Laboratori 2: Projecte Integrat de Software

Projecte Integrat de Software UB 2019/20
Grau d'Enginyeria informàtica

Gerard Puch Camacho

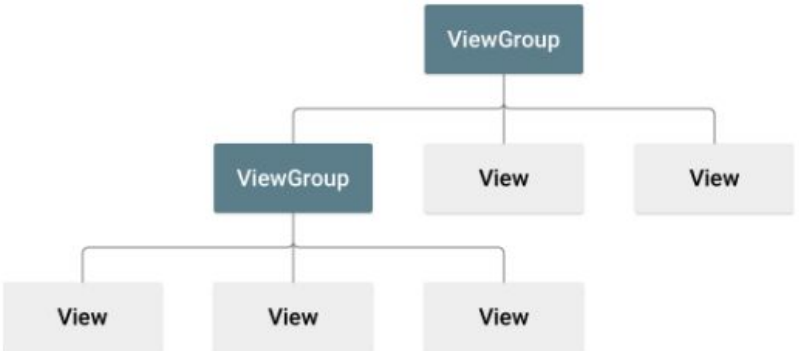
Disseny de la aplicaió - Interfície d'usuari

La interfície d'usuari de la teva app és tot allò que l'usuario pot veure i interactuar. Android ofereix una varietat de components IU prèviament compilats, com objectes de disseny estructurats i controls de IU que permeten compilar la interfície gràfica d'usuari per la teva app.

Disseny

L'estructura de la interfície utilitza la següent jerarquía d'objectes:

- **ViewGroup:** Contenidor invisible que defineix la estructura on estaran continguts altres ViewGroup o Views. Es denominen "dissenys" o "layouts".
- **View:** Principal classe que defineix tot allò amb el que l'usuari pot interactuar. Se'ls denomina "widget".



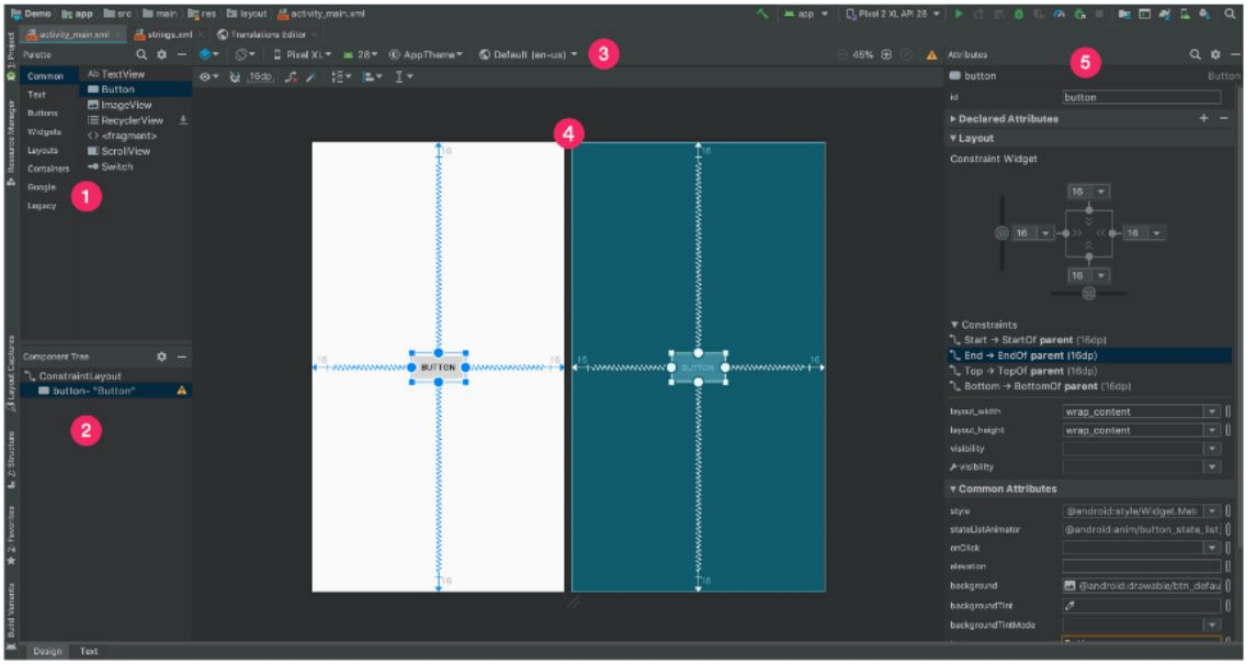
Per editar els layout de una app, existeixen dos mètodes:

