
Universidad
Tecmilenio®

Universidad Tecmilenio

Desarrollo de aplicaciones en plataforma Android

Elementos de interfaz gráfica

Introducción

Android proporciona una gran variedad de vistas básicas para poder diseñar interfaces de usuario de manera sencilla.

Explicación

Usando las vistas es posible desarrollar aplicaciones consistentes visualmente con las del resto del sistema, pero también es posible modificarlas o ampliar sus funcionalidades.

Solo para conocerlas, entre las vistas más comunes están:

- TextView: una etiqueta de sólo lectura. Permite texto multilínea, formateado de cadenas, etc.
- EditText: una caja de texto editable. Permite texto multilínea, texto flotante, etc.
- ListView: un grupo de vistas que crea y administra una lista vertical de vistas, correspondiéndose cada una de ellas a una fila de la lista.

Explicación

- Spinner: un cuadro de selección que permite seleccionar un elemento de una lista de posibles opciones. Se muestra como un botón que al ser presionado muestra la lista de posibles opciones.
- Button: un botón normal.
- CheckBox: un botón con dos posibles estados representado por un recuadro que puede estar marcado o no.
- RadioButton: un grupo de botones con dos posibles estados. Uno de estos grupos muestra al usuario un conjunto de opciones de las cuales sólo una puede estar activa simultáneamente.
- SeekBar: una barra de desplazamiento, que permite escoger visualmente un valor dentro de un rango entre un valor inicial y final.

Explicación

TextView corresponde a una etiqueta de texto simple, su principal función es mostrar un texto al usuario.

Su atributo más importante es android:text, cuyo valor indica el texto
a mostrar por pantalla. También pueden ser de interés los atributos android:textColor y android:textSize.

Una curiosidad es que sólo se podrá editar texto a través de su subclase EditText.

Explicación

EditText no es más que una subclase de TextView que está preparada para la edición de texto.

Al presionar sobre la vista la interfaz de Android mostrará un teclado que permite introducir datos del usuario. En el código se maneja igual que un TextView (por medio de los métodos getText o setText, principalmente).

Explicación

En cuanto a los botones:

Button: representa un botón normal y común, con un texto asociado.

- Para cambiar el texto del botón usa su atributo android:text.
- La forma más habitual de interactuar con un botón es pulsarlo para desencadenar un evento.
- Para que una aplicación realice una determinada acción cuando el botón sea pulsado, se debe implementar un manejador para el evento OnClick.

CheckBox: Es un botón que puede encontrarse en dos posibles estados: seleccionado o no seleccionado. Este tipo de botones, al contrario que en el caso de los pertenecientes a la vista RadioButton, son totalmente independientes unos de otros, por lo que varios de ellos pueden encontrarse seleccionados al mismo tiempo.

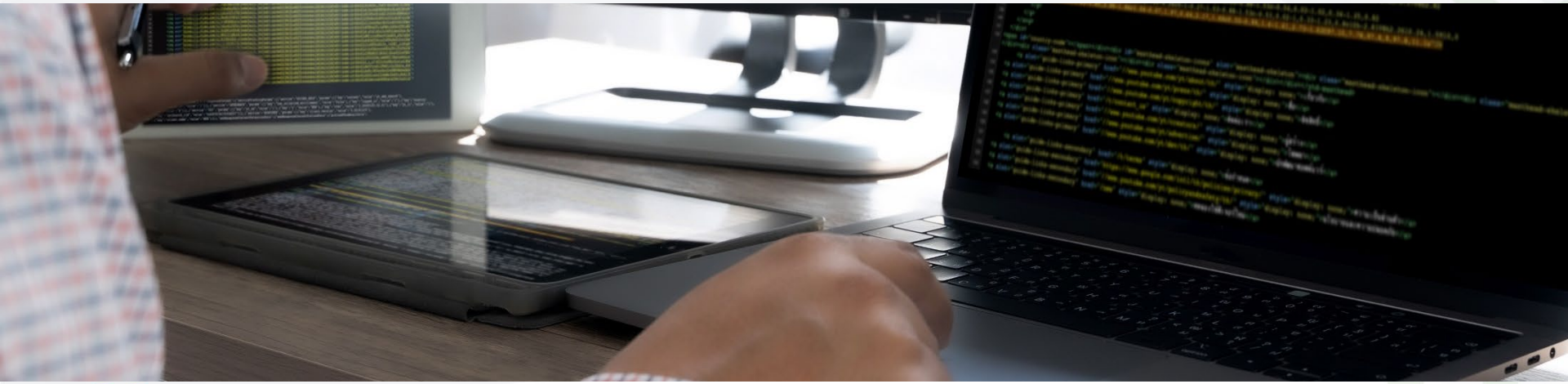
- Los métodos más importantes son isChecked, que devuelve un booleano si el botón está seleccionado, y setChecked, que recibe un valor booleano para indicar el estado del botón.

