Développement des interfaces graphiques

1. Objectif

L'objectif principal de ce chapitre est de présenter comment définir une interface graphique avec les ConstraintLayout.

On peut définir l'interface d'un contrôleur (Activité ou Fragment) de deux façons différentes :

* Statiquement dans un fichier XML.
* Dynamiquement via le contrôleur.

Dans ce chapitre, seule la façon statique est présentée.

1. Les fichiers de type Layout
   1. Présentation

Le dossier layouts du dossier res contient tous les fichiers définissant les interfaces graphiques d'une application android. Ces fichiers définissant les interfaces graphiques sont des fichiers XML.

* 1. Fonctionnement de l'interface utilisateur.

Tous les éléments de l'interface utilisateur d'une application Android sont construits à l'aide d'objets de types View et ViewGroup.

Un objet de type View est un objet qui dessine quelque chose sur l'écran avec lequel l'utilisateur peut interagir.

Un objet de type ViewGroup est un objet qui contient d'autres objets View afin de définir la disposition de l'interface.

Android fournit tout un panel de View et ViewGroup qui offrent la possibilité de créer des interfaces graphiques riches.

L'interface utilisateur de chaque composant de votre application est définie à l'aide d'une hiérarchie d'objets View et ViewGroup.

Chaque objet de type ViewGroup est un conteneur invisible qui organise des objets enfants de types View et ViewGroup.



* 1. Les conteneurs de type ViewGroup

Il est nécessaire d'utiliser des ViewGroup pour positionner des composants dans votre interface graphique. Ces ViewGroup sont appelés "Layout" :

Voici les quatre principaux :

|  |  |
| --- | --- |
| Layout | Description |
| ConstraintLayout | Permet de spécifier l'emplacement des composants les uns par rapport aux autres avec des fonctionnalités plus avancées que celles du RelativeLayout. |
| GridLayout | Permet une mise en page qui place les composants dans une grille rectangulaire. |
| LinearLayout | Permet une mise en page qui organise les composants en une seule ligne verticale ou horizontale. Il crée un scroll bar si la longueur de la fenêtre dépasse la longueur de l'écran. |
| RelativeLayout | Permet de spécifier l'emplacement des composants les uns par rapport aux autres. |

Celui recommandé par google est le ConstraintLayout et c'est celui-ci qui sera présenté dans ce livre.

* 1. Attributs des composants d'un ConstraintLayout

Tous les composants d'un ConstraintLayout doivent avoir des informations sur leur manière de se positionner à l'horizontal et à la vertival. Pour se faire, un panel d'attributs sont disponibles.

