Programmation Nomade (Android)

Dominique Archambault
Master HANDI

1 Introduction

1.1 Les systèmes d'exploitation mobiles

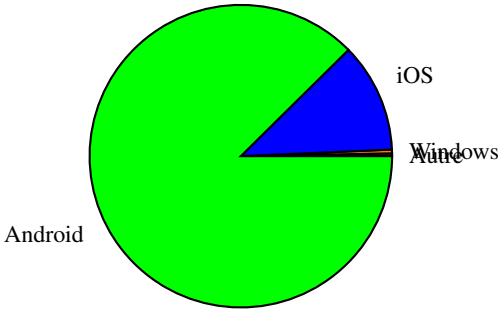
Les systèmes d'exploitation mobiles

Une concurrence féroce

1. Android (Google Inc.)
2. iOS (Apple Inc.)
3. Windows Phone (Microsoft)
4. BlackBerry OS (RIM)
5. Bada (Samsung Electronics)
6. Firefox OK (Mozilla)
7. HP webOS (Palm Inc.)
8. Linux
9. ...
10. Symbian OS (Nokia and Accenture)

Parts de marché (OS)

Période	Android	iOS	Windows Phone	Autres
3ème trim. 2015	84.3%	13.4%	1.8%	0.5%
4ème trim. 2015	79.6%	18.6%	1.2%	0.5%
1er trim. 2016	83.4%	15.4%	0.8%	0.4%
2ème trim. 2016	87.6%	11.7%	0.4%	0.3%



Parts de marché (appareils)

Période	Samsung	Apple	Huawei	OPPO	vivo	Autres
T3 2015	23.3%	13.4%	7.6%	3.2%	2.9%	49.6%
T4 2015	20.4%	18.6%	8.2%	3.6%	3.0%	46.2%
T1 2016	23.8%	15.4%	8.4%	5.9%	4.4%	42.1%
T2 2016	22.8%	11.7%	9.3%	1.0%	5.9%	40.2%

Les *"Boutiques d'applications"*

Principaux OS mobiles (10/2016)

IOS	App Store	1 000 000+
Android	Google Play	800 000+
Windows Phone	Marketplace	> 9 000

Développement d'applications

— **IOS :** Objective C, Support d'OpenGL Plate-forme : XCode sur Mac Autre : Apple Developper Program

— **Windows phone :** C# ou VB.net Plate-forme : Silverlight ou XNA

— **Android :** Java, Support d'OpenGL Plateforme : ligne de commande ou Eclipse

Développement d'Applications mobiles

Difficultés

— Taille réduite des écrans
— Taille des claviers (réels ou virtuels)
— Dispositifs de pointage peu pratiques ou imprécis

Écueils à éviter

— L'application utilise tellement de ressources qu'on ne peut pas recevoir d'appels
— L'application ne passe pas bien en arrière plan, ou s'arrête (et les tâches en cours sont perdues).
— L'application provoque un plantage du téléphone.

Points forts d'Android

— Langage de programmation très classique, disposant de nombreuses bibliothèques (Java)
— Outils de développement intégrés à Eclipse (plugin)
— Framework stable, en mode protégé (les applications ne peuvent pas interférer entre elles ou avec le système d'exploitation).

Développement d'Applications mobiles

Fonctionnalités disponibles

— Stockage interne et sur carte SD (fichier ou BD)
— Multimédia (musique, vidéo, photo, mémos vocaux)
— Réseau (stockage dans le *cloud*, applications Web
— Géolocalisation (via puce GPS ou via réseau)
— Différents capteurs : Accéléromètre, Gyroscope, Boussole, Capteur de proximité, Baromètres...
— Écrans multitouch
— Lecteur de RFID
— Services téléphoniques (appels voix/vidéo, SMS)

Le système d'exploitation Android

Android

C'est un système d'exploitation pour terminaux mobiles (smartphones, tablettes, PDA, *etc.*), basé sur le noyau Linux.

- Licence open source (Apache v2)
- Développé par Google Inc.

