



ANDROID

An Open Handset Alliance Project

Programmation Événementielle

G.Seront

Programmation Événementielle

- L'utilisateur génère des événements:
 - Touch (court ou long)
 - Bouton hardware (back, menu, home)
 - Touche clavier
 - Fin application, ...
- Pour les « desktops »
 - Mouvement de la souris
 - Click gauche ou droit
 - Entrée sortie de la souris d'un widget



Programmation Événementielle

- Le programme s'abonne à certains événements
 - Indique quelle méthode appeler
 - Le framework Android se charge de la gestion



Enregistrement Event (XML + Java)

- En XML dans le layout

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/exit"
    android:id="@+id/button_exit"
    android:onClick="onExit"/>
```

Nom de la méthode
gestionnaire

- En Java dans l'Activity

```
public void onExit(View view) {
    this.finish();
}
```

Objet déclencheur
(ici le Button)



Enregistrement Event (Full Java)

```
// Create an anonymous implementation of OnClickListener
private OnClickListener mCorkyListener = new OnClickListener() {
    public void onClick(View v) {
        // do something when the button is clicked
    }
};

protected void onCreate(Bundle savedInstanceState) {
    ...
    // Capture our button from layout
    Button button = (Button)findViewById(R.id.corky);
    // Register the onClick listener with the implementation above
    button.setOnClickListener(mCorkyListener);
    ...
}
```

Id du widget



Enregistrement Event

- `widget.setOnClickListener(listener)`
 - abonnement à l'évènement « click »
 - *widget* est l'objet sur lequel on clique
 - *listener* est un objet implémentant l'interface *OnClickListener*
- *OnClickListener*
 - Interface qui impose la méthode *onClick*
 - Une interface par type d'évènement (Click, Drag, Hover, ...)



Classe Anonyme

- C'est quoi ce truc??

```
// Create an anonymous implementation of OnClickListener
private OnClickListener mCorkyListener = new OnClickListener() {
    public void onClick(View v) {
        // do something when the button is clicked
    }
};
```

- Créé un objet d'une classe implémentant l'interface OnClickListener
- La classe n'a juste pas de nom



Classe Anonyme

- C'est un raccourci de

```
public class MyClickListener implements View.OnClickListener {
    @Override
    public void onClick(View view) {
        // Do something
    }
}
```

- et

classes « compatibles »

```
// Dans la classe Activity
private View.OnClickListener mCorkyListner = new MyClickListener();
```

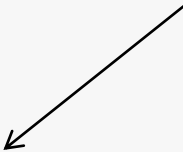



C'est clair?

```
// Create an anonymous implementation of OnClickListener
private OnClickListener mCorkyListener = new OnClickListener() {
    public void onClick(View v) {
        // do something when the button is clicked
    }
};

protected void onCreate(Bundle savedInstanceState) {
    ...
    // Capture our button from layout
    Button button = (Button)findViewById(R.id.corky);
    // Register the onClick listener with the implementation above
    button.setOnClickListener(mCorkyListener);
    ...
}
```

Id du widget




En plus compact

```
button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

    }
});
```

Attention à la lisibilité, ...



Lambda expression

- Pas supporté par défaut
- Il faut utilisé le compilateur « Jack »
- Voir ref en bas de diapo



Types d'événements

- Une interface par type d'événement

```
View.On {
  OnClickListener (android.view.View)
  OnApplyWindowInsetsListener (android.view.View)
  OnAttachStateChangeListener (android.view.View)
  OnContextClickListener (android.view.View)
  OnCreateContextMenuListener (android.view.View)
  OnDragListener (android.view.View)
  OnFocusChangeListener (android.view.View)
  OnGenericMotionListener (android.view.View)
  OnHoverListener (android.view.View)
  OnKeyListener (android.view.View)
  OnLayoutChangeListener (android.view.View)
```

Did you know that Quick Documentation View (Ctrl+Q) works in completion lookups as well? >> π

<http://developer.android.com/guide/topics/ui/ui-events.html>



Consommation des events

- Par défaut les events remontent (bubbling) vers la vue parent

```
button.setOnLongClickListener(  
    new Button.OnLongClickListener() {  
        public boolean onLongClick(View v) {  
            TextView myTextView =  
                (TextView) findViewById(R.id.myTextView);  
            myTextView.setText("Long button click");  
            return true;  
        }  
    }  
);
```

Pas de bubbling
(événement consommé)



Les menus

- Les menus sont défini en XML

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/new_game"
        android:icon="@drawable/ic_new_game"
        android:title="@string/new_game"
        android:showAsAction="ifRoom"/>
    <item android:id="@+id/help"
        android:icon="@drawable/ic_help"
        android:title="@string/help" />
</menu>
```



Menus et events

- Lorsqu'on « demande » le menu

déjà par défaut dans *Activity*;
pas besoin de s'enregistrer

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.game_menu, menu);
    return true;
}
```

déséréalise le menu



Menus et events

- Lorsqu'on sélectionne une option

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    // Handle item selection
    switch (item.getItemId()) {
        case R.id.new_game:
            newGame();
            return true;
        case R.id.help:
            showHelp();
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}
```



Autre events déjà vu?



Autre events déjà vu?

- Cycle de vie!
 - onCreate
 - onResume
 - ...