|  |  |
| --- | --- |
| Attribut | Description |
| layout\_constraintLeft\_toLeftOf | Positionne le bord gauche de l'élément au même niveau que le bord gauche de l'élément passé en paramètre. |
| layout\_constraintLeft\_toRightOf | Positionne le bord gauche de l'élément au même niveau que le bord droit de l'élément passé en paramètre. |
| layout\_constraintRight\_toLeftOf | Positionne le bord droit de l'élément au même niveau que le bord gauche de l'élément passé en paramètre. |
| layout\_constraintRight\_toRightOf | Positionne le bord droit de l'élément au même niveau que le bord droit de l'élément passé en paramètre. |
| layout\_constraintTop\_toTopOf | Positionne le bord haut de l'élément au même niveau que le bord haut de l'élément passé en paramètre. |
| layout\_constraintTop\_toBottomOf | Positionne le bord haut de l'élément au même niveau que le bord bas de l'élément passé en paramètre. |
| layout\_constraintBottom\_toTopOf | Positionne le bord bas de l'élément au même niveau que le bord haut de l'élément passé en paramètre. |
| layout\_constraintBottom\_toBottomOf | Positionne le bord bas de l'élément au même niveau que le bord bas de l'élément passé en paramètre. |
| layout\_constraintBaseline\_toBaselineOf | Positionne la ligne de base de l'élément au même niveau que la ligne de base de l'élément passé en paramètre. |
| layout\_constraintStart\_toEndOf | Positionne l'élément à la fin de l'élément passé en paramètre. |
| layout\_constraintStart\_toStartOf | Positionne l'élément au début de l'élément passé en paramètre. |
| layout\_constraintEnd\_toStartOf | Positionne la fin de l'élément au début de l'élément passé en paramètre. |
| layout\_constraintEnd\_toEndOf | Positionne la fin de l'élément à la fin de l'élément passé en paramètre. |
| android:layout\_marginStart | Applique une marge entre le début de l'élément et l'élément. |
| android:layout\_marginEnd | Applique une marge entre la fin de l'élément et l'élément. |
| android:layout\_marginLeft | Applique une marge entre l'élément et l'élément à sa gauche. |
| android:layout\_marginTop | Applique une marge entre l'élément et l'élément au-dessus. |
| android:layout\_marginRight | Applique une marge entre l'élément et l'élément à sa droite. |
| android:layout\_marginBottom | Applique une marge entre l'élément et l'élément en dessous. |
| layout\_goneMarginStart | Quand l'élément cible permettant de se positionner devient invisible, alors cet attribut prend le relais pour définir une nouvelle marge de début. |
| layout\_goneMarginEnd | Quand l'élément cible permettant de se positionner devient invisible, alors cet attribut prend le relais pour définir une nouvelle marge de fin. |
| layout\_goneMarginLeft | Quand l'élément cible permettant de se positionner devient invisible, alors cet attribut prend le relais pour définir une nouvelle marge de gauche. |
| layout\_goneMarginTop | Quand l'élément cible permettant de se positionner devient invisible, alors cet attribut prend le relais pour définir une nouvelle marge du dessus. |
| layout\_goneMarginRight | Quand l'élément cible permettant de se positionner devient invisible, alors cet attribut prend le relais pour définir une nouvelle marge de droite. |
| layout\_goneMarginBottom | Quand l'élément cible permettant de se positionner devient invisible, alors cet attribut prend le relais pour définir une nouvelle marge du dessous. |
| layout\_constraintHorizontal\_bias | Permet de redéfinir la position centrée horizontalement de l'élément. |
| layout\_constraintVertical\_bias | Permet de redéfinir la position centrée verticalement de l'élément. |
| layout\_constraintCircle | Positionne le centre de l'élément par rapport au centre de l'élément passé en paramètre en prenant en compte un angle et une distance. Pour définir l'angle et la distance, il faut utiliser les attributs layout\_constraintCircleRadius et layout\_constraintCircleAngle. |
| layout\_constraintCircleRadius | Permet de définir la distance entre l'élément et l'élément passé en paramètre de l'attribut layout\_constraintCircle. |
| layout\_constraintCircleAngle | Permet de définir l'angle entre l'élément et l'élément passé en paramètre de l'attribut layout\_constraintCircle. |
| android:minWidth | Permet de définir la largeur minimale de l'élément. |
| android:minHeight | Permet de définir la hauteur minimale de l'élément. |
| android:maxWidth | Permet de définir la largeur maximale de l'élément. |
| android:maxHeight | Permet de définir la hauteur maximale de l'élément. |
| app:layout\_constrainedWidth | Permet de prendre en compte les contraintes de largeur même si la largeur est définie à WRAP\_CONTENT. |
| app:layout\_constrainedHeight | Permet de prendre en compte les contraintes de hauteur même si la hauteur est définie à WRAP\_CONTENT. |
| layout\_constraintWidth\_min | Attribut pris en compte lorsque la taille de l'élément est définie avec la valeur MATCH\_CONSTRAINT ou 0dp. Cet attribut permet de définir la largeur minimum de l'élément. |
| layout\_constraintHeight\_min | Attribut pris en compte lorsque la taille de l'élément est définie avec la valeur MATCH\_CONSTRAINT ou 0dp. Cet attribut permet de définir la hauteur minimum de l'élément. |
| layout\_constraintWidth\_max | Attribut pris en compte lorsque la taille de l'élément est définie avec la valeur MATCH\_CONSTRAINT ou 0dp. Cet attribut permet de définir la largeur maximale de l'élément. |
| layout\_constraintHeight\_max | Attribut pris en compte lorsque la taille de l'élément est définie avec la valeur MATCH\_CONSTRAINT ou 0dp. Cet attribut permet de définir la hauteur maximale de l'élément. |
| layout\_constraintWidth\_percent | Attribut pris en compte lorsque la taille de l'élément est définie avec la valeur MATCH\_CONSTRAINT ou 0dp. Cet attribut permet de définir la largeur en pourcentage. |
| layout\_constraintHeight\_percent | Attribut pris en compte lorsque la taille de l'élément est définie avec la valeur MATCH\_CONSTRAINT ou 0dp. Cet attribut permet de définir la hauteur en pourcentage. |
| layout\_constraintHorizontal\_chainStyle | Permet de définir un comportement de groupe sur un axe horizontal. |
| layout\_constraintVertical\_chainStyle | Permet de définir un comportement de groupe sur un axe vertical. |

* 1. Les principales View

Un objet de type View est un objet qui dessine quelque chose sur l'écran avec lequel l'utilisateur peut interagir.