- Amb codi XML, definint les diferents instàncies del teu disseny.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
              android:layout_width="match_parent"
              android:layout_height="match_parent"
              android:orientation="vertical" >
    <TextView android:id="@+id/text"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:text="Hello, I am a TextView" />
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button" />
</LinearLayout>
```

- Amb el Layout editor, un editor de disseny interactiu amb generació automàtica del codi XML.

1. **Palette**: Es la lista de vistas y grupos de vistas que puedes arrastrar a tu diseño.
2. **Component Tree**: Corresponde a la jerarquía de vistas para tu diseño.
3. **Toolbar**: Tiene los botones para configurar la apariencia de tu diseño en el editor y cambiar algunos atributos de diseño.
4. **Design editor**: Corresponde al diseño en la vista Design o Blueprint, o ambas.
5. **Attributes**: Contiene controles para los atributos de las vistas seleccionadas.



Layouts

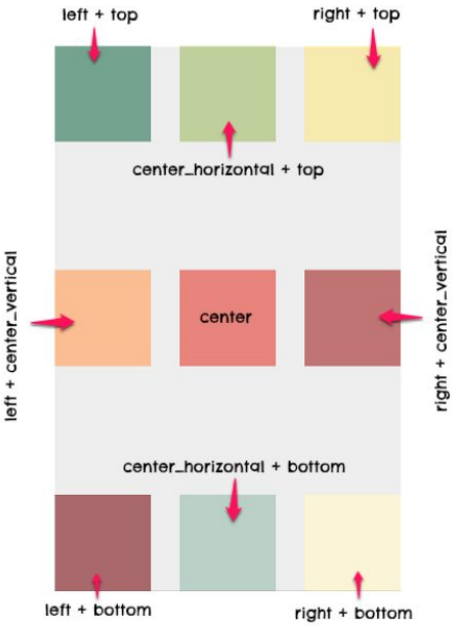
En els diferents layouts definits, existeixen paràmetres bàsics i comuns:

- **android:id="@+id/name"**: Identificador único per a cada element. La @ indica que és un parser intern de XML. Si s'afegeix +, significa que el id no està creat encara i es crea automàticament en el fitxer R.strings.
- **layout width, layout height**: Defineixen l'ample i alçada de qualsevol view. Si utilitzem "wrap_content", la mida s'ajusta a la mínima visualització correcta. Amb "match_parent", s'ajusta al màxim permès pel pare (el layout on és contingut l'element).

Frame Layout

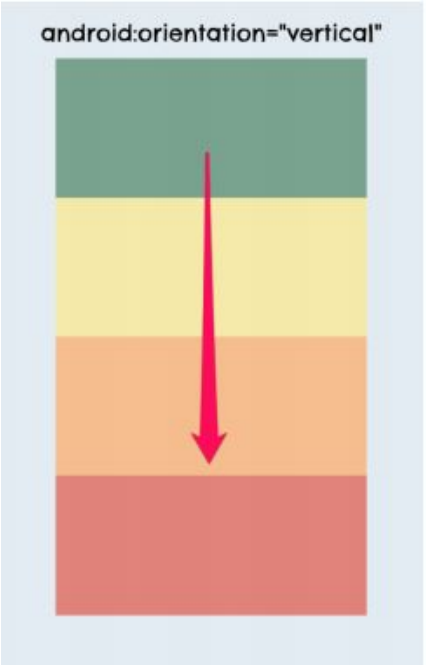
Està pensat per mostrar un sol element. Tot i així, es poden elaborar quadrícules, per fer-ho podem utilitzar el paràmetre **gravity** per alinear els FrameLayout fills com es desitji.

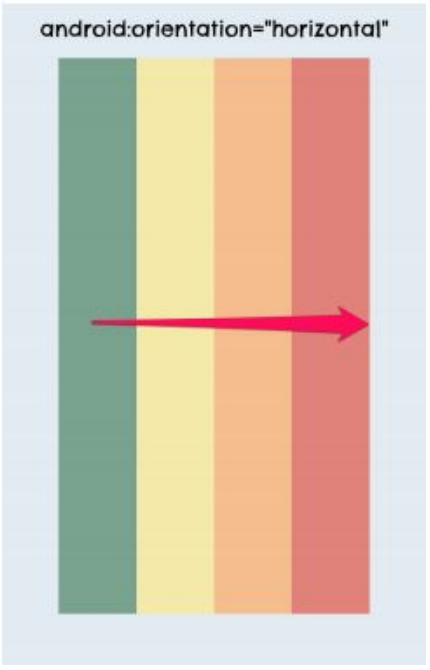
```
android:layout_gravity="right|bottom"
```



Linear Layout

Un Linear Layout distribueix tots els contents (Views) dins del ViewGroup al llarg de la dimensió establerta. La orientació d'aquest es pot definir amb el paràmetre **android:orientation**.





També existeix la opció **android:layout_weight** que permet definir la importància de cada View dins de la interfície. En el paràmetre weight es defineix a través d'un Int que representa la divisió dels diferents pesos.

