Menús, listas y barras de progreso - Ejercicios

Índice

1. Barra de progreso lineal

Crea una aplicación llamada *BarraProgreso* conteniendo una única actividad llamada *Principal*, cuya interfaz gráfica sea la siguiente:

Interfaz de la aplicación BarraProgreso

Como puedes observar, la interfaz se compone de dos cuadros de edición de texto, un botón y una barra de progreso lineal. Los valores de los dos cuadros de edición marcan un valor inicial y un valor final para el proceso que se lanzará al pulsar el botón. Al pulsar el botón, una propiedad de la actividad llamada `cuenta` deberá ir avanzando desde el valor inicial hasta el valor final, haciendo una pausa de un segundo tras cada incremento.

Mientras se realice este progreso tanto el botón como los cuadros de edición de texto deberán estar deshabilitados, y la barra de progreso deberá ir mostrando la evolución del mismo, avanzando tras cada incremento. Cuando finalice el proceso, todas las vistas deberán volver a estar habilitadas.

Por último, antes de comenzar el progreso, hemos de asegurarnos de que el valor inicial sea menor que el valor final. En caso contrario el proceso no se lanzará y se deberá mostrar un `Toast` con un mensaje de error.

Nota:

Para habilitar o deshabilitar una vista invocamos a su método `setEnabled` pasándole como parámetro `true` o `false`, respectivamente.

Nota:

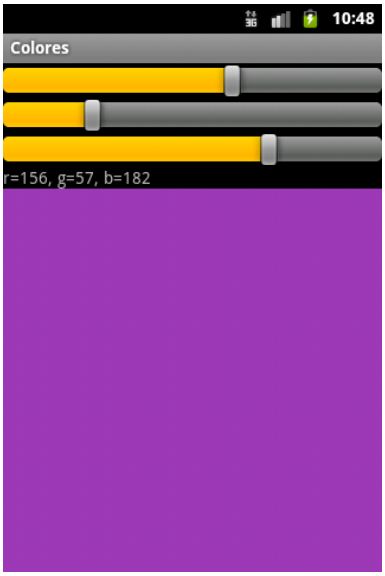
El valor máximo de una barra de progreso se puede establecer desde el código con el método `setMax` de `ProgressBar`.

Aviso:

Si intentamos modificar algún atributo de alguna de las vistas desde el método `run` del elemento `Runnable` se producirá una excepción. Sólo podremos modificar la interfaz gráfica desde el `Handler`.

2. Selección de color

En este ejercicio vamos a crear una aplicación basada en vistas de tipo `SeekBar` llamada *Colores*. La aplicación permitirá visualizar colores para diferentes valores de las tres componentes básicas rojo (r), verde (v) y azul (a) de manera dinámica. El aspecto de la interfaz será el siguiente:



Interfaz de la aplicación Colores

Cada uno de los tres `SeekBar` permitirá modificar el valor de cada una de las componentes básicas. El valor de estos `SeekBar` será como mínimo de 0 y como máximo de 255. Debajo de ellos se mostrará mediante un `TextView` el valor de cada componente de color. Finalmente, tendremos un último `TextView` sin texto asociado y cuya propiedad `android:layout_height` valdrá `fill_parent`. El color de fondo de éste se obtendrá a partir de la mezcla de la proporción indicada de los tres colores básicos.

Nota:

El color de fondo de un elemento `TextView` puede ser modificado con su método `setBackground`, que debe recibir como parámetro un color. Para indicar el color utilizamos la

clase `Color`, y más concretamente el método estático `Color.rgb()`, que recibe tres parámetros: el valor de rojo, verde y azul.

3. Lista de tareas

El objetivo de este ejercicio será crear una sencilla aplicación llamada *ListaTareas* para gestionar una lista de tareas. La aplicación tendrá una única actividad, que heredará de `ListActivity`, y en cuya interfaz tan sólo habrá un cuadro de edición de texto, un botón y una vista de tipo `ListView`. Cada vez que pulsemos el botón, el texto introducido en el cuadro de edición se añadirá a la lista como una nueva lista de tareas y el contenido de dicho cuadro de edición se borrará para dejar paso a la nueva tarea. Tienes una figura mostrando un ejemplo de la aplicación en ejecución al final del enunciado del ejercicio.

