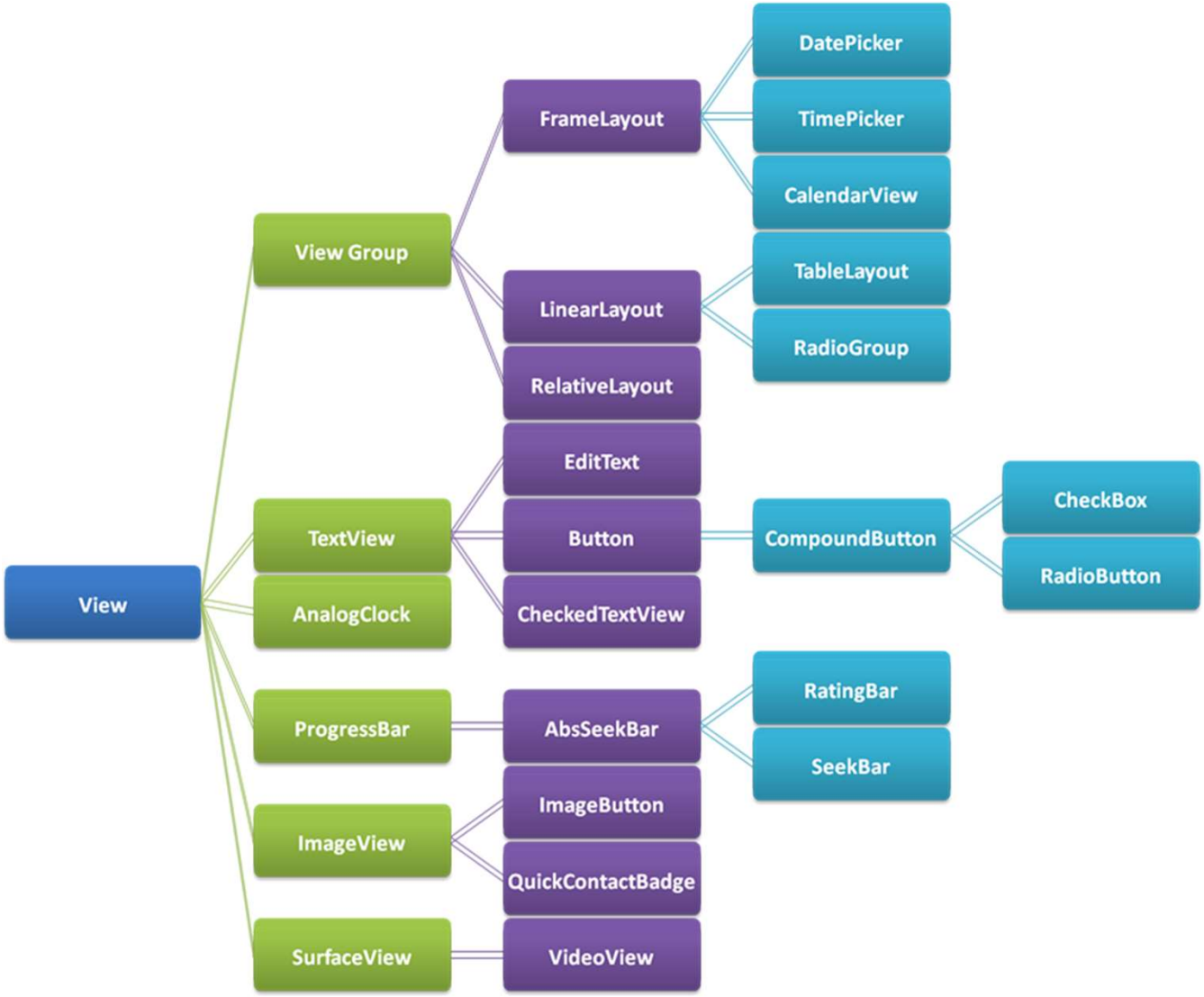
Android

IHM



Une vue est un élément affichable de l'interface utilisateur (classe de base android.view.View).

Une vue de type ViewGroup peut contenir d'autres vues.