Voici les principaux :

|  |  |
| --- | --- |
| Composant | Description |
| Button | Permet de définir un bouton standard. |
| ToggleButton | Permet de définir un bouton avec un indicateur visuel. |
| CheckBox | Permet de définir une liste de choix multiples. |
| RadioButton | Permet de définir une liste avec un choix unique. |
| Spinner | Permet de définir une liste de choix déroulante. |
| ProgressBar | Permet de définir une barre de progression soit avec indicateur de progression soit sans indicateur de progression. |
| SeekBar | Permet de sélectionner une valeur à l'aide d'une barre visuelle. |
| RatingBar | Permet de sélectionner une note à l'aide d'une barre d'étoiles. |
| Switch | Permet de définir un interrupteur on/off. |
| TextView | Permet d'afficher du texte. |
| EditText | Permet de saisir du texte. |
| ListView | Permet d'afficher une liste d'items sous forme de liste. |
| GridView | Permet de définir une liste d'items sous forme de grille. |
| WebView | Permet d'afficher une page web locale ou distante. |
| ImageButton | Permet d'afficher un bouton avec une image intégrée. |
| ImageView | Permet d'afficher une image. |
| VideoView | Permet d'afficher une vidéo. |
| TimePicker | Permet d'afficher un écran pour sélectionner une heure. |
| DatePicker | Permet d'afficher un écran pour sélectionner une date. |
| GuideLine | Permet de créer une ligne transparente permettant de positionner des éléments. Disponible uniquement avec les ConstraintLayout. |

Voici leurs principales propriétés communes :

|  |  |
| --- | --- |
| Propriété | Description |
| Id | Permet de définir l'identifiant du composant. Il est vivement recommandé de définir un identifiant pour chaque composant. |
| Layout\_width | Permet de définir la largeur du composant. Trois façons de définir la valeur de cet attribut :  WRAP\_CONTENT (taille fixée par rapport au contenu)  MATCH\_PARENT (taille fixée par rapport à l’élément englobant)  Valeur en dur avec l'unité sp |
| Layout\_height | Permet de définir la hauteur du composant. Trois façons de définir la valeur de cet attribut :  WRAP\_CONTENT (taille fixée par rapport au contenu)  MATCH\_PARENT (taille fixée par rapport à l’élément englobant)  Valeur en dur avec l'unité sp |
| Text | Permet de définir le texte de présentation du composant. |
| onClick | Permet de définir une fonction qui sera exécutée lors d'un clic sur le composant. |
| Padding | Permet de définir un padding sur le composant. Le padding permet de définir l'écart entre le contenu interne du composant et ses côtés. |
| visibility | Permet de définir la visibilité du composant. Il y a trois valeurs possibles :  Visible  Gone  Invisible |

* 1. Démonstration 1

Cette première démonstration permet de créer une horloge. Cette démonstration est accompagnée du rendu visuel ainsi que des sources commentées permettant de réaliser l'affichage.

* Les exemples sont réalisés à chaque fois dans un nouveau projet dont l'activité principale se nomme MainActivity.
  + 1. Rendu final de la démonstration



* + 1. Source de la démonstration

Dans cette démonstration, il est intéressant de visualiser les éléments suivants :

* La balise ConstraintLayout
* La balise GuideLine
* L'élément central dont l'id est tv\_centre
* Les attributs constraintCircle, constraintCircleRadius et constraintCircleAngle des éléments en orbites autour de l'élément central.