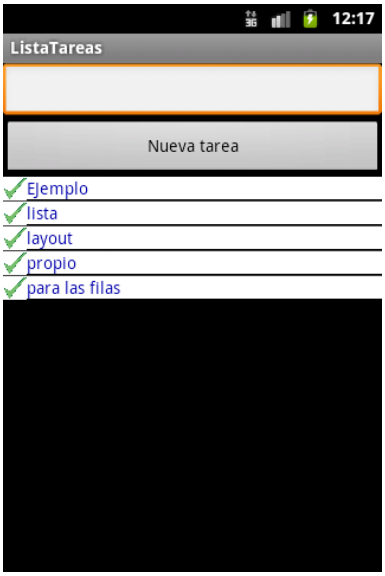
Para completar el ejercicio debes seguir los sigiuentes pasos:

- Crea la aplicación *ListaTareas*, con una actividad llamada *Principal* que debe heredar de `ListActivity`.
- Modifica el layout de la actividad para que incluya un `EditText` y un `Button` en el borde superior, además de un componente `ListView`. Recuerda para que dicho `ListView` pueda ser manejado por la actividad, su identificador debe ser `android:id="@android:id/list"`.
- Añade a la clase *Principal* un `ArrayList` de cadenas, que servirá para almacenar las tareas que el usuario vaya introduciendo. Inicializalo en el método `onCreate` con el operador `new`, pero de momento no introduzcas ninguna cadena.
- Añade a la clase principal un objeto de tipo `ArrayAdapter<String>`. Inicialízalo y asocialo a la lista de la actividad. Al inicializar el adaptador, recuerda que el primer parámetro debe ser el contexto de la aplicación (puedes usar `this`), para el segundo utilizaremos el identificador del layout por defecto para las filas `android.R.layout.simple_list_item_1`, y el tercero será el `ararayList` que creaste en el punto anterior.
- Añade un manejador al botón. Cada vez que se pulse el botón se debe incorporar el texto del `EditText` al `ArrayList` de cadenas y luego limpiar el `EditText`.
- Por último, en ese mismo manejador, invoca al método `notifyDataSetChanged` del adaptador para que la lista se actualice cada vez que se introduzca un elemento en el `ArrayList`

Interfaz de la aplicación ListaTareas

4. Modificando el aspecto de la lista de tareas (*)

Crea un layout personalizado para las filas de la lista de la aplicación *ListaTareas*. Cada elemento de la lista debe mostrarse con texto azul oscuro sobre fondo blanco. A la izquierda de cada elemento de la lista debe aparecer la imagen *check.png* que se te proporciona en las plantillas de la sesión. Tanto la anchura como la altura de la imagen será de 20dip.



Interfaz de la aplicación ListaTareas con un layout propio para las filas

5. Menú contextual (*)

Basándonos también en el ejercicio de la lista de tareas, el siguiente paso que vamos a seguir es añadir un menú contextual. Tras realizar una pulsación larga sobre alguno de los elementos de la lista de tareas se debe mostrar un menú contextual con dos posibles operaciones: eliminar el elemento de la lista de tareas o duplicarlo, colocando la nueva copia de la tarea al final.

Para que todo funcione correctamente debes asociar el menú contextual al `ListView` y desde allí acceder al elemento seleccionado, en lugar de intentar asociar un menú contextual a cada uno de los elementos individuales de la lista.

El menú contextual de la lista de tareas

Nota:

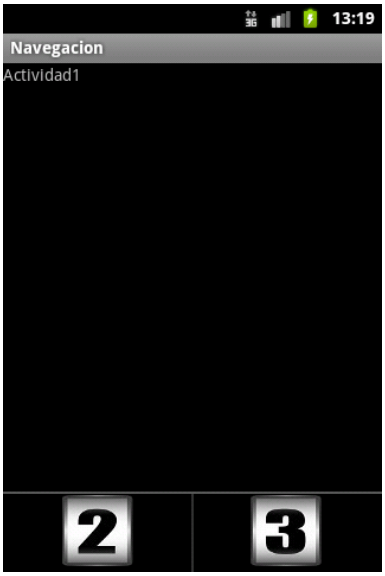
Para saber dentro de `onContextItemSelected` para qué elemento de la lista se mostró el menú contextual, debemos introducir en dicha función la línea `AdapterView.AdapterContextMenuInfo info = (AdapterView.AdapterContextMenuInfo)item.getMenuInfo();`. Una vez hecho esto, podremos acceder al entero `info.position` para obtener esta información.

6. Lanzando actividades desde un menú

En este ejercicio vamos a crear una aplicación llamada *Navegacion* para navegar entre tres actividades. Las tres actividades tendrán de nombre *Actividad1*, *Actividad2* y *Actividad3*. La interfaz gráfica de cada una de ellas consistirá únicamente en un `TextView`

mostrnado su nombre. Desde cada una de ellas se podrá acceder a las otras dos mediante un menú de iconos. Esto quiere decir que el menú de cada actividad será distinto y se compondrá de dos únicas opciones.

En las plantillas de la sesión se incluyen tres archivos gráficos llamados *icono1.png*, *icono2.png* y *icono3.png* que deberemos utilizar en los menús.



Interfaz de la aplicación Navegacion

Nota:

Asigna al atributo `android:noHistory` de cada actividad en el *Manifest* de la aplicación el valor `true`.