Conçu pour intégrer les application Google

- Gmail
- Google Maps
- Google Agenda
- Google Talk
- YouTube
- Google Latitude

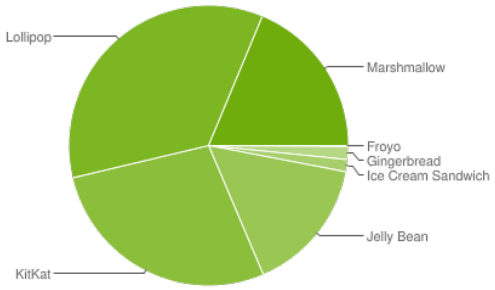
Le système d'exploitation Android

Versions

Version		Sortie	API level
1.0	*Apple Pie* (α) Tarte aux pommes	fin 2007	1
1.1	*Banana Bread* (β) Cake à la banane	fin 2008	2
1.5	*Cupcake* Petit gâteau	30/04/2009	3
1.6	*Donut* Beignet	15/09/2009	4
2.0 - 2.1	*Eclair* Éclair	26/10/2009	5-6-7
2.2.x	*Froyo (Frozen Yogourt)* Yaourt glacé	20/05/2010	8
2.3.x	*Gingerbread* Pain d'épices	06/12/2010	9-10
3.x	*Honeycomb* Rayon de miel	22/02/2011	11-13
4.0.x	*Ice Cream Sandwich* Sandwich à la glace	19/10/2011	14-15
4.1.x - 4.3.x	*Jelly Bean* Dragibus	09/07/2012	16-18
4.4.x	*KitKat* KitKat	4/11/2013	19
5.0.x-5.1.x	*Lollipop*	3/11/2014	21-22
6.0-6.0.1	*Marshmallow*	5/10/2015	23
7.0	*Nougat*	22/08/2016	24

Le système d'exploitation Android

Versions (2016)



voir `http://developer.android.com`

[About > Dashboards]

Le système d'exploitation Android

Problèmes

- Nombreuses versions
- ROMs modifiés par les constructeurs
- ROMs modifiés par les opérateurs
- ROMs alternatives

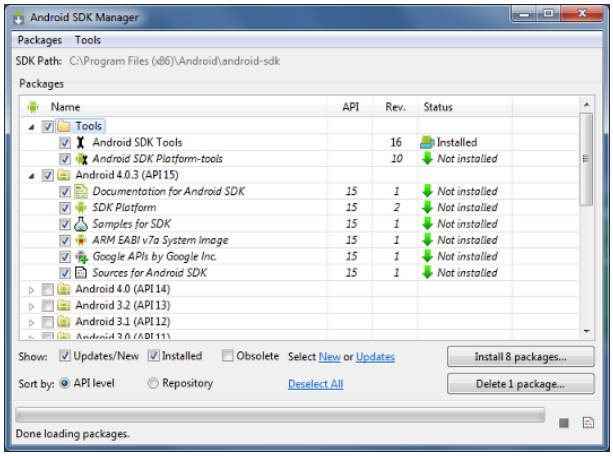
1.2 Développer des applications Android

Développement d'application Android

Outils de développement

- **le SDK** à télécharger sur
  `http://developer.android.com`
  [Develop > Tools > Download]

- **les plateformes et paquets** Dans le répertoire du SDK, lancer le **Android SDK Manager**. Les paquets recommendés sont sélectionnés automatiquement, choisir ceux qu'on veut installer et lancer l'installation.



- **le plugin Eclipse** Dans Eclipse,
  menu [Help > Install New Software]
  Ajouter le *Repository* : **ADT Plugin** avec l'URL suivante :
  `https://dl-ssl.google.com/android/eclipse/`
- **Android Studio**

Développement d'application Android

Langages de développement

- **XML** pour la définition des écrans Un outil graphique est disponible
- **JAVA** pour l'implémentation

Développement d'application Android

Pré-requis en java

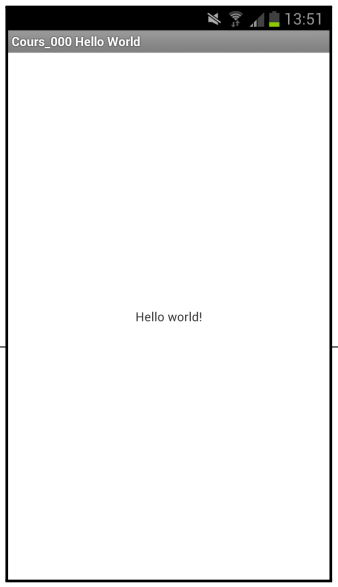
- **Héritage**
- **Interfaces**
- notion d'**Adapteurs**

Développement d'application Android

Contenu du programme Android

- **Activités (*Activities*)**
- **Services (*Services*)**
- **Fournisseurs de contenu (*Content providers*)**
- **Intentions (*Intents*)**

Annexe 1 : Application “Hello World !”