Explicación

RadioButton. Un RadioButton, al igual que un CheckButton, tiene dos estados (seleccionado y no seleccionado). La diferencia con el anterior es que una vez que el botón está en el estado de seleccionado no puede cambiar de estado por intervención directa del usuario.

Los elementos de tipo RadioButton suelen agruparse normalmente en un elemento de tipo RadioGroup, de tal forma que la activación de uno de los botones del grupo supondrá la desactivación de los demás.

Spinner. Un Spinner es una vista que permite escoger uno de entre una lista de elementos. Este se analizará en temas posteriores.



Actividad

- ¿Qué opciones tienes para modificar la información a través de las etiquetas TextView y los EditText?
- Dentro del diseño y la programación de la aplicación ¿Cómo es posible combinar los textos con los botones para la interfaz de usuario?
- ¿Cuáles son las principales clases que se involucran en el diseño de las etiquetas?

Cierre

El uso de los elementos de interfaz gráfica es algo sencillo y fácil de practicar, solo se requiere tener en cuenta qué es lo quela aplicación requiere, y practicar una y otra vez.

Visualiza diferentes aplicaciones y examina qué elementos se utilizaron, te ayudará a reforzar lo visto en este tema y estar más preparado; ya que el éxito de tu interfaz gráfica será por un lado tu imaginación.

Universidad Tecmilenio

Desarrollo de aplicaciones en plataforma Android

Más sobre interfaz gráfica.

Introducción

El *widget* **Spinner** de Android muestra una lista desplegable para seleccionar un único elemento.

Es equivalente al típico select de html o los ComboBox de otros entornos de desarrollo.

Explicación

El control Spinner muestra una lista de String y nos permite seleccionar uno de ellos. Cuando se lo selecciona se abre y muestra todos sus elementos para permitir seleccionar uno de ellos.

Al igual que un ListView, el contenido de la lista desplegable con los elementos del Spinner proviene de un Adapter (en el ejemplo de la documentación es un array definido en un fichero xml).

Explicación

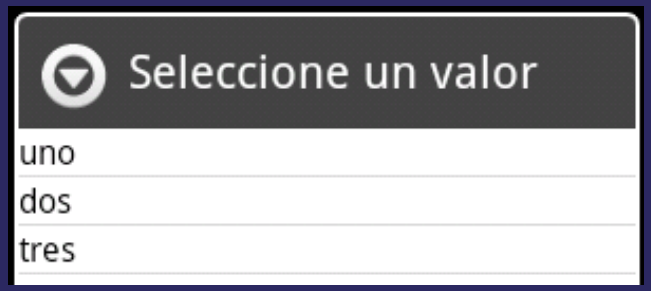
Al igual que un ListView, el contenido de la lista desplegable con los elementos del Spinner proviene de un Adapter (puede ser a través de un array definido en el archivo xml).

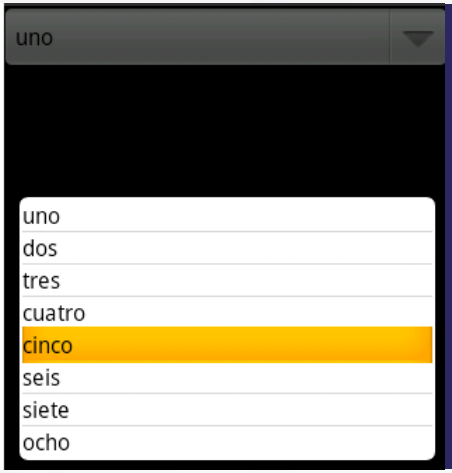
Ejemplo básico de Spinner con su layout y la Activity asociada la cual muestra el Toast cuando se selecciona un elemento.

```
<Spinner
    android:id="@+id/spinner"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content" />
```

Explicación

En Android, la lista podría mostrarse de diferentes maneras, pero mucho depende de las características del emulador, algunas vistas de ejemplo pueden ser:





Afortunadamente es posible personalizar los Spinner para que su estilo se adapte a nuestra app y para darle un toque personalizado.