Hiérarchie des classes de l'interface utilisateur

Exemple d'un composant de type View

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

<Button
xmlns:android="http://schemas.android.com/apk/res/android"

android:layout_width="match_parent"

android:layout_height="match_parent"

    android:text="Un bouton" >

</Button>
```

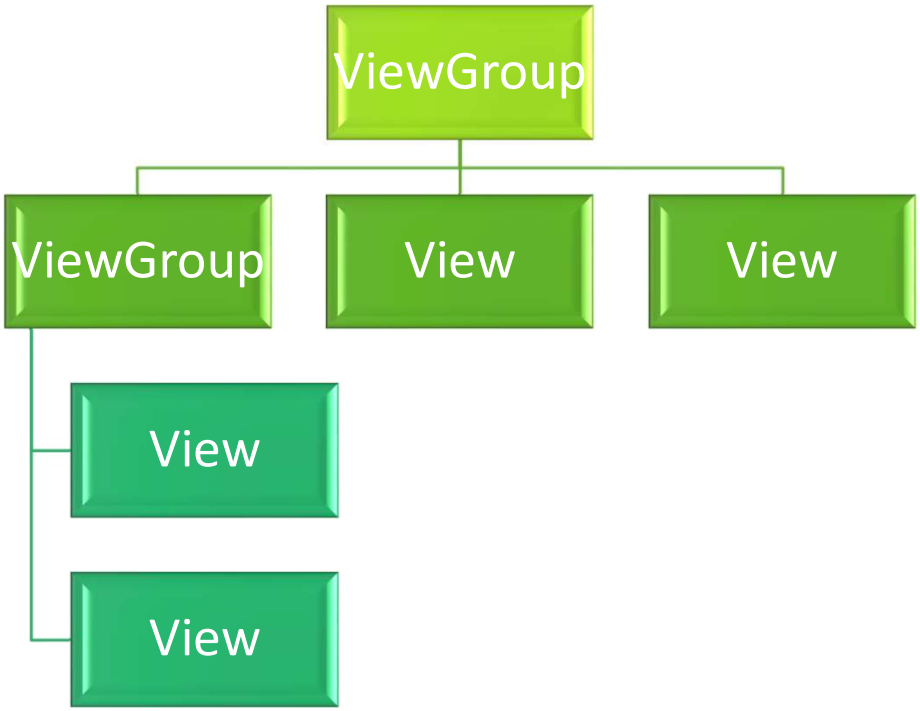
L'élément racine d'une vue doit déclarer le namespace « http://schemas.android.com/apk/res/android »

Valeurs possibles pour les attributs android:layout_width et android:layout_height:

- *match_parent* : l'élément remplit tout l'élément parent.
- *wrap_content* :le composant est dimensionné selon son contenu.

Id: l'id permet d'identifier la vue d'une manière unique dans l'arborescence. l'id est défini dans le document xml comme une chaîne, mais il est référencé dans le code java par un entier généré automatiquement.

Les vues



Une vue est un fichier xml enregistré dans le dossier res\Layout et qui contient une arborescence de composants graphique dont la racine est généralement un élément de type ViewGroup.

Le nom du fichier xml, par example main.xml permet de retrouver le layout dans le code java au travers de **R.layout.main**.

