Interfaz de usuario:
Controles Básicos:

"Imágenes (ImageView)" y "Etiquetas (TextView)"

Mª Alejandra Tomás Fernández

Control ImageView (I)
(ver su API)

- Es un **subtipo de View**.
- Permite **mostrar imágenes** en la aplicación.
- **Propiedades más usadas**:
  - **android:src**
    - Permite **indicar la imagen a mostrar**.
    - **Suele indicarse** un **recurso** de nuestra carpeta /res/drawable.
    - *Por ejemplo* :

      android:src="@drawable/unaimagen".

Control ImageView (II)
(ver su API)

- **android:contentDescription**
  - Al ser un control de tipo **imagen**, **se aconseja usar esta propiedad**.
  - Ofrece una **breve descripción textual de la imagen** (***accesibilidad***).
- **android:maxWidth**
  - **Ancho máximo** que puede ocupar la imagen.
- **android:maxHeight**
  - **Alto máximo** que puede ocupar la imagen.

Control ImageView (III)
(ver su API)

- **Nota**: **android:maxWidth** y **android:maxHeight** se podrán utilizar siempre y cuando los valores de los atributos obligatorios **android:layout_width** y **android:layout_height** **no sean "match_parent".**

  Es recomendable **usarlos** cuando estos dos últimos atributos tienen el valor **"wrap_content"**

Control ImageView (IV)
(ver su API)

- **android:scaleType**
  - Indica cómo debe adaptarse la imagen al tamaño del control.
  - Los valores más usados son:

CENTER	Centra la imagen en la vista, pero **no realiza escalado**.
CENTER_CROP	Escala la imagen de manera uniforme (mantiene la relación de aspecto de la imagen), de modo que ambas **dimensiones** (anchura y altura) de la imagen serán **iguales o mayores que la dimensión correspondiente de la vista** (menos padding).
CENTER_INSIDE	Escala la imagen de manera uniforme (mantiene la relación de aspecto de la imagen), de modo que ambas **dimensiones** (anchura y altura) de la imagen **serán iguales o menores que la dimensión correspondiente de la vista** (menos padding).
FIT_CENTER	Escala la imagen **centrándola**.
FIT_END	Escala la imagen situándola **abajo a la derecha**
FIT_START	Escala la imagen situándola **arriba a la izquierda**.
FIT_XY	**Escala la imagen en ambos ejes X e Y**, lo cual puede **deformar** la imagen.

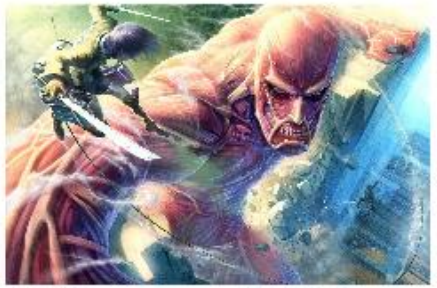
Control ImageView (V)
Establecer la imagen mediante XML

- Lo vemos con este **ejemplo**:

```
<ImageView android:id="@+id/ImgFoto"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@drawable/icon"
    android:contentDescription="@string/imagen_ejemplo" />
```

Control ImageView (VI)
Establecer la imagen mediante XML

android:scaleType



center	centerCrop	centerInside	fitCenter	fitEnd	fitStart	fitXY
						
圖片不縮放 圖片size超出View就被截斷) 居中顯示	圖片按比例縮放 圖片填滿整個View 居中顯示	圖片按比例縮放 圖片能完整地在View裡顯示 居中顯示	圖片按比例縮放 圖片寬等於View的寬度 居中顯示	圖片按比例縮放 圖片寬等於View的寬度 放在底部顯示	圖片按比例縮放 圖片寬等於View的寬度 放在頂部顯示	圖片不按比例縮放 圖片填滿整個View
						

Control ImageView (VII)
Establecer la imagen mediante código JAVA

- Si quisiéramos **establecer la imagen mediante código** utilizaríamos el método **setImageResource(…)**, pasándole el ID del recurso a utilizar como contenido de la imagen:

```
ImageView img= (ImageView)findViewById(R.id.ImgFoto);
img.setImageResource(R.drawable.icon);
```

Control ImageView (VIII)
Eventos de ImageView

- Al igual que los botones se puede hacer que responda al evento **onClick**.
- La única **diferencia con los botones** es que **ImageView** y **TextView** **no son "clickables" por defecto**. Por tanto, **no reaccionarán al hacer click.**
- Para que escuchen el evento "**onClick**" se debe **añadir** esta propiedad en el **XML**:

android:clickable="true"

- Ahora ya se pueden manejar como los botones.