Activity\_main.xml

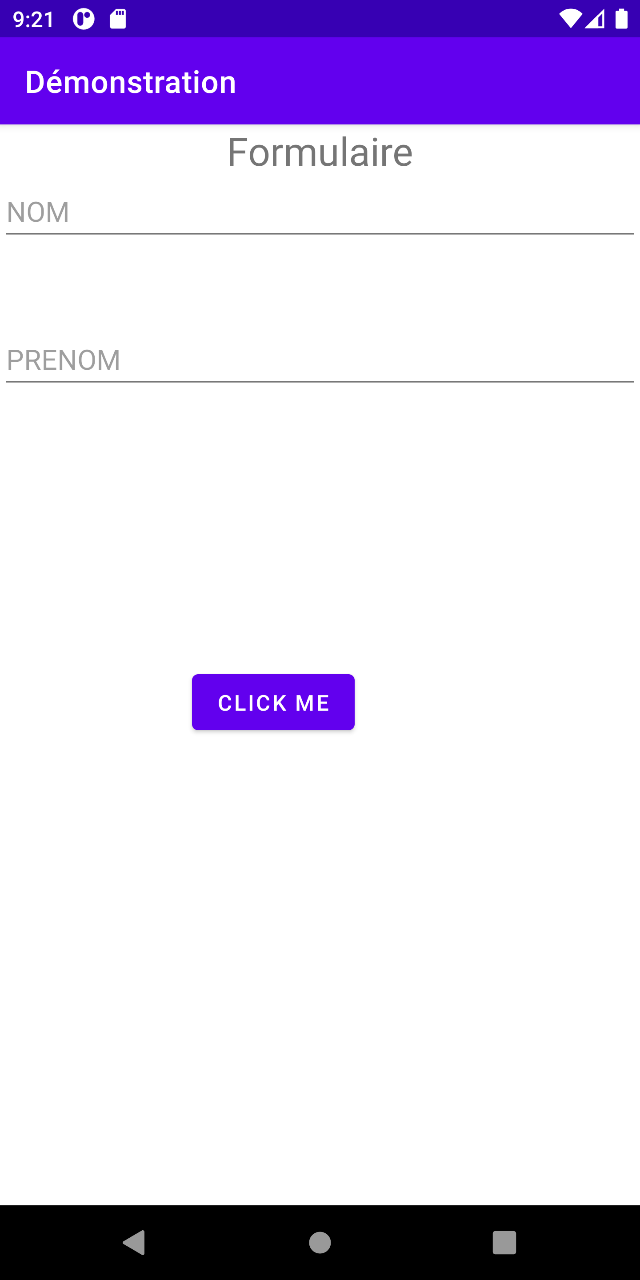
<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity">

<!--Permet de définir une ligne horizontale transparente au milieu -->  
 <androidx.constraintlayout.widget.Guideline  
 android:id="@+id/guidelineH"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 app:layout\_constraintGuide\_percent="0.5" />  
  
<!--Permet de définir une ligne verticale transparente au milieu -->  
 <androidx.constraintlayout.widget.Guideline  
 android:id="@+id/guidelineV"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 app:layout\_constraintGuide\_percent="0.5" />  
  
<!-- Positionne le centre de l'horloge grâce aux GuideLine -->  
 <TextView  
 android:id="@+id/tv\_centre"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="O"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 />  
  
<!-- Positionne le 1 à 70 dp du centre avec un décalage de 30 degrés -->  
 <TextView  
 android:id="@+id/tv\_1"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="1"  
 app:layout\_constraintCircle="@id/tv\_centre"  
 app:layout\_constraintCircleAngle="30"  
 app:layout\_constraintCircleRadius="70dp"  
 tools:ignore="MissingConstraints" />  
  
<!-- Positionne le 2 à 70 dp du centre avec un décalage de 60 degrés -->  
 <TextView  
 android:id="@+id/tv\_2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="2"  
 app:layout\_constraintCircle="@id/tv\_centre"  
 app:layout\_constraintCircleAngle="60"  
 app:layout\_constraintCircleRadius="70dp"  
 tools:ignore="MissingConstraints" />  
  
 <TextView  
 android:id="@+id/tv\_3"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="3"  
 app:layout\_constraintCircle="@id/tv\_centre"  
 app:layout\_constraintCircleAngle="90"  
 app:layout\_constraintCircleRadius="70dp"  
 tools:ignore="MissingConstraints" />  
  
 <TextView  
 android:id="@+id/tv\_4"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="4"  
 app:layout\_constraintCircle="@id/tv\_centre"  
 app:layout\_constraintCircleAngle="120"  
 app:layout\_constraintCircleRadius="70dp"  
 tools:ignore="MissingConstraints" />  
  
 <TextView  
 android:id="@+id/tv\_5"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="5"  
 app:layout\_constraintCircle="@id/tv\_centre"  
 app:layout\_constraintCircleAngle="150"  
 app:layout\_constraintCircleRadius="70dp"  
 tools:ignore="MissingConstraints" />  
  