Code java : `HelloWorldActivity.java`

```
package net.chezdom.android.cours_000;

import android.os.Bundle;

public class HelloWorldActivity extends Activity {

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

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.activity_hello_world, menu);
        return true;
    }

}
```

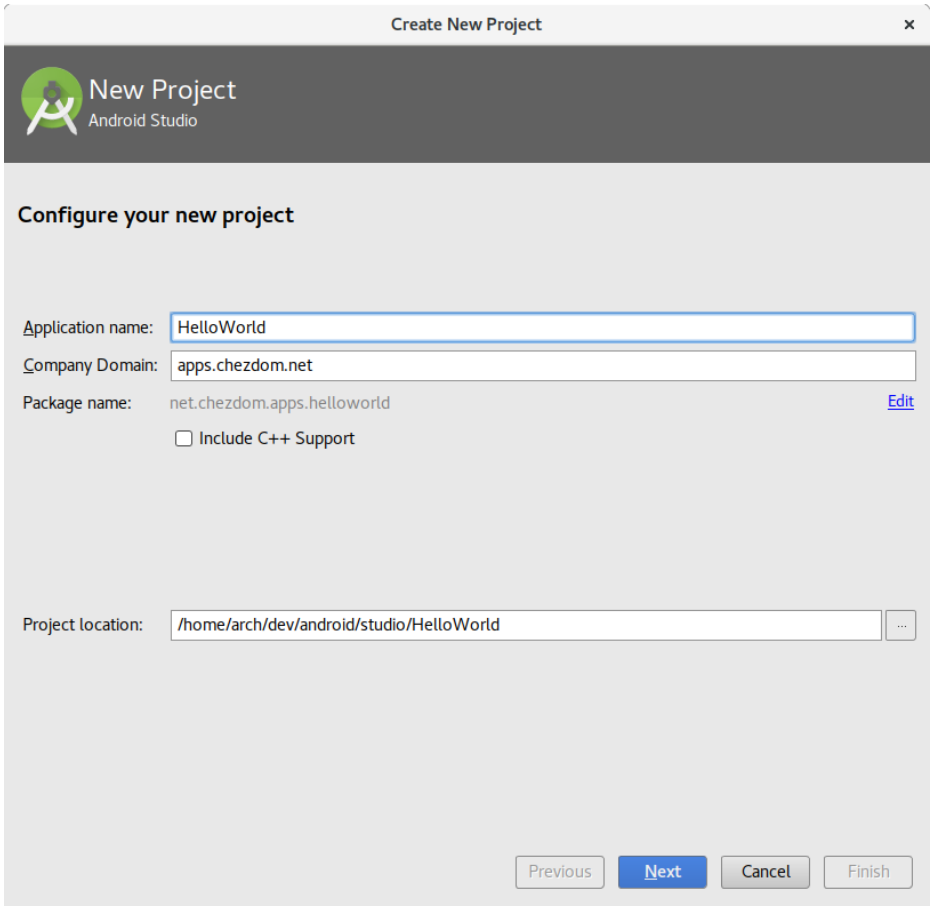
Code XML (écran) : `HelloWorldActivity.xml`

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

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:padding="@dimen/padding_medium"
        android:text="@string/hello_world"
        tools:context=".HelloWorlsActivity" />