Control TextView (I)
(ver su API)

- Es un **subtipo de View**.
- Son las **etiquetas de texto**.
- Se usa para **mostrar un texto al usuario**.
- El **texto a mostrar** se establece con el **atributo** ***(como en los botones)***:

android:text="*texto*"

Control TextView (II)
(ver su API)

- **Propiedades más usadas**:

  Suelen ser las que dan **formato** al **texto**.

  También existen en los botones vistos antes, y son:

  - **android:background**
    - Se usa para cambiar **el color de fondo o imagen de fondo**.
    - **Suele indicarse** un **recurso** de nuestra carpeta **/res/drawable**, o algún **color** que esté en **/res/values/styles.xml**.
    - ***Por ejemplo*** :

      android:background="@color/rojo"

Control TextView (II)
(ver su API)

- **android:textColor**
  - Se usa para cambiar **el color del texto.**
  - **Suele indicarse** un **recurso** color de nuestra carpeta **/res/values/styles.xml** o de **Android**.
  - ***Por ejemplo*** :

android:textColor=”**@color/azul**”.

android:textColor= ”**@android:color/holo_orange_dark** ”

Control TextView (III)
(ver su API)

- **android:textSize**
  - Se usa para cambiar **el tamaño de la fuente.**
  - Se suele usar una **medida concreta** en **sp** o una **dimensión** creada por nosotros o de Android, o un **estilo** nuestro o de Android.
  - ***Por ejemplo*** :

android:textSize=”16sp”.

android:textSize= "@android:style/TextAppearance.Large “

Control TextView (IV)
(ver su API)

- **android:typeface**
  - Se usa para cambiar **el tipo de fuente (*sans, monospace,…*).**
  - **Android sólo trae 3 tipos de fuente**, pero podemos añadir más siguiendo estos pasos:

    ¿cómo añadir nuevas fuentes a Android?
  - ***Por ejemplo*** :

    android:typeface=”sans”.

Control TextView (V)
(ver su API)

- **android:textStyle**
  - Se usa para cambiar **el estilo del texto (*negrita, cursiva,…*).**
  - ***Por ejemplo*** :

    android:textStyle=”**italic**”

- **android:shadowColor** → color de la sombra
- **android:shadowRadius** → radio de la sombra
- **android:shadowDx y android:shadowDy**
  - Se usan para añadir **sombra al texto**

Control TextView (VI)
(ver su API)

- ***Por ejemplo*** :

```
android:shadowColor="@color/verde"
android:shadowRadius="1.5"
android:shadowDx="1"
android:shadowDy="2"
```

Los **valores numéricos** del "radio" y desplazamiento respecto al eje X e Y **no van acompañados de ninguna unidad, son factores arbitrarios**.

Control TextView (VII)
(ver su API)

- **android:autoLink**
  - Controla si enlaces como urls, emails, teléfonos, mapas,… son encontrados en el TextView y los convierte en enlaces clickables.
  - ***Por ejemplo*** :

    android:autoLink="web|email"

    android:autoLink="all"

Control TextView (VIII)
Establecer sus propiedades mediante XML

- Lo vemos con este **ejemplo**:

```
<TextView android:id="@+id/LblEtiqueta"
    android:layout_width="matchfill_parent"
    android:layout_height="wrap_content"
    android:text="@string/escribe_algo"
    android:background="#AA44FF"
    android:typeface="monospace" />
```

Control TextView (IX)
Establecer sus propiedades mediante código JAVA

- Si quisiéramos **establecer las propiedades de un TextView mediante código** utilizaríamos diferentes métodos **get…(…) y set…(…).**
- **Ejemplo**:

```
final TextView lblEtiqueta = (TextView)findViewById(R.id.LblEtiqueta);
String texto = lblEtiqueta.getText().toString();
texto += "123";
lblEtiqueta.setText(texto);
lblEtiqueta.setBackgroundColor(Color.BLUE);
```