

Android

20 Décembre 2024

Table des matières


| | | |
|----------|--------------------------------------|-----------|
| 1 | Programmation | 2 |
| 1.1 | Interface graphique | 2 |
| 1.2 | Implémentation | 4 |
| 1.3 | Évolution de l'application | 8 |
| 2 | Utiliser la documentation | 13 |

1 Programmation

1.1 Interface graphique

Android est un système d'exploitation mobile pour smartphones, tablettes tactiles, PDA, smartwatches et terminaux mobiles. C'est un système open source utilisant le noyau Linux. Il a été lancé par une startup du même nom rachetée par Google en 2005.

On se propose de calculer l'IMC d'une personne. C'est un nombre qui se calcule à partir de la taille et de la masse corporelle d'un individu, afin qu'il puisse déterminer s'il est trop svelte ou trop corpulent. L'interface graphique doit ressembler à la figure suivante :



Instructions :

- On utilisera uniquement le XML.
- Pour mettre plusieurs composants dans un layout, on se contentera de mettre les composants entre les balises de ce layout.
- On n'utilisera qu'un seul layout.
- Les deux EditText permettront de n'insérer que des nombres. Pour cela, on utilise l'attribut android :inputType auquel on donne la valeur numbers.
- Les TextView qui affichent « Poids : » et « Taille : » sont centrés, en rouge et en gras.
- Pour mettre un TextView en gras on utilisera l'attribut android :textStyle en lui attribuant comme valeur bold.

- Pour mettre un TextView en rouge on utilisera l'attribut android:textColor en lui attribuant comme valeur #FF0000. Vous pourrez trouver d'autres valeurs pour indiquer une couleur à cet endroit.
- Afin de centrer du texte dans un TextView, on utilise l'attribut android:gravity="center".

Voici le layout :

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Poids : "
        android:textStyle="bold"
        android:textColor="#FF0000"
        android:gravity="center"
    />
    <EditText
        android:id="@+id/poids"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:hint="Poids"
        android:inputType="numberDecimal"
    />
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Taille : "
        android:textStyle="bold"
        android:textColor="#FF0000"
        android:gravity="center"
    />
    <EditText
        android:id="@+id/taille"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:hint="Taille"
        android:inputType="numberDecimal"
    />
    <RadioGroup
        android:id="@+id/group"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:checkedButton="@+id/radio2"
        android:orientation="horizontal"
    >
        <RadioButton
            android:id="@+id/radio1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Metre"
```

```

/>
<RadioButton
    android:id="@+id/radio2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Centimetre"
/>
</RadioGroup>
<CheckBox
    android:id="@+id/mega"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Mega fonction !"
/>
<Button
    android:id="@+id/calcul"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Calculer l'IMC"
/>
<Button
    android:id="@+id/raz"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="RAZ"
/>
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Resultat:"
/>
<TextView
    android:id="@+id/result"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:text="c"
/>
</LinearLayout>

```

1.2 Implémentation

Il faut maintenant de relier tous les boutons de l'application pour pouvoir effectuer tous les calculs, en respectant les quelques règles suivantes :

- La CheckBox de megafonction permet de changer le résultat du calcul en un message élogieux pour l'utilisateur.
- La formule pour calculer l'IMC est

$$IMC = \frac{poids(en\ kilo)}{taille(en\ metres) * taille(en\ metres)}$$

- Le bouton RAZ remet à zéro tous les champs (sans oublier le texte pour le résultat).

- Les éléments dans le RadioGroup permettent à l'utilisateur de préciser en quelle unité il a indiqué sa taille. Pour obtenir la taille en mètres depuis la taille en centimètres il suffit de diviser par 100 :

$$\frac{171cm}{100} = 1,71m$$

- Dès qu'on change les valeurs dans les champs Poids et Taille, on remet le texte du résultat par défaut puisque la valeur calculée n'est plus valable pour les nouvelles valeurs.
- On enverra un message d'erreur si l'utilisateur essaie de faire le calcul avec une taille égale à zéro grâce à un **Toast**.

"Un Toast est un widget un peu particulier qui permet d'afficher un message à n'importe quel moment sans avoir à créer de vue. Il est destiné à informer l'utilisateur sans le déranger outre mesure ; ainsi l'utilisateur peut continuer à utiliser l'application comme si le Toast n'était pas présent".

Consignes :

Voici la syntaxe pour construire un Toast :

```
static Toast makeText(Context context, CharSequence texte, int duration).
```

duration pour un message court :

```
Toast.LENGTH_SHORT
```

pour un message qui durera plus longtemps.