Pour afficher une vue à partir de l'activité:

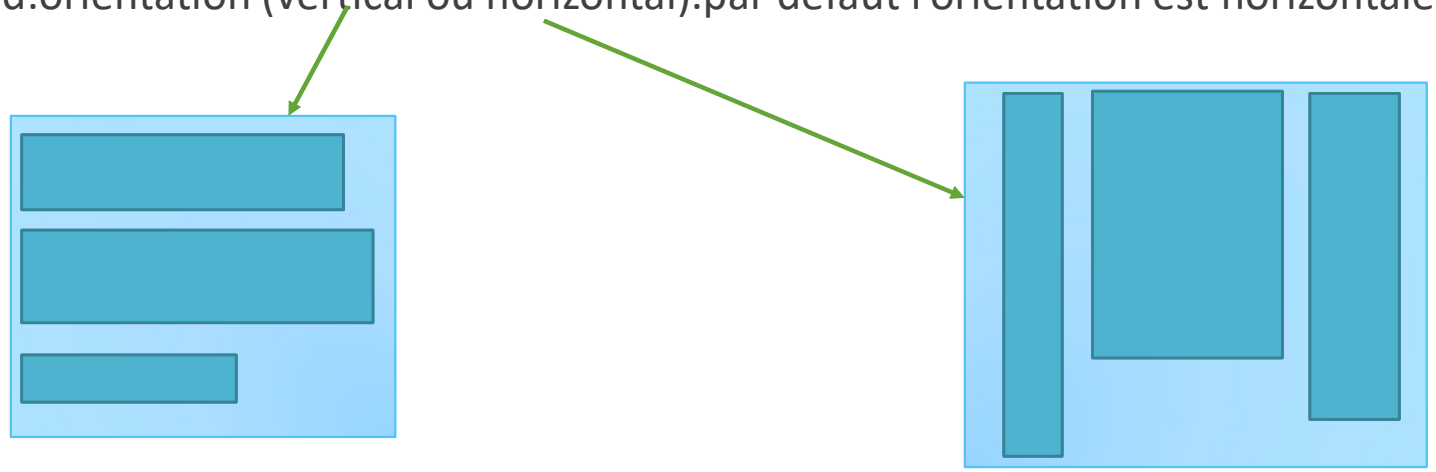
```
setContentView(R.layout.main);
```

Les gestionnaires de disposition

Un gestionnaire de disposition définit la structure visuelle d'une interface utilisateur.

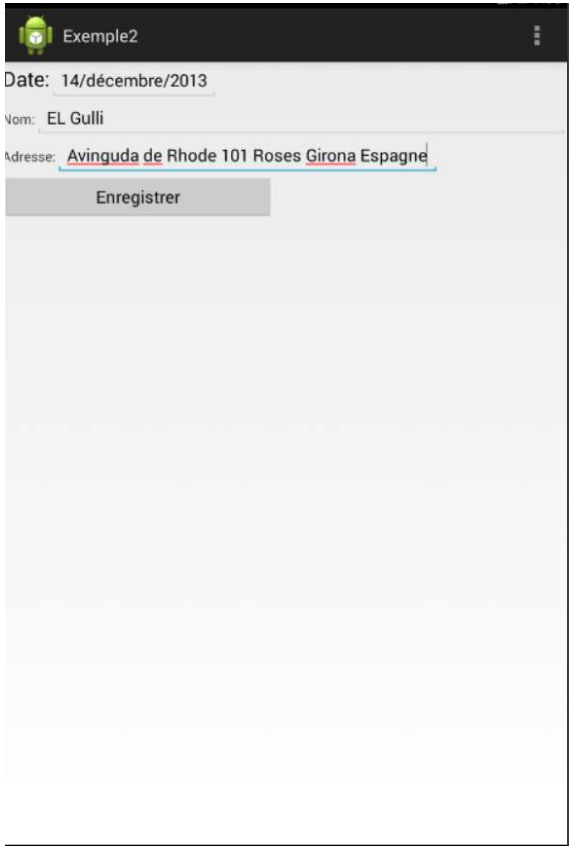
LinearLayout

Dispose les éléments de gauche à droite ou du haut vers le bas selon la valeur de la propriété android:orientation (vertical ou horizontal).par défaut l'orientation est horizontale.



Exemple

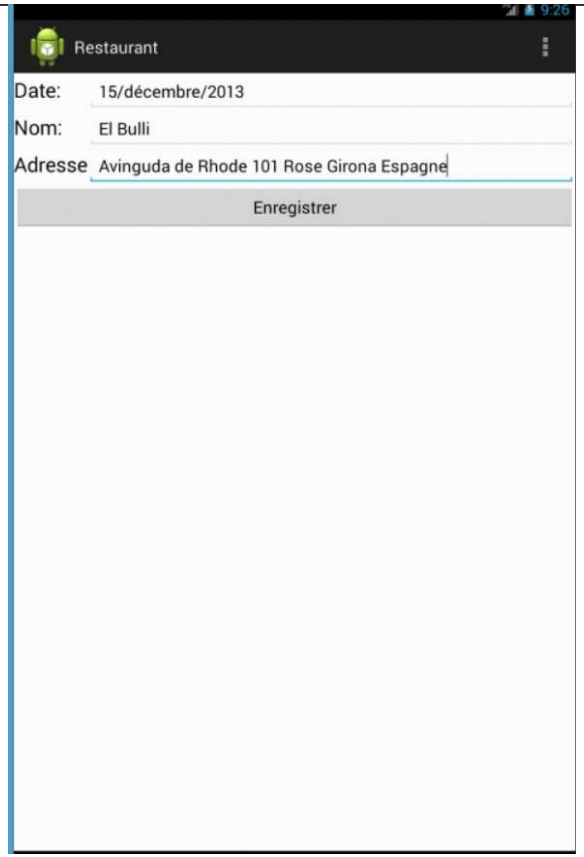
```
<LinearLayout
 xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <LinearLayout android:layout_width="match_parent"
        android:layout_height="wrap_content" >
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Date:"/>
        <EditText android:id="@+id/txtDate"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
    </LinearLayout>
    <LinearLayout android:layout_width="match_parent"
        android:layout_height="wrap_content" >
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Nom:" />
        <EditText
            android:id="@+id/txtNom"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />
    </LinearLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Adresse:" />
        <EditText
            android:id="@+id/txtAdresse"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
    </LinearLayout>
      <Button  android:id="@+id/bEnregistrer"
        android:layout_width="289dp"
        android:layout_height="wrap_content"
        android:text="Enregistrer" />
</LinearLayout>
```

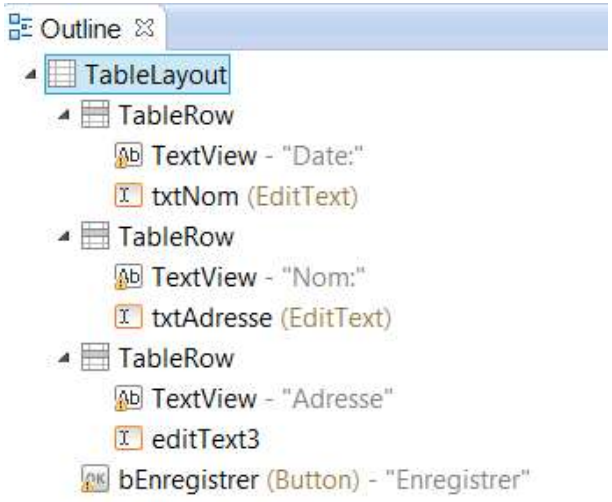


- LinearLayout: @+id/llv1
  - LinearLayout
    - TextView: @+id/textView1
    - EditText: @+id/txtNom
  - LinearLayout
    - TextView: @+id/textView2
    - EditText: @+id/txtAdresse
  - LinearLayout
    - TextView: @+id/textView3
    - EditText: @+id/editText3
  - Button: @+id/bEnregistrer

TableLayout

- Un TableLayout est composé d'éléments de type TableRow, il dispose les vues dans des lignes et des colonnes.
- Le nombre de colonnes est déterminé par la ligne qui contient le plus grand nombre de cellules.
- La largeur d'une colonne est définie par la cellule la plus large de cette colonne.
- android:stretchColumns="*1*": La colonne 1 (EditText) occupe la largeur restante dans la ligne.



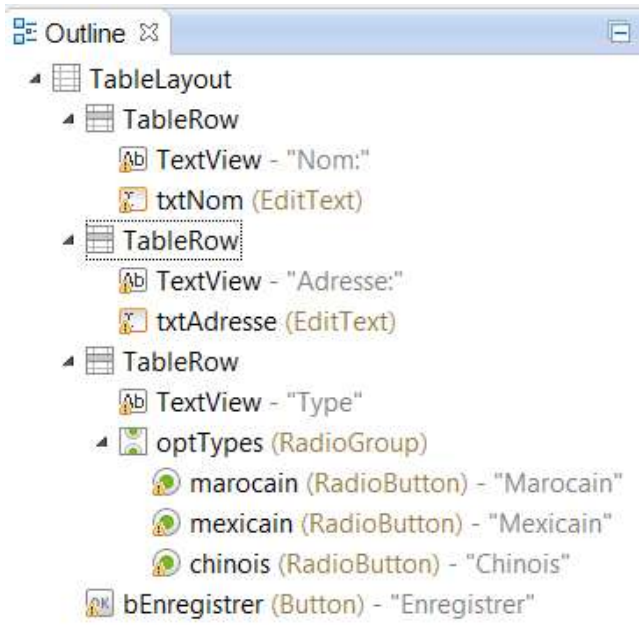