```
android:weightSum="6"
```

Relative Layout

És l'element més flexible de tots. Permet una distribució més avançada i es determina segons com definim la posició relativa dels View respecte als altres. Els paràmetre per declarar la posició relativa són els següents:

- **android:layout_above**: Posiciona la vora inferior de l'element actual amb la vora superior del View declarat amb la id.
- **android:layout_centerHorizontal**: Utilitza true per indicar que el view estarà centrat en l'eix horitzontal respecte al pare.
- **android:layout_alignParentBottom**: Utilitza true per alinear la vora inferior del view amb la vora inferior dels pare.
- **android:layout_alignStart**: Alinea la vora inicial de l'element amb la vora inicial del view referida per id.

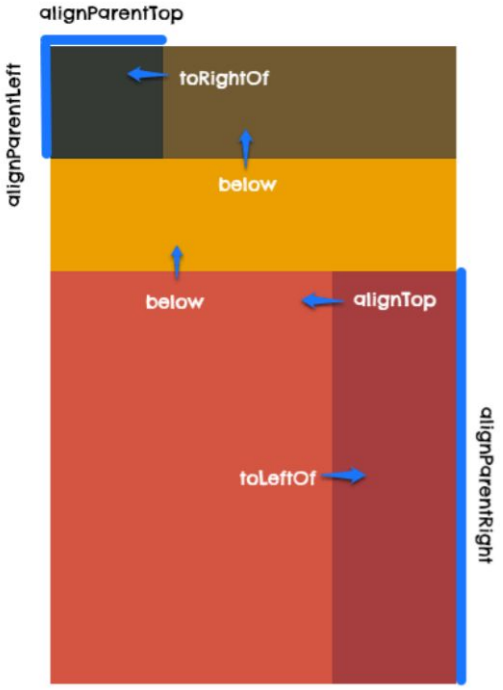


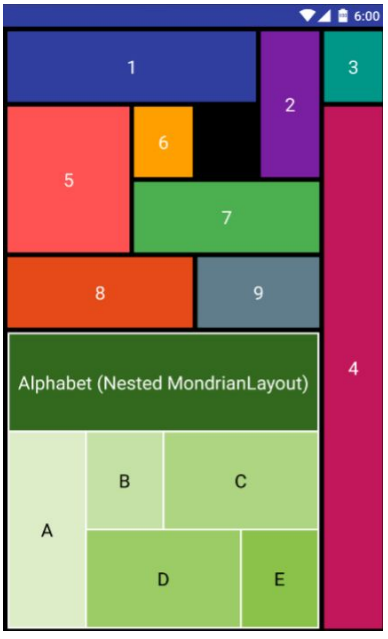
Table Layout

Distribució que serveix per crear taules al estil “excel”. Segueix un disseny seqüencial i s’ha d’agrupar en branques de codi utilitzant <Table Row>.
El paràmetre **android:layout_column** asigna la columna a la que pertany cada view declarat. També es pot fer servir el paràmetre weight.

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp"
    android:stretchColumns="1">

    <TableRow android:background="#128675">

        <TextView
            android:layout_column="0"
            android:padding="3dip"
            android:text="Producto"
            android:textColor="@android:color/white" />

        <TextView
            android:layout_column="2"
            android:padding="3dip"
            android:text="Subtotal"
            android:textColor="@android:color/white" />
    </TableRow>

    <TableRow>

        <TextView
            android:layout_column="0"
            android:padding="3dip"
            android:text="Jabón de manos x 1" />

        <TextView
            android:layout_column="2"
            android:gravity="left"
            android:padding="3dip"
            android:text="$2" />
    </TableRow>
```