Explicación

Es posible personalizar el elemento de selección que le aparece al usuario, esto a través de las imágenes en el folder drawable, sin embargo, también es posible personalizar su contenido, a través de cear un layout con simplemente un TextView con las características que se requieren y utilizarlo con el ArrayAdapter del Spinner.

Ejemplo:

```
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@android:id/text1"
    android:textColor="#de0000"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"/>
```

Y el ejemplo en el adaptador es:

```
spinner.setAdapter(new ArrayAdapter<String>(this, R.layout.textview_spinner, valores));
```

Explicación

Para visualizarse de la siguiente manera, sin depender de las características de un emulador:

Actividad

- ¿Qué diferencias encuentras entre un spinner básico y un que se puede personalizar?
- ¿Cuál sería un ejemplo para una aplicación que emplee un spinner básico y uno personalizado?
- ¿Cuál es la función del adaptador en el Spinner?

Cierre

Como ves, ya estas obteniendo más herramientas para manejar la interfaz gráfica, conforme la comprendas y practiques te será más dinámica emplearla y personalizarla.

Es importante tener en cuenta que los elementos se definen en la vista de Android pero también se controlan con la programación aún y cuando no se hayan definido en el archivo XML, todo depende de que lo que se requiera hacer.

Universidad Tecmilenio

Desarrollo de aplicaciones en plataforma Android

Transiciones

Introducción

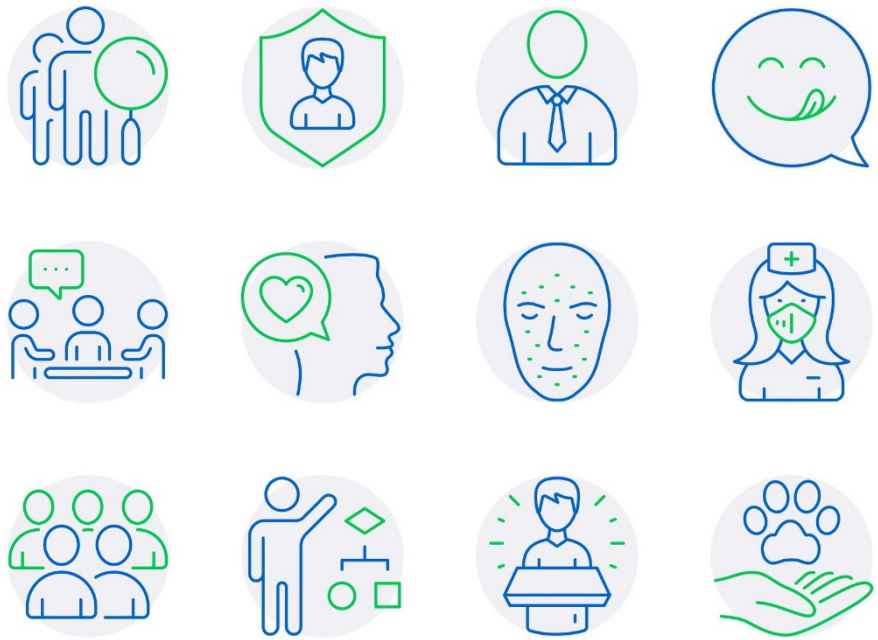
Las animaciones de las pantallas de una aplicación, además de aportar una cierta sofisticación, permiten hacer evidente al usuario los cambios que se producen en la pantalla.

Quizá navegar en una aplicación con una vista que lleve a otra no resulte tan novedoso, pero cuando se incluye movimiento o animación, ya se vuelve más llamativo visualmente para los usuarios.

Explicación

Las transiciones de actividades de las aplicaciones son las encargadas de proporcionar las conexiones visuales entre diferentes estados a través del movimiento y las transformaciones entre elementos comunes.

Toda transición que extiende la clase Visibility se admite como una transición de entrada o salida.