 <TextView  
 android:id="@+id/tv\_6"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="6"  
 app:layout\_constraintCircle="@id/tv\_centre"  
 app:layout\_constraintCircleAngle="180"  
 app:layout\_constraintCircleRadius="70dp"  
 tools:ignore="MissingConstraints" />  
  
 <TextView  
 android:id="@+id/tv\_7"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="7"  
 app:layout\_constraintCircle="@id/tv\_centre"  
 app:layout\_constraintCircleAngle="210"  
 app:layout\_constraintCircleRadius="70dp"  
 tools:ignore="MissingConstraints" />  
  
 <TextView  
 android:id="@+id/tv\_8"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="8"  
 app:layout\_constraintCircle="@id/tv\_centre"  
 app:layout\_constraintCircleAngle="240"  
 app:layout\_constraintCircleRadius="70dp"  
 tools:ignore="MissingConstraints" />  
  
 <TextView  
 android:id="@+id/tv\_9"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="9"  
 app:layout\_constraintCircle="@id/tv\_centre"  
 app:layout\_constraintCircleAngle="270"  
 app:layout\_constraintCircleRadius="70dp"  
 tools:ignore="MissingConstraints" />  
  
 <TextView  
 android:id="@+id/tv\_10"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="10"  
 app:layout\_constraintCircle="@id/tv\_centre"  
 app:layout\_constraintCircleAngle="300"  
 app:layout\_constraintCircleRadius="70dp"  
 tools:ignore="MissingConstraints" />  
  
 <TextView  
 android:id="@+id/tv\_11"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="11"  
 app:layout\_constraintCircle="@id/tv\_centre"  
 app:layout\_constraintCircleAngle="330"  
 app:layout\_constraintCircleRadius="70dp"  
 tools:ignore="MissingConstraints" />  
  
 <TextView  
 android:id="@+id/tv\_12"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="12"  
 app:layout\_constraintCircle="@id/tv\_centre"  
 app:layout\_constraintCircleAngle="360"  
 app:layout\_constraintCircleRadius="70dp"  
 tools:ignore="MissingConstraints" />  
  
</androidx.constraintlayout.widget.ConstraintLayout>

* 1. Démonstration 2

Cette première démonstration permet de créer un formulaire de saisie très simple. Cette démonstration est accompagnée du rendu visuel ainsi que des sources commentées permettant de réaliser l'affichage.

* Les exemples sont réalisés à chaque fois dans un nouveau projet dont l'activité principale se nomme MainActivity.
  + 1. Rendu final de la démonstration



* + 1. Source de la démonstration

Dans cette démonstration, il est intéressant de visualiser les éléments suivants :

* La balise ConstraintLayout
* Les attributs de type layout\_constraint permettant de positionner les éléments.

Activity\_main.xml

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.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"  
 tools:context=".MainActivity">  
  
 <TextView  
 android:id="@+id/tv\_titre"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Formulaire"  
 android:textSize="25sp"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 />  
  
 <EditText  
 android:id="@+id/et\_nom"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 app:layout\_constraintTop\_toBottomOf="@id/tv\_titre"  
 android:hint="NOM"  
 />  
  
 <EditText  
 android:id="@+id/et\_prenom"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 app:layout\_constraintTop\_toBottomOf="@id/et\_nom"  
 android:layout\_marginTop="50dp"  
 android:hint="PRENOM"  
 />  
  
 <androidx.constraintlayout.widget.Guideline  
 android:id="@+id/gl\_vertical"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 app:layout\_constraintGuide\_percent="0.3"  
 />  
  
 <androidx.constraintlayout.widget.Guideline  
 android:id="@+id/gl\_horizontal"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 app:layout\_constraintGuide\_percent="0.5"  
 />  
  
 <Button  
 android:id="@+id/btn\_click\_me"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Click me"  
 app:layout\_constraintTop\_toTopOf="@id/gl\_horizontal"  
 app:layout\_constraintLeft\_toLeftOf="@id/gl\_vertical"  
 />  
  
</androidx.constraintlayout.widget.ConstraintLayout>

* Les guidelines sont des éléments non visibles permettant d'aider au positionnement des éléments visibles.
  1. Démonstration 3

Cette première démonstration permet de créer un formulaire de saisie avec une diversité d'éléments visuels. Cette démonstration est accompagnée du rendu visuel ainsi que des sources commentées permettant de réaliser l'affichage.