```
<TableLayout
xmlns:android="http://schemas.android.com/apk/res/and
roid"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:stretchColumns="1" >
    <TableRow>
        <TextView android:text="Date:" />
        <EditText android:id="@+id/txtDate" />
    </TableRow>
    <TableRow>
        <TextView android:text="Nom:" />
        <EditText android:id="@+id/txtNom" />
    </TableRow>
    <TableRow>
        <TextView android:text="Adresse" />
        <EditText android:id="@+id/txtAdresse" />
    </TableRow>
    <Button android:id="@+id/bEnregistrer"
        android:text="Enregistrer" />
</TableLayout>
```

L'activité

```
 override fun
onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity_main)
        /* Affichage de la date */
       val  formatDate=
SimpleDateFormat("dd/MMMM/yyyy",
Locale.FRANCE)
       val  date =
formatDate.format(Calendar.getInstance()
.getTime())
       val  txtDate =
findViewById<EditText>(R.id.txtDate)
        txtDate.setText(date)

/* Création et initialisation du bouton
Enregistrer */
        val  cmdEnregistrer =
findViewById<Button>(R.id.bEnregistrer);

cmdEnregistrer.setOnClickListener{

Toast.makeText(it.context,"Restaurant
ajouté",Toast.LENGTH_LONG).show()
       }
        }
```

Ajout d'un groupe de boutons radios




```
<TableRow>
    <RadioGroup android:id="@+id/optTypes" >
        <RadioButton
            android:id="@+id/marocain"
            android:text="Marocain"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"/>

        <RadioButton
            android:id="@+id/mexicain"
            android:text="Mexicain"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"/>

        <RadioButton
            android:id="@+id/chinois"
            android:text="Chinois"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"/>
    </RadioGroup>

</TableRow>
```

Ajout d'un groupe de boutons radios

```
val types = findViewById<RadioGroup>(R.id.optTypes)

val type_restaurant = when (types.checkedRadioButtonId) {

    R.id.marocain -> "Marocain"
    R.id.mexicain -> "Mexicain"
    R.id.chinois -> "Chinois"

    else -> "Indéfini"
}
val restaurant =
Restaurant(findViewById<EditText>(R.id.txtNom).editableText.toString(),
    findViewById<EditText>(R.id.txtAdresse).editableText.toString(),
    type_restaurant)
Toast.makeText(it.context, restaurant.toString() + " est ajouté",
Toast.LENGTH_LONG)
    .show()
```

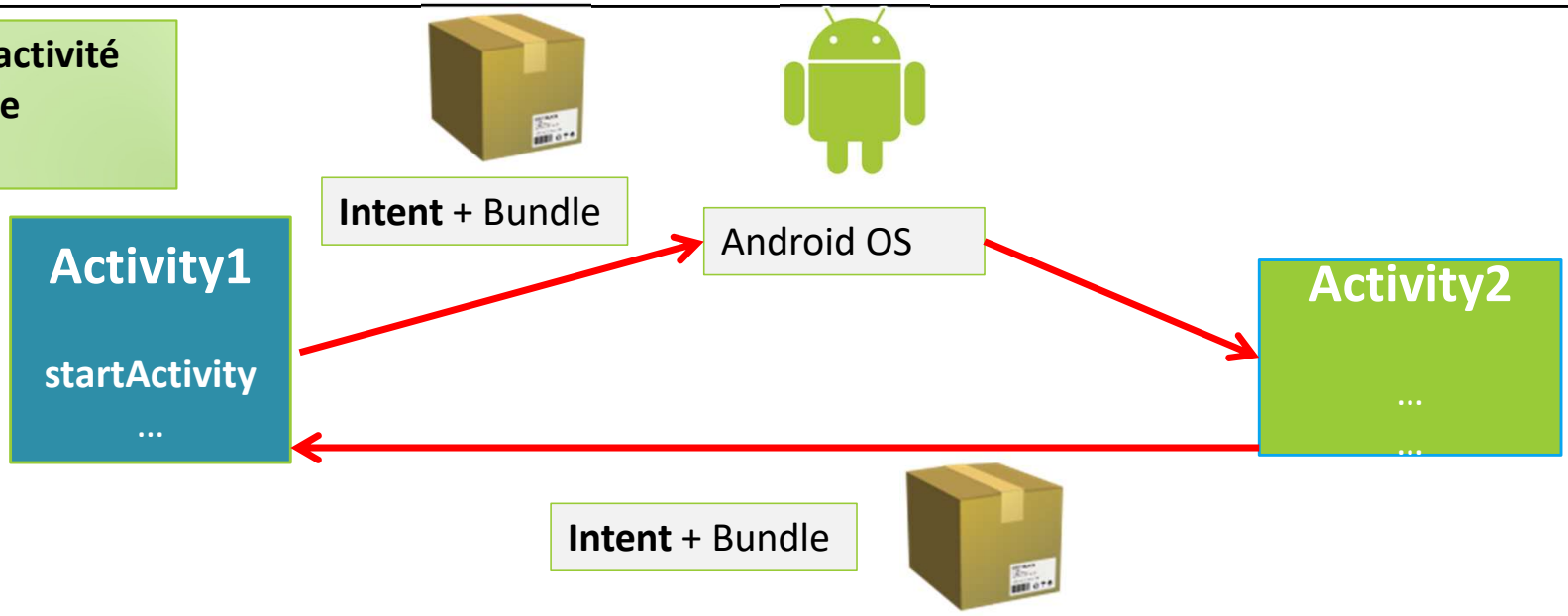
La méthode getCheckedRadioButtonId de RadioGroup retourne l'Id du bouton radio sélectionné.

Passer d'une activité à une autre

Un objet de type Intent est utilisé pour passer d'une activité à une autre.

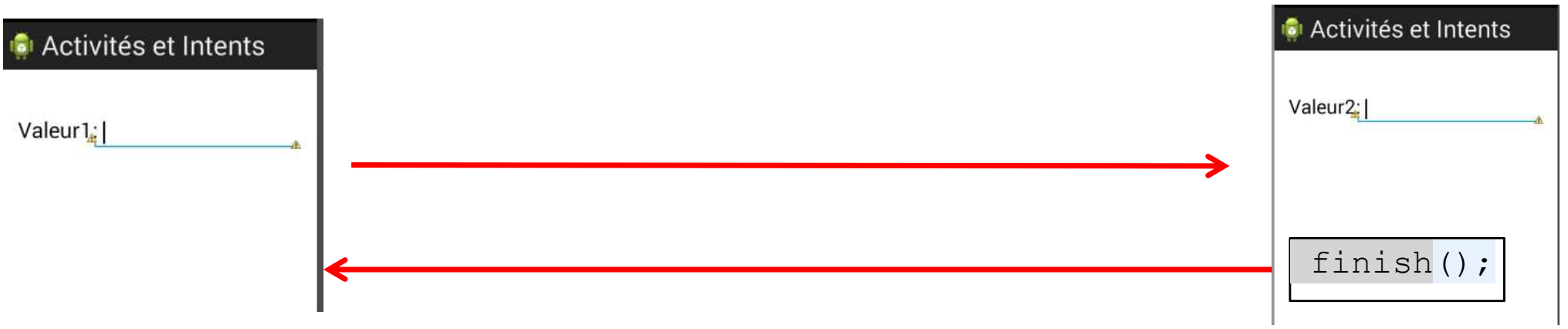
Un objet Intent (ou intention) est créé dans l'activité de départ

Démarre une activité sans attente de résultat



Scénario 1 : Afficher une activité sans attente de résultat et sans envoi de données.