```
Toast.LENGTH_LONG
```

Enfin, il est possible d'afficher le Toast avec la méthode void show().

- Pour savoir si une CheckBox est sélectionnée, on utilisera la méthode boolean isChecked() qui renvoie true le cas échéant.
- Pour récupérer l'identifiant du RadioButton qui est sélectionné dans un RadioGroup il faut utiliser la méthode int getCheckedRadioButtonId().
- On peut récupérer le texte d'un EditText à l'aide de la fonction Editable getText() et ensuite vider le contenu de cet objet Editable à l'aide de la fonction void clear().
- On ne prend pas en compte les cas extrêmes (taille ou poids < 0 ou null par exemple).
- Pour détecter le moment où l'utilisateur écrit dans un EditText, on peut utiliser l'évènement onKeyDown. Problème, cette technique ne fonctionne que sur les claviers virtuels, alors si l'utilisateur a un clavier physique, ce qu'il écrit n'enclenchera pas la méthode de callback... Pour faire ce genre de surveillance, on préférera utiliser un TextWatcher.

```
import android.app.Activity;
import android.os.Bundle;
import android.view.KeyEvent;
import android.view.MotionEvent;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.View.OnKeyListener;
import android.widget.Button;
import android.widget.CheckBox;
```

```
import android.widget.EditText;
import android.widget.RadioGroup;
import android.widget.TextView;
import android.widget.Toast;

public class IMCAActivity extends Activity {

    private final String default = "Calculer IMC ";

    private final String megaString = " Great job!";

    Button envoyer = null;
    Button raz      = null;
    EditText poids = null;
    EditText taille = null;
    RadioGroup group = null;
    TextView result = null;
    CheckBox mega    = null;

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

        // Getting items
        envoyer = (Button)findViewById(R.id.calcul);
        raz     = (Button)findViewById(R.id.raz);
        taille  = (EditText)findViewById(R.id.taille);
        poids   = (EditText)findViewById(R.id.poids);
        mega    = (CheckBox)findViewById(R.id.mega);
        group   = (RadioGroup)findViewById(R.id.group);
        result  = (TextView)findViewById(R.id.result);

        // Attribute listeners and assign actions
        envoyer.setOnClickListener(envoyerListener);
        raz.setOnClickListener(razListener);
        taille.addTextChangedListener(textWatcher);
        poids.addTextChangedListener(textWatcher);
        mega.setOnClickListener(megaListener);
    }

    private TextWatcher textWatcher = new TextWatcher() {

        @Override
        public void onTextChanged(CharSequence s, int start, int before, int count) {
            result.setText(default);
        }

        @Override
        public void beforeTextChanged(CharSequence s, int start, int count,
            int after) {

        }
    }
}
```

```

@Override
public void afterTextChanged(Editable s) {

}
};

private OnClickListener envoyerListener = new OnClickListener() {
@Override
public void onClick(View v) {
    if(!mega.isChecked()) {

        String t = taille.getText().toString();

        String p = poids.getText().toString();

        float tValue = Float.valueOf(t);

        if(tValue == 0)
            Toast.makeText(IMCActivity.this, "So skinny !!", Toast.LENGTH_SHORT).show();
        else {
            float pValue = Float.valueOf(p);

            if(group.getCheckedRadioButtonId() == R.id.radio2)
                tValue = tValue / 100;

            tValue = (float)Math.pow(tValue, 2);
            float imc = pValue / tValue;
            result.setText("Votre IMC est " + String.valueOf(imc));
        }
    } else
        result.setText(megaString);
}
};

private OnClickListener razListener = new OnClickListener() {
@Override
public void onClick(View v) {
    poids.getText().clear();
    taille.getText().clear();
    result.setText(default);
}
};

private OnClickListener checkedListener = new OnClickListener() {
@Override
public void onClick(View v) {

    if(!((CheckBox)v).isChecked() && result.getText().equals(megaString))
        result.setText(default);
}
};

```