Grid Layout

Es distribueixen els elements de forma tabular, en files i columnes. A diferència del Table layout, aqui s’ha de declarar el número de files i de columnes com a paràmetre del layout, amb **android:rowCount** i **android:columnCount**. D’aquesta manera tots els elements es van ordenant un a un (sense declarar <Table Row>).



Action Bar

Per afegir una barra d'activitat, primer s'ha de comprovar si existeix la llibreria de compatibilitat.

```
class MyActivity : AppCompatActivity() {
  // ...
}
```

S'ha d'evitar que per defecte, el layout utilitzi la barra d'acció. Això s'ha de modificar en el Manifest i afegir la en el disseny de la activity.

```
<application
    android:theme="@style/Theme.AppCompat.Light.NoActionBar"
    />
```

```
<android.support.v7.widget.Toolbar
   android:id="@+id/my_toolbar"
   android:layout_width="match_parent"
   android:layout_height="?attr/actionBarSize"
   android:background="?attr/colorPrimary"
   android:elevation="4dp"
   android:theme="@style/ThemeOverlay.AppCompat.ActionBar"
   app:popupTheme="@style/ThemeOverlay.AppCompat.Light"/>
```

Estils y Temes

Estil: Es defineix com una col·lecció d'atributs d'aparença en una sola View (una activitat concreta).
Tema: És un estil aplicat a tota l'aplicació i a totes les seves vistes.

Per crear un nou estil o tema, s'ha d'obrir el fitxer res/values/styles.val del projecte, per a cada estil que desitgis, has de:

1. Agrega un element <style> amb un nombre que identifiqui l'estil exclusivament.
2. Agrega un element <item> per a cada atribut d'estil que vulguis definir.
   El "name" en cada element especifica un atribut que d'altra forma utilitzaries com un atribut XML, en el teu disseny. El valor de l'element <item> és el valor d'aquell atribut.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <style name="GreenText" parent="TextAppearance.AppCompat">
        <item name="android:textColor">#00FF00</item>
    </style>
</resources>
```

Es pot crear un tema de la mateixa manera en la que crees estils, la diferència és com ho apliques: enlloc d'aplicar l'estil amb l'atribut style en una vista, utilitzes **android:theme** en la etiqueta <application> o en una etiqueta <activity> dins del fitxer **AndroidManifest.xml**.

```
<manifest ... >
    <application android:theme="@style/Theme.AppCompat" ... >
    </application>
</manifest>
```

Intents

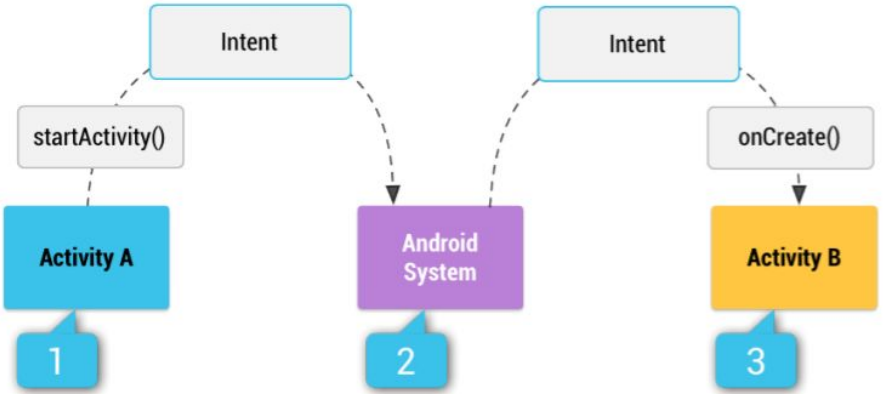
Un Intent és un objecte de missatgeria que pots utilitzar per a solicitar una acció d'un altre component de la app. Poden ser:

- **Intents explícites:** Especifiquen que l'aplicació administra (rebre) el intent. S'utilitzen normalment per iniciar activitys dins d'una mateixa app, pel simple fet que en el propi intent es pot determinar amb el nom de la classe o activity que crida.