</RelativeLayout>
```

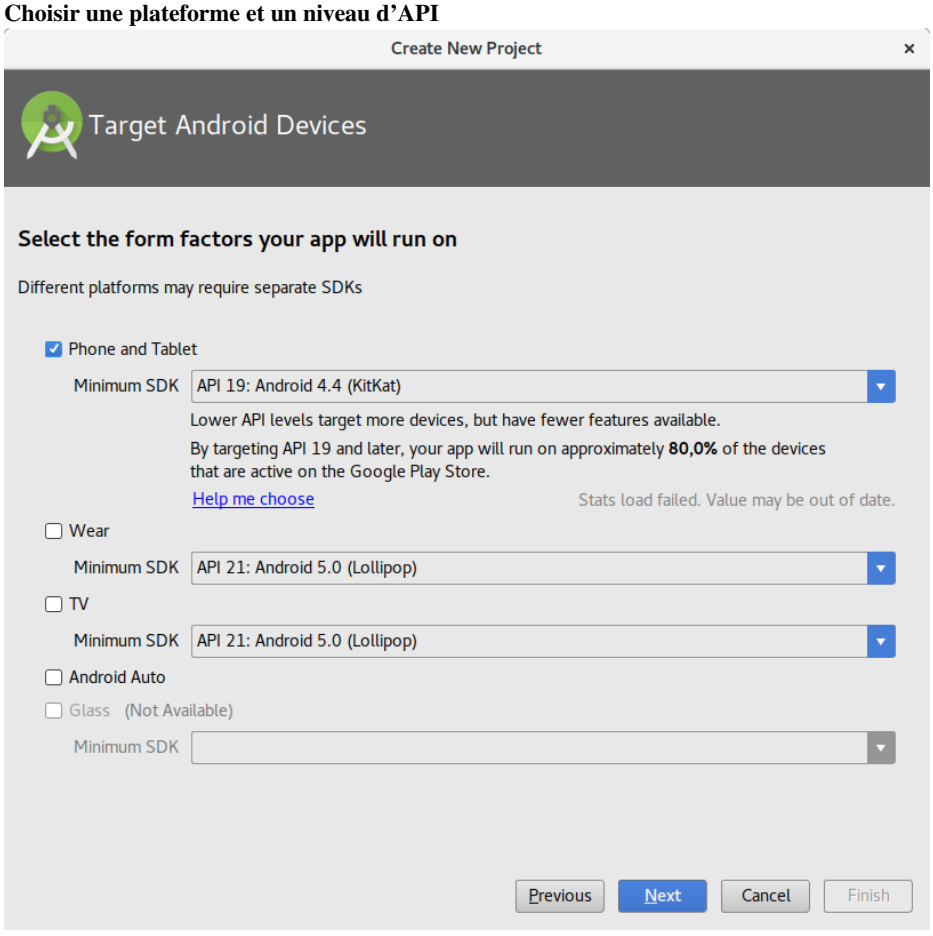
Code XML (chaînes) : `strings.xml`

```
<resources>

    <string name="app_name">Cours_000 Hello World</string>
    <string name="hello_world">Hello world!</string>
    <string name="menu_settings">Settings</string>
    <string name="title_activity_hello_world">Cours_000 Hello World</string>