```
}  
};  
}
```

L'évènement `onKey` sera lancé avant que l'écriture soit prise en compte par le système. Ainsi, si vous renvoyez `true`, Android considérera que l'évènement a été géré, et que vous avez vous-même écrit la lettre qui a été pressée. Si vous renvoyez `false`, alors le système comprendra que vous n'avez pas écrit la lettre et il le fera de lui-même. Sans toucher à l'interface graphique, on a pu effectuer toutes les modifications nécessaires au bon fonctionnement de notre application. C'est l'intérêt de définir l'interface dans un fichier XML et le côté interactif en Java : vous pouvez modifier l'un sans toucher l'autre !

1.3 Évolution de l'application

On veut modifier le layout pour obtenir quelque chose ressemblant à la figure suivante :



Les `EditText` prennent le plus de place possible, mais comme ils ont un poids plus fort que les `TextView`, ils n'ont pas la priorité.

```
<?xml version="1.0" encoding="utf-8"?>
```



```

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">
    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
    >
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Poids : "
            android:textStyle="bold"
            android:textColor="#FF0000"
            android:gravity="center"
        />
        <EditText
            android:id="@+id/poids"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:hint="Poids"
            android:inputType="numberDecimal"
            android:layout_weight="1"
        />
    </LinearLayout>
    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
    >
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Taille : "
            android:textStyle="bold"
            android:textColor="#FF0000"
            android:gravity="center"
        />
        <EditText
            android:id="@+id/taille"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:hint="Taille"
            android:inputType="numberDecimal"
            android:layout_weight="1"
        />
    </LinearLayout>
    <RadioGroup
        android:id="@+id/group"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:checkedButton="@+id/radio2"
        android:orientation="horizontal"
    >

```

```

>
<RadioButton
    android:id="@+id/radio1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Metre"
/>
<RadioButton
    android:id="@+id/radio2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Centimetre"
/>
</RadioGroup>
<CheckBox
    android:id="@+id/mega"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Mega fonction !"
/>
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
>
    <Button
        android:id="@+id/calcul"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Calculer l'IMC"
        android:layout_weight="1"
        android:layout_marginLeft="25dip"
        android:layout_marginRight="25dip"
    />
    <Button
        android:id="@+id/raz"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="RAZ"
        android:layout_weight="1"
        android:layout_marginLeft="25dip"
        android:layout_marginRight="25dip"
    />
</LinearLayout>
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Resultat: "
/>
<TextView
    android:id="@+id/result"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:text="Cliquez sur \textbf{Calculer l'IMC} pour obtenir un resultat."

```

```

/>
</LinearLayout>

```

De manière générale, on évite d'empiler les LinearLayout (avoir un LinearLayout dans un LinearLayout, dans un LinearLayout, etc.), c'est mauvais pour les performances d'une application. Le layout ci-dessous permet de placer les éléments les uns en fonction des autres (RelativeLayout). Le problème de ce layout, c'est qu'une petite modification dans l'interface graphique peut provoquer de grosses modifications dans tout le fichier XML, il faut donc savoir par avance très précisément ce qu'on veut faire. Cependant, c'est le plus puissant tout en étant l'un des moins gourmands en ressources.

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <TextView
        android:id="@+id/textPoids"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Poids : "
        android:textStyle="bold"
        android:textColor="#FF0000"
    />
    <EditText
        android:id="@+id/poids"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:hint="Poids"
        android:inputType="numberDecimal"
        android:layout_toRightOf="@id/textPoids"
        android:layout_alignParentRight="true"
    />
    <TextView
        android:id="@+id/textTaille"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Taille : "
        android:textStyle="bold"
        android:textColor="#FF0000"
        android:gravity="left"
        android:layout_below="@id/poids"
    />
    <EditText
        android:id="@+id/taille"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:hint="Taille"
        android:inputType="numberDecimal"
        android:layout_below="@id/poids"
        android:layout_toRightOf="@id/textTaille"
        android:layout_alignParentRight="true"
    />
    <RadioGroup
        android:id="@+id/group"
        android:layout_width="wrap_content"

