▶ Une classe Java
 - ▶ Hériter de la classe **Activity**
 - ▶ Surcharger au minimum la méthode **onCreate()**
 - ▶ Lier l'activité à l'interface

Chaque nouvelle activité doit être déclarée dans le manifeste !
Fait automatiquement par Eclipse ou Android Studio

II. Premières applications

Exemple: Hello World

Application Hello World

2. Création de l'activité

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

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

II. Premières applications

Exemple: Hello World

Application Hello World

2. Création de l'activité

La classe doit hériter de **Activity**

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

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

Appeler la méthode **onCreate** de la classe mère

Surcharger la méthode **onCreate()**

Lier l'activité à la vue créée

II. Premières applications

Exemple: Hello World

Application Hello World

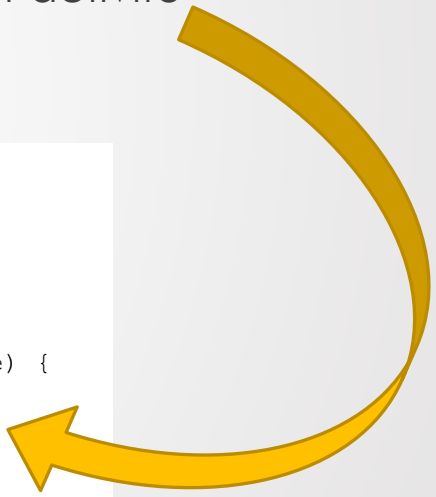
2. Création de l'activité

La vue devient également une ressource, utilisée par l'activité

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

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



II. Premières applications

Exemple: Hello World

Application Hello World

2. Création de l'activité

La classe doit hériter de **Activity**

```
import android.os.Bundle;
import android.support.design.widget.FloatingActionButton;
import android.support.design.widget.Snackbar;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.TextView;

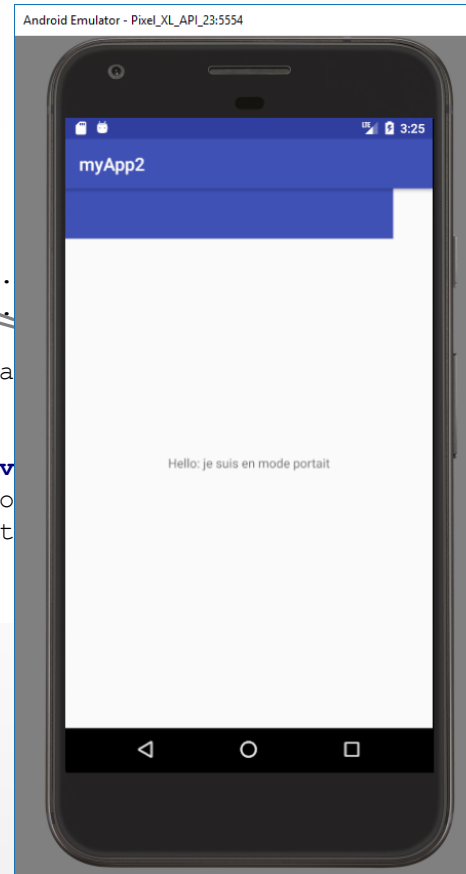
public class MainActivity extends AppCompatActivity {

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

Appeler la méthode **onCreate** de la classe mère

Surcharger la méthode **onCreate()**

Lier l'activité à la vue créée



II. Premières applications

Déploiement sur un périphérique physique

Comment installer une application sur un périphérique physique ?

1. Télécharger et installer les drivers du périphérique
2. Activer le mode développeur du téléphone
Paramètre -> A propos du téléphone -> Cliquer 7x sur Numéro de build
3. Vérifier que "option pour développeur" apparait dans les paramètres
4. Brancher le périphérique via USB
5. Activer débogage USB dans les option développeur
6. Installer sur le périphérique