* Les exemples sont réalisés à chaque fois dans un nouveau projet dont l'activité principale se nomme MainActivity.
  + 1. Rendu final de la démonstration



* + 1. Source de la démonstration

Dans cette démonstration, il est intéressant de visualiser les éléments suivants :

* La balise ConstraintLayout
* Les attributs de type layout\_constraint permettant de positionner les éléments.
* La balise GuideLine
* La diversité des éléments

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

<androidx.constraintlayout.widget.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"

tools:context=".MainActivity"

android:padding="30dp"

>

<TextView

android:id="@+id/tv\_the\_questionnaire"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="The questionnaire"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintLeft\_toLeftOf="parent"

app:layout\_constraintRight\_toRightOf="parent"

android:textSize="25sp" />

<TextView

android:id="@+id/tv\_cava"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Est ce que ca va ?"

android:layout\_marginTop="30dp"

app:layout\_constraintTop\_toBottomOf="@id/tv\_the\_questionnaire"

app:layout\_constraintLeft\_toLeftOf="parent"

/>

<Switch

android:id="@+id/sw\_cava"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

app:layout\_constraintLeft\_toRightOf="@id/tv\_cava"

app:layout\_constraintRight\_toRightOf="parent"

app:layout\_constraintBaseline\_toBaselineOf="@id/tv\_cava"

/>

<TextView

android:id="@+id/tv\_question\_bilboquet"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

app:layout\_constraintTop\_toBottomOf="@id/tv\_cava"

android:text="Quel est ton niveau de bilboquet ?"

app:layout\_constraintLeft\_toLeftOf="parent"

app:layout\_constraintRight\_toRightOf="parent"

android:layout\_marginTop="30dp"

/>

<SeekBar

android:id="@+id/sb\_bilboquet"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

app:layout\_constraintTop\_toBottomOf="@id/tv\_question\_bilboquet"

android:layout\_marginTop="10dp"

/>

<TextView

android:id="@+id/tv\_mail"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Quel est ton mail ?"

app:layout\_constraintTop\_toBottomOf="@id/sb\_bilboquet"

app:layout\_constraintLeft\_toLeftOf="parent"

app:layout\_constraintRight\_toRightOf="parent"

android:layout\_marginTop="30dp"

/>

<EditText

android:id="@+id/et\_mail"

android:layout\_width="150dp"

android:layout\_height="wrap\_content"

app:layout\_constraintTop\_toBottomOf="@id/tv\_mail"

app:layout\_constraintLeft\_toLeftOf="parent"

app:layout\_constraintRight\_toRightOf="parent"

android:hint="MAIL"

/>

<androidx.constraintlayout.widget.Guideline

android:id="@+id/gl\_y"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:orientation="vertical"

app:layout\_constraintGuide\_percent="0.5"

/>

<androidx.constraintlayout.widget.Guideline

android:id="@+id/gl\_x"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="100dp"

android:orientation="horizontal"

app:layout\_constraintGuide\_begin="365dp"

app:layout\_constraintTop\_toBottomOf="@id/et\_mail" />

<Button

android:id="@+id/btn\_1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/titre\_ok"

app:layout\_constraintBottom\_toBottomOf="@id/gl\_x"

app:layout\_constraintRight\_toRightOf="@id/gl\_y"

android:layout\_margin="5dp"

/>

<Button

android:id="@+id/btn\_2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/titre\_ok"

app:layout\_constraintBottom\_toBottomOf="@id/gl\_x"

app:layout\_constraintLeft\_toLeftOf="@id/gl\_y"

android:layout\_margin="5dp"

/>

<Button

android:id="@+id/btn\_3"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/titre\_ok"

app:layout\_constraintTop\_toTopOf="@id/gl\_x"

app:layout\_constraintRight\_toRightOf="@id/gl\_y"

android:layout\_margin="5dp"

/>

<Button

android:id="@+id/btn\_4"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/titre\_ok"

app:layout\_constraintTop\_toTopOf="@id/gl\_x"

app:layout\_constraintLeft\_toLeftOf="@id/gl\_y"

android:layout\_margin="5dp"

/>

</androidx.constraintlayout.widget.ConstraintLayout>

1. Conclusion

Ce chapitre a présenté comment définir une IHM simple, l'utilisation des constraint Layout pour la mise en forme des IHM est à privilégier.