```

```

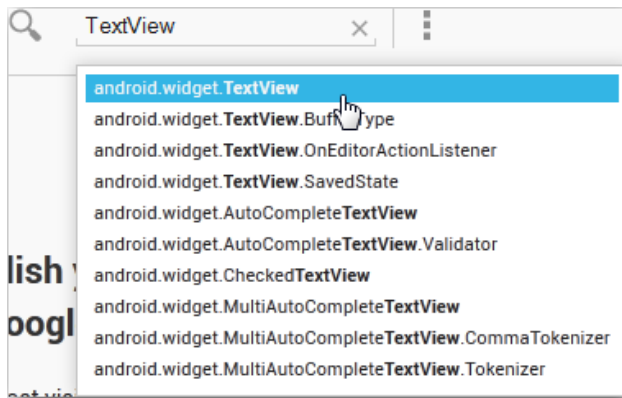
    android:layout_height="wrap_content"
    android:checkedButton="@+id/radio2"
    android:orientation="horizontal"
    android:layout_below="@id/taille"
  >
  <RadioButton
    android:id="@+id/radio1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Metre"
  />
  <RadioButton
    android:id="@+id/radio2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Centimetre"
  />
</RadioGroup>
<CheckBox
  android:id="@+id/mega"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text="Mega fonction !"
  android:layout_below="@id/group"
/>
<Button
  android:id="@+id/calcul"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text="Calculer l'IMC"
  android:layout_below="@id/mega"
  android:layout_marginLeft="25dip"
/>
<Button
  android:id="@+id/raz"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text="RAZ"
  android:layout_below="@id/mega"
  android:layout_alignRight="@id/taille"
  android:layout_marginRight="25dip"
/>
<TextView
  android:id="@+id/resultPre"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text="Resultat: "
  android:layout_below="@id/calcul"
/>
<TextView
  android:id="@+id/result"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:text="Cliquez sur \textbf{Calculer l'IMC} pour obtenir un resultat."

```

```
        android:layout_below="@id/resultPre"
    />
</RelativeLayout>
```

2 Utiliser la documentation

Dans le champ de recherche en haut à gauche on va insérer le nom de la classe que l'on recherche : `TextView`. Une liste s'affiche et permet de sélectionner la classe qui pourrait nous intéresser :



On a cliqué sur `Android.widget.TextView` puisque c'est celle qui nous intéresse. Nous arrivons alors sur une page qui vous décrit toutes les informations possibles et imaginables sur la classe `TextView` :

Ensuite, on peut voir les classes qui dérivent directement de cette classe (Known Direct Subclasses) et les classes qui en dérivent indirectement, c'est-à-dire qu'un des ancêtres de ces classes dérive de `View` (Known Indirect Subclasses). Enfin, on trouve en haut à droite un résumé des différentes sections qui se trouvent dans le document :

1. Nested Classes est la section qui regroupe toutes les classes internes. Vous pouvez cliquer sur une classe interne pour ouvrir une page similaire à celle de la classe `View`.
2. XML Attrs est la section qui regroupe tous les attributs que peut prendre un objet de ce type en XML. Pour chaque attribut XML on trouve associé un équivalent Java.
3. Constants est la section qui regroupe toutes les constantes dans cette classe.
4. Fields est la section qui regroupe toutes les structures de données constantes dans cette classe (listes et tableaux).
5. Ctors est la section qui regroupe tous les constructeurs de cette classe.
6. Methods est la section qui regroupe toutes les méthodes de cette classe.
7. Protected Methods est la section qui regroupe toutes les méthodes protégées (accessibles uniquement par cette classe ou les enfants de cette classe).