</resources>
```

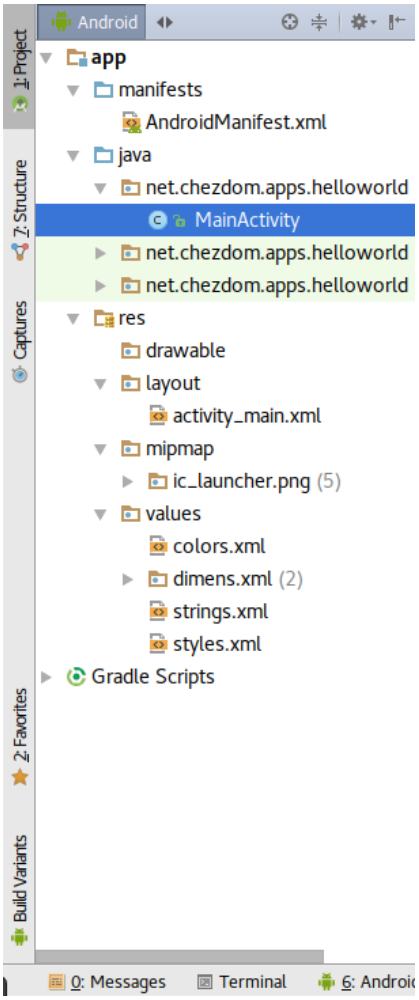
Assistant *"New Android Project"*



Choisir une plateforme et un niveau d'API



Projet Android



2 Interfaces utilisateurs

Créer une interface utilisateur

Étapes

- Créer un *"layout"* en utilisant l'assistant graphique
- Créer les objets correspondants aux widgets qu'on souhaite manipuler dans le programme JAVA
- Associer ces objets aux éléments du *"layout"* par l'intermédiaire de leurs IDs
- Faire implémenter les interfaces de manipulation de ces objets à la classe *Activity* de l'application (ou créer des classes "controleurs" dédiées).
- Associer les objets correspondants aux widgets à la classe implémentant leurs interfaces de manipulation (this, si on a utilisé la classe Activity.
- Implémenter les méthodes de manipulation.

Quelques éléments d'interface

***Views* (widgets)**

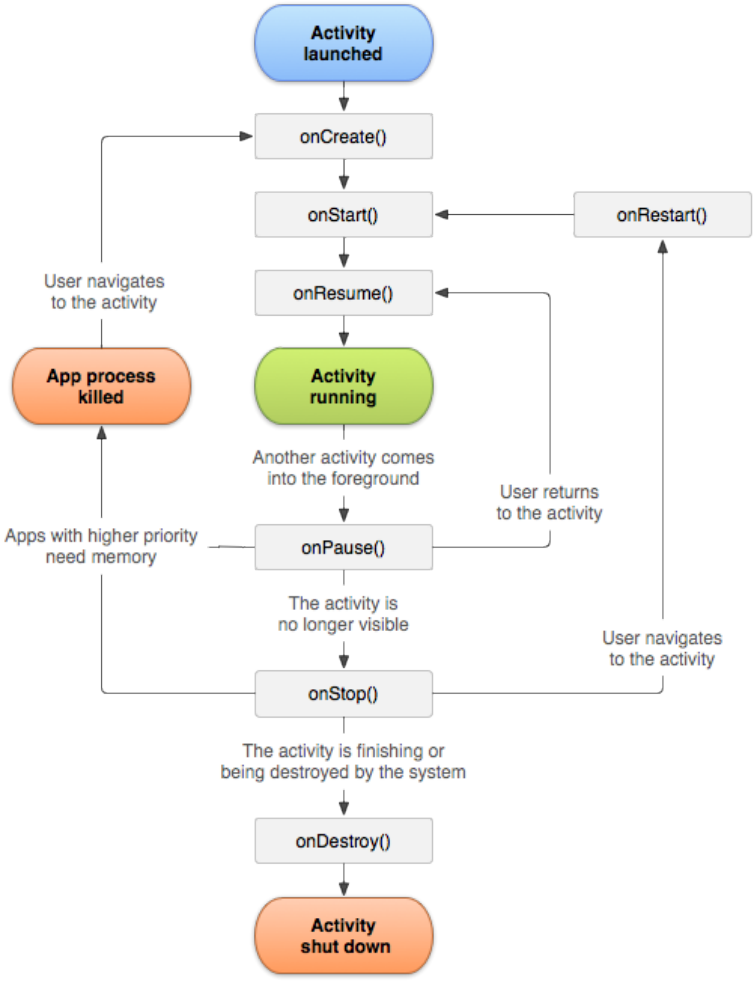
- Button
- CheckBox
- EditText
- RadioButton
- Toast

- DatePicker
- ImageButton
- SlidingDrawer

***Layouts* (conteneurs)**

- LinearLayout
- RelativeLayout
- GridLayout
- FrameLayout

Cycle de vie d'une activité



Principaux Listeners utiles

View.OnClickListener	onClick(View v)
View.OnLongClickListener	onLongClick(View v)
View.OnFocusChangeListener	onFocusChange(View v, boolean hasFocus)
View.OnKeyListener	onKey(View v, int keyCode, KeyEvent event)
View.OnTouchListener	onTouch(View v, MotionEvent event)

```
public abstract void onClick (View v)
```

Paramètres

v L'élément `View` qui a reçu le click.

```
public abstract boolean onLongClick (View v)
```

Paramètres

v L'élément `View` qui a reçu le click.

Returns true if the callback consumed the long click, false otherwise.