Explicación

Es importante para habilitar las transiciones del contenido de la ventana en código, se debe llamar a la función Window.requestFeature():

```
// inside your activity (if you did not enable transitions in your theme)
getWindow().requestFeature(Window.FEATURE_CONTENT_TRANSITIONS);
```

Para especificar transiciones en código, se llama a las siguientes funciones con un objeto Transition:

Window.setEnterTransition()
Window.setExitTransition()
Window.setSharedElementEnterTransition()
Window.setSharedElementExitTransition()

Explicación

Las funciones **setExitTransition()** y **setSharedElementExitTransition()** permiten definir la transición de salida para la actividad de llamada.

Las **funciones setEnterTransition()** y **setSharedElementEnterTransition()** definen la transición de entrada para la actividad llamada.

Para obtener el efecto completo de una transición, es necesario habilitar las transiciones del contenido de las ventanas tanto para las actividades que realizan la llamada como para aquellas que son invocadas.

Explicación

Para iniciar una actividad mediante transiciones, si se habilita una transición y esta es una de salida para la actividad, la transición entonces se activa al iniciar otra actividad a través de:

```
startActivity(intent,
              ActivityOptions.makeSceneTransitionAnimation(this).toBundle());
```

Actividad

- ¿Cuál es el propósito de las transiciones?
- ¿Cómo se conecta el contenido entre transiciones?
- ¿Cuál es la estructura principal de una transición?

Cierre

Es importante practicar el poder hacer transiciones entre escenas, para que conforme se practica sea más fácil incorporarlas a una aplicación y hacer así una experiencia inolvidable para el usuario, quizá integrando imágenes o difuminar o incluir animaciones.

Universidad Tecmilenio

Desarrollo de aplicaciones en plataforma Android

Uso de aplicaciones externas

Introducción

En el sistema operativo Android, una aplicación puede utilizar otra aplicación que ya está definida previamente e incluso haber interacción entre ellas, en una forma muy sencilla para realizar aplicaciones por ejemplo para tomar una foto o hacer una llamada.

Así que ahora analizaremos cómo utilizar otras aplicaciones que tienen relevancia para la aplicación que se está desarrollando.

Explicación

Un **Intent** es un objeto de mensajería que se puede usar para solicitar una acción de otro componente de una aplicación.

Existen tres casos de uso principales:

Iniciar una actividad:

- Una Activity representa una única pantalla en una aplicación. Es posible iniciar una nueva instancia de una Activity pasando una Intent a **startActivity(intent).**
- El Intent describe la actividad que se debe iniciar y contiene los datos necesarios para ello.

Explicación

Iniciar un servicio:

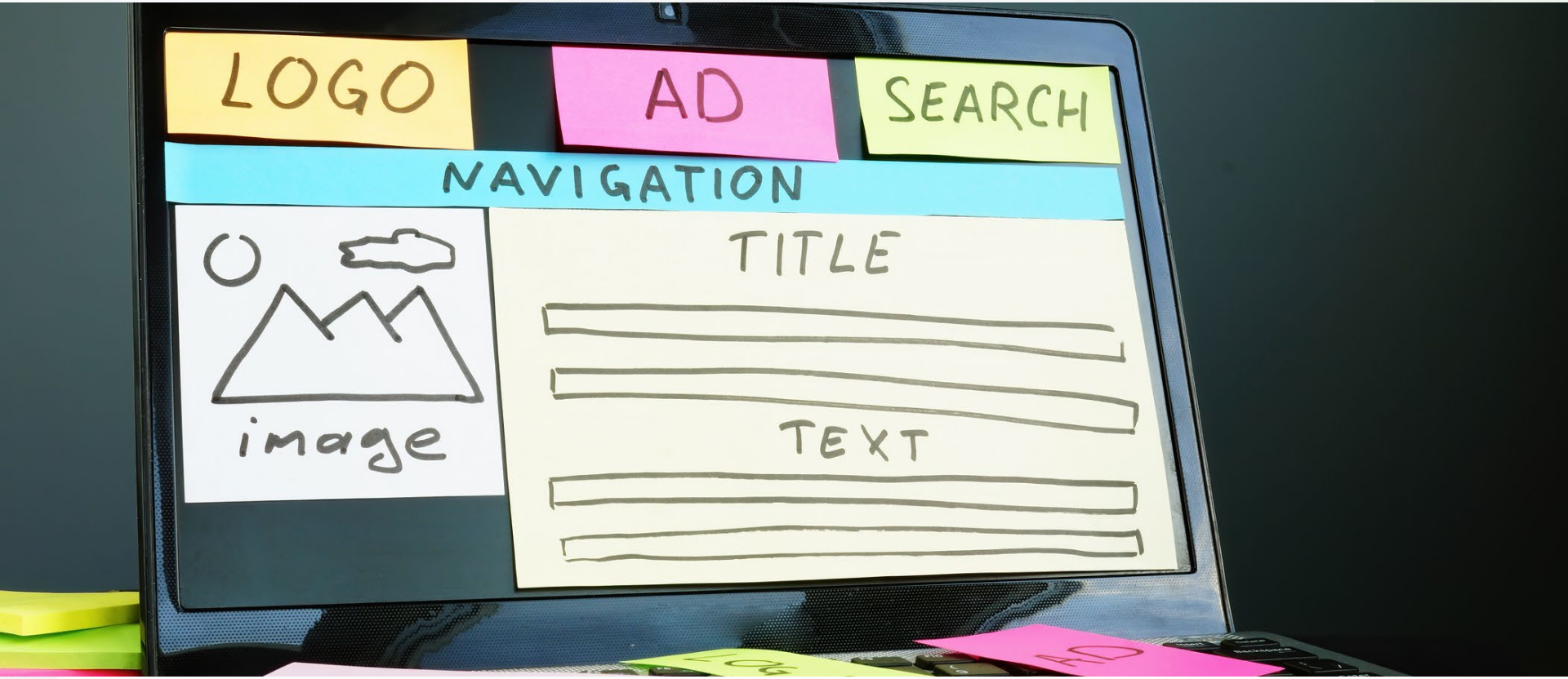
- Un **Service** es un componente que realiza operaciones en segundo plano sin una interfaz de usuario.

Transmitir una emisión:

- Una emisión es un aviso que cualquier aplicación puede recibir. El sistema transmite varias emisiones de eventos, como cuando se inicia el sistema o comienza a cargarse el dispositivo.
- Es posible transmitir una emisión a otras apps pasando el Intent através de sendBroadcast() o sendOrderedBroadcast().

Explicación

Intent, como objeto tiene información que el sistema Android usa para determinar qué componente debe iniciar (como la categoría que debe recibir intent), además de tiene la información que el componente receptor usa para realizar dicha acción de manera correcta.



Explicación

A través de Intent se puede especificar de manera explícita el nombre correspondiente a la clase de la actividad que se desea iniciar, por ejemplo:

Crear un nuevo Intent al que se le pase como parámetro el contexto de la aplicación y la clase de la actividad que se va a ejecutar:

```
Intent intent = new Intent(MiActividad.this, MiOtraActividad.class);
startActivity(intent);
```

Explicación

Otra forma de iniciar una actividad es a través de un Intent implícito, donde se solicita un componente anónimo de una aplicación sin determinar se encargue de satisfacer una petición concreta.

Por ejemplo, para que se puedan hacer llamadas de teléfono desde nuestra aplicación se requiere implementar una nueva actividad para ello, o utilizar un Intent implícito que solicite que se lleve a cabo la tarea de realizar la llamada a un número de teléfono representado por una URI:

```
Intent intent = new Intent(Intent.ACTION_DIAL, Uri.parse("tel:666666666");
startActivity(intent);
```

En el caso en el que existan varias actividades igualmente capaces de llevar a cabo la tarea se le permitirá escoger al usuario cuál de ellas utilizar por medio de una ventana de diálogo.

Actividad

- ¿Para qué se emplean los Intent en Android?
- ¿Entonces, cuando una actividad A lanza una actividad B y que pertenece a una aplicación distinta, cuál actividad reutiliza a cuál?
- ¿Cuál es la función de URI dentro de los Intent?

Cierre

Es muy común querer acceder a otras aplicaciones a través de los dispositivos móviles, por ejemplo hacer una llamada o hacer uso del mapa, para brindar estas aplicaciones y que sean satisfactorias para el usuario, como programador necesitarás aplicar los Intent.