```
val intentAct2= Intent (this,// Contexte
Activite2::class.java // Activité à démarrer
);
startActivity(intentAct2);
```



Scénario 2 : Afficher Activité sans attente de résultat et avec envoi de données.

```
val intentAct2 = Intent(this, act2::class.java)
 intentAct2.putExtra("valeur", 5)
 //L'objet Intent possède
 /* un Bundle nommé extra (un Bundle a une structure de type dictionnaire */
 startActivity(intentAct2)
```

Dans l'événement onCreate

```
val extra=  intent.getIntExtra("valeur", 0)
  //le deuxième paramètre est la valeur par défaut
 /*Pour un objet la classe doit être serialisable
 extra=intent.getSerializableExtra("restaurant");
  */
val t = findViewById<EditText>(R.id.t1)
t.setText(extra.toString())
```

Scénario 3 : Activité avec attente de résultat

Pour démarrer l'activité 2

```
startActivityForResult(intentAct2, 1);
```

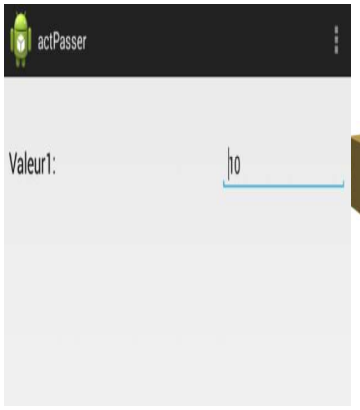
1 permet d'identifier la requête.

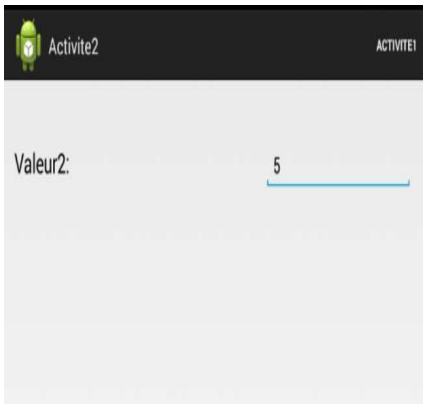
Réception du résultat dans activité1