```
public abstract void onFocusChange (View v, boolean hasFocus)
```

Paramètres

v L'élément `View` qui a reçu le click.

hasFocus The new focus state of v.

```
public abstract boolean onKey (View v, int keyCode, KeyEvent event)
```

Paramètres

v L'élément `View` qui a reçu le click.

keyCode The code for the physical key that was pressed

event The KeyEvent object containing full information about the event.
Returns True if the listener has consumed the event, false otherwise.

```
public abstract boolean onTouch (View v, MotionEvent event)
```

Paramètres

v L'élément `View` qui a reçu le click.

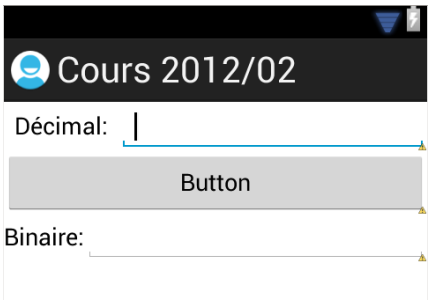
event The MotionEvent object containing full information about the event.

Returns True if the listener has consumed the event, false otherwise.

TP 1

Nous allons créer une première application, dont le but est de convertir des nombres entiers en binaire.

1. Créer un layout avec l'assistant graphique



Indication : Le layout ci-dessus est formé d'un **`LinearLayout`** vertical, contenant :

- un **`LinearLayout`** horizontal, contenant lui-même :
  - une étiquette (**`TextView`**)
  - et un champ de saisie **`EditText`**.
- un **`Button`**,
- et un second **`LinearLayout`** horizontal. contenant aussi une étiquette et un champ de saisie.

On pourra aussi (dans l'éditeur graphique), utiliser un champ de saisie qui n'accepte que les nombres.

2. Créer les objets correspondants et les associer aux éléments du layout

```
import android.widget.Button;

public class MainActivity extends Activity {

  private Button bGO;

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

    bGO = (Button) this.findViewById(R.id.button1);
  }
```

NB : Attention à bien importer la classe **`Button`** du package android.widget (voir la première ligne du code ci-dessus).
On trouvera l'ID du bouton dans le fichier XML "layout".

```
    <Button
        android:id="@+id/button1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Button" />
```

3. La classe Activity implémente l'interface de manipulation

```
import android.view.View.OnClickListener;

public class MainActivity extends Activity
                    implements OnClickListener {
  //...

  @Override
  public void onClick(View v) {
    // ...
  }
```

NB : Attention à bien importer l'interface **`OnClickListener`** du package android.widget.View.

4. Associer les objets widgets à leur interface de manipulation

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

    bGO = (Button) this.findViewById(R.id.button1);
    bGO.setOnClickListener(this);
  }
```

5. Implémenter les méthodes de manipulation

```
import android.widget.Toast;

//...

@Override
public void onClick(View v) {
  if (v.getId()==R.id.button1)
      Toast.makeText(getApplicationContext(),
                     "Conversion !",
                     Toast.LENGTH_SHORT).show();

}
```

6. Récupérer une valeur dans un champ de texte

On associe l'objet EditText avec son champ de saisie, en utilisant l'ID.

```
import android.widget.EditText;

public class MainActivity extends Activity
                    implements OnClickListener {

  //...
  private EditText tfDecimal, tfBinaire;

  public void onCreate(Bundle savedInstanceState) {
    //..
    tfDecimal=(EditText) this.findViewById(
                                    R.id.editText1);

  }
```

```
  String chDecimal=tfDecimal.getText().toString();
  int d=Integer.parseInt(chDecimal);
```

7. Écrire dans un champ de texte

Ne pas oublier d'associer l'objet EditText avec son champ de saisie, en utilisant l'ID comme précedemment.

```
  tfBinaire.setText(""+convertInBinary(d));
```

Rappel : Conversion en binaire

(voir TD intensifs)

```
1  bin=""
2  tantque n>0
3  faire
4    si n%2=0
5    alors
6      bin = "0" + bin
7    sinon
8      bin = "1" + bin
9    finsi
10   n=n/2
11 fait
```