```
// Executed in an Activity, so 'this' is the Context
// The fileUrl is a string URL, such as "http://www.example.com/image.png"
val downloadIntent = Intent(this, DownloadService::class.java).apply {
    data = Uri.parse(fileUrl)
}
startService(downloadIntent)
```

- **Intents implícites:** No nombren ningun component o clase en específic, així poden utilitzar-se (interpretar-se) per altres apps del sistema.

```
// Create the text message with a string
val sendIntent = Intent().apply {
    action = Intent.ACTION_SEND
    putExtra(Intent.EXTRA_TEXT, textMessage)
    type = "text/plain"
}

// Verify that the intent will resolve to an activity
if (sendIntent.resolveActivity(packageManager) != null) {
    startActivity(sendIntent)
}
```



Un objecte Intent té informació que el sistema Android utilitza per determinar quin component s'ha d'iniciar (com el nom exacte del component o la categoria que ha de rebre l'Intent), a més de la informació que el component receptor utilitza per a realitzar la acció correctament. Els camps a crear a la hora de definir un Intent són:

- **Nombre del component:** Component que s'ha de inicialitzar. És el que defineix un Intent com a explícit.
- **Acció:** String que defineix la acció genèrica a realitzar (View, send,etc.).
- **Dades:** Tipus de dades a tractar per l'aplicaió que genera i/o rep.
- **Categoria:** String amb informació sobre les dades a tractar.

Selector d'apps

Quan crees un intent implícit, poden existir vàries aplicacions que puguin executar el Intent, és per això que a vegades requerim dissenyar un selector que ens permeti escollir l'aplicació amb la que volguem interactuar.

Per mostrar el diàleg de selecció, s'ha de crear un Intent utilitzant **createChooser()** i transferir-la a **startActitivity()**, es mostra una llista de apps que responen a la Intent transferida al mètode **createChooser()**, amb el text proporcionat com a títol de diàleg.

```
val sendIntent = Intent(Intent.ACTION_SEND)
...

// Always use string resources for UI text.
// This says something like "Share this photo with"
val title: String = resources.getString(R.string.chooser_title)
// Create intent to show the chooser dialog
val chooser: Intent = Intent.createChooser(sendIntent, title)

// Verify the original intent will resolve to at least one activity
if (sendIntent.resolveActivity(packageManager) != null) {
    startActivity(chooser)
}
```

Filtres

En el procés de rebre un Intent, has de definir quins paràmetres ha de tindre per a poder-lo rebre en una activitat determinada. Per això es dissenyen els filtres, amb un element <intent-filter> declarat en el <manifest.xml>. Un component de l'aplicació ha de declarar filtres independents per a cada tasca única que pot fer, s'han d'espp ecificar aquests elements:

- **<action>**
- **<data>**
- **<category>**

```xml
<activity android:name="MainActivity">
    <!-- This activity is the main entry, should appear in app launcher -->
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>

<activity android:name="ShareActivity">
    <!-- This activity handles "SEND" actions with text data -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="text/plain"/>
    </intent-filter>
    <!-- This activity also handles "SEND" and "SEND_MULTIPLE" with media data -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <action android:name="android.intent.action.SEND_MULTIPLE"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="application/vnd.google.panorama360+jpg"/>
        <data android:mimeType="image/*"/>
        <data android:mimeType="video/*"/>
    </intent-filter>
</activity>
```