```
override fun onActivityResult(requestCode: Int,
resultCode: Int, data: Intent?) {
    super.onActivityResult(requestCode, resultCode,
data)
        if (resultCode == 2) {
            val valeur = data?.getIntExtra("valeur", 0)
            val t = findViewById<TextView>(R.id.text2)
            t.text = valeur.toString()
        }

    }
}
```

Retour du résultat

```
val intentionRetour = Intent()
intentionRetour.putExtra("valeur",
10)
setResult(2, intentionRetour)
finish()
```

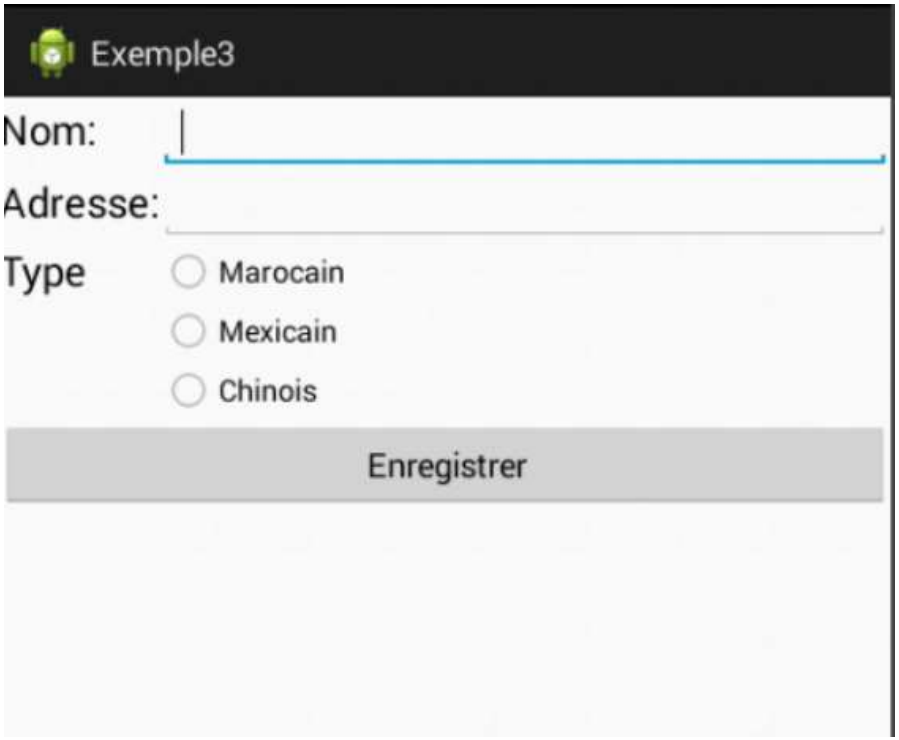




TP: Exemple3

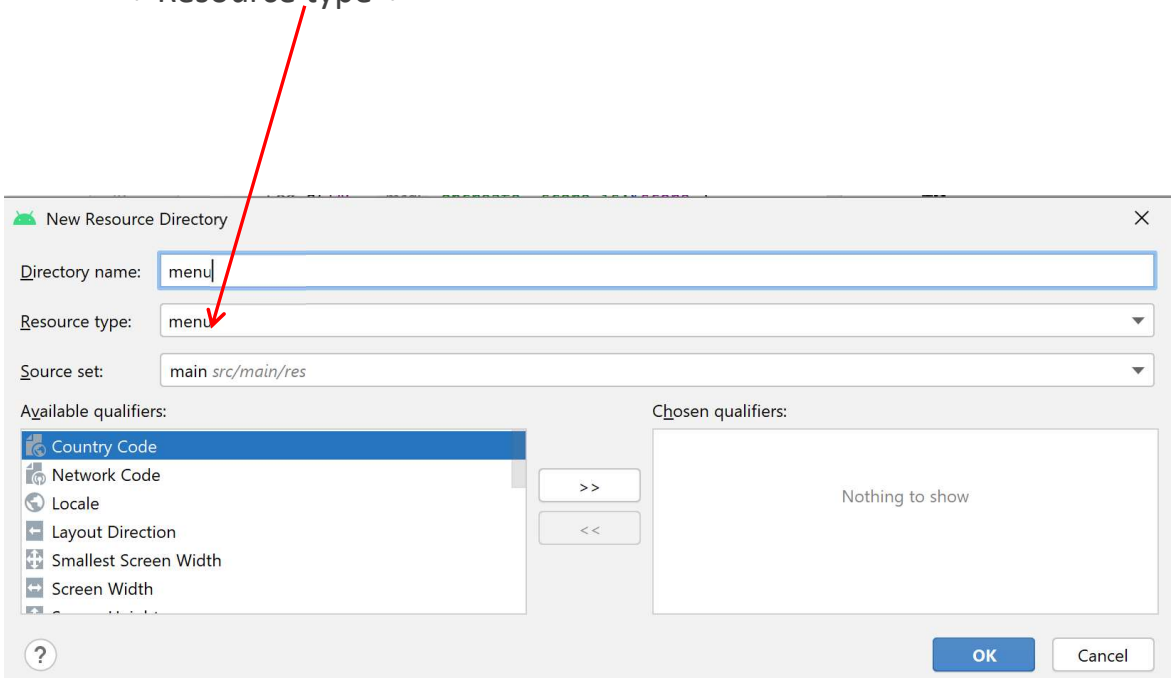
Etape1: ajout d'un groupe de boutons radios

- Ajouter le champ type dans la classe Restaurant
- Ajouter un constructeur à trois paramètres dans la classe
- Ajouter le groupe de boutons radios
- Mettre à jour « Enregistrer » pour tenir compte du type de restaurant



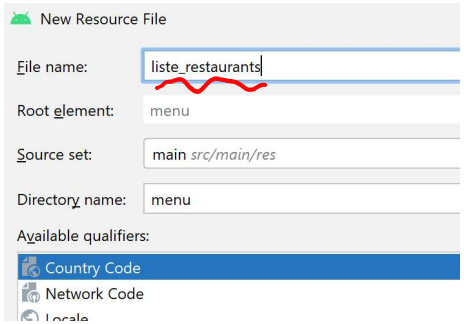
Les menus

- Un menu est un ensemble d'élément situés dans la barre d'action de l'application, et qui permettent d'exécuter des actions selon l'activité courante.
- Pour ajouter des menus dans l'application, il faut au préalable ajouter un dossier menu, dans la section « res »:
  - Cliquer sur la commande « Android resource directory » du menu contextuel associé à « res » et sélectionner « menu » dans la liste « Resource type »



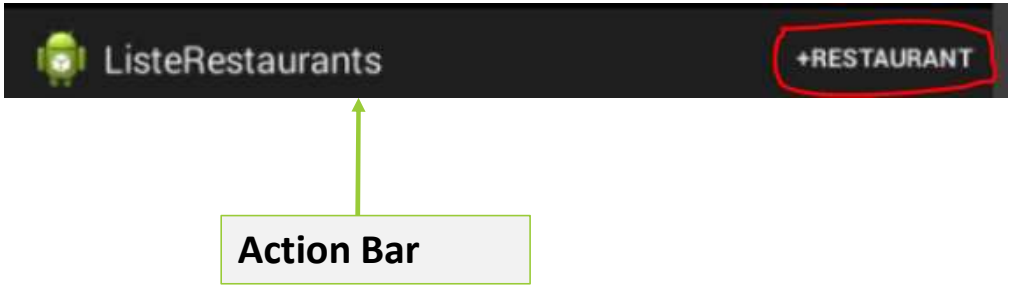
Les menus

- Ajouter un menu dans le dossier menu créé (New/Menu resource file) nommé liste_restaurant



Etape 5: Ajouter un élément de menu

- Dans le fichier de ressources res/menu/liste_restaurants.xml, modifier l'élément de menu existant (ou ajouter un nouveau):
  - id:*@+id/ajouter*
  - *Show as Action: cocher les cases « withText » et « alwyas » (l'élément de menu s'affichera dans la barre d'action)*
  - *Title: +Restaurant*



- L'affichage du menu dans la barre d'action est réalisé dans l'activité, à l'aide de la méthode onCreateOptionsMenu

```
override fun onCreateOptionsMenu(menu: Menu?): Boolean {
    menuInflater.inflate(R.menu.liste_restaurants,menu)
    return true
}
```

Evénement : clic sur un élément de menu

- Le clic sur un élément de menu déclenche l'exécution de la méthode « onOptionsItemSelected », qui fournit en argument l'objet MenuItem concerné par le clic.
- Pour ajouter la méthode onOptionsItemSelected à l'aide de l'assistant :
  - Clic droit sur le nom de la classe : Generate/ « Override Methods » et cocher la méthode dans la liste (Ctrl +O).

```
override fun onOptionsItemSelected(item: MenuItem): Boolean {
        if (item.itemId == R.id.ajouter) {
        … code à ajouter lors du clic sur le menu « ajouter »
        return true
    }
```