NOTIFICACIONES

Pág. 245 y sgts. del Manual de SGOliver

1. INTRODUCCION

- En ocasiones es necesario mostrar al usuario pequeños mensajes de alerta o de aviso, para los cuales no interesa configurar una pantalla completa.
- Android dispone de diferentes métodos para ello:
  - Los avisos basados en la clase **Toast**.
  - Las **ventanas de diálogo**.
  - Las **notificaciones** en la barra de estado o barra de tareas.

2. TOAST

- Aunque aparecen por defecto en la parte inferior de la pantalla, son personalizables.
- Su uso defectivo se basa en el código que ya hemos utilizado en multitud de ocasiones:

```
Toast t1 = Toast.makeText(getApplicationContext(), "Toast por defecto", Toast.LENGTH_SHORT);
t1.show();
// lo que abreviamos mediante
//Toast.makeText(getApplicationContext(), "Toast por defecto", Toast.LENGTH_SHORT).show();
```

3. VENTANAS DE DIALOGO

Documentacion en http://developer.android.com/guide/topics/ui/dialogs.html

- Un diálogo es una pequeña ventana que aparece delante de la actividad en curso. La actividad que está detrás pierde el foco y el diálogo es el que recibe todas las interacciones con el usuario.
- Se suelen utilizar para que el usuario tome una decisión o reciba un mensaje informativo.
- No cubren toda la pantalla, sino que flotan sobre la pantalla de fondo, que queda inactiva.

- Android define varios tipos diferentes de ventanas de diálogo:
  - **AlertDialog**: muestra entre cero y tres botones y/o una lista de items seleccionables que pueden incluir casillas de verificación (checkbox) o botones de opción (radio buttons).
  - **ProgressDialog**: muestra una barra o rueda de progreso. También se pueden incluir botones.
  - **DatePickerDialog**: este diálogo permite seleccionar una fecha.
  - **TimePickerDialog**: este diálogo permite seleccionar una hora.

- La utilización de las ventanas de diálogo puede basarse en el concepto de "**fragmentos**" (a partir de la API 11 – Android 3.0) pero nosotros lo veremos sin usar esto.
- Una ventana de diálogo siempre se crea y se muestra como **parte de una Activity**.
- En los siguientes apartados vamos a probar algunos tipos de ventanas de diálogo.

4. VENTANA CON MENSAJE

- Es una ventana de diálogo que obliga a que el usuario vea un mensaje ya que bloquea la pantalla hasta que se pulse la tecla "Atrás" o se haga click fuera de la misma. Por ejemplo:

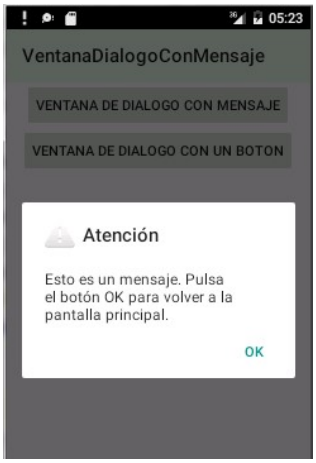
- El proceso a seguir para crear una ventana de diálogo como la anterior es:
  - Crear un objeto ***AlertDialog*** a través de la clase ***Builder***, para la ventana de diálogo.
  - Configurar el título, el mensaje y el icono de la ventana, con los métodos ***setTitle()***, ***setMessage()*** y ***setIcon()***.
  - Por último, pedir que se muestre la ventana mediante el método ***show()***.

```
AlertDialog.Builder ventana = new AlertDialog.Builder(this);
ventana.setTitle("Atención");
ventana.setMessage("Esto es un mensaje. Pulsa el botón...");
ventana.setIcon(android.R.drawable.ic_dialog_alert);
ventana.show();
```

- Podemos probar esto con el **ejercicio 2**.

5. VENTANA CON BOTON/ES

- Puede haber hasta tres botones.
- Ventana de diálogo con **un botón**:



- Igual que en el caso anterior, creamos un objeto ***AlertDialog.Builder***, y configuramos su icono, título y mensaje. Pero, antes de pedir que se visualice la ventana, debemos implementar la existencia del botón, así como deshabilitar la posibilidad de que la ventana se cierre por otros medios:

```
// Inhabilitamos la posibilidad de que el usuario cierre la ventana sin pulsar el botón
ventana.setCancelable(false);
// Método que indica el texto del botón y la clase anónima que capturará su evento onClick
ventana.setPositiveButton("Ok", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {
        dialog.cancel();
    }
});
```

- Lo principal en este tipo de ventanas es la implementación del evento ***onClick*** en forma de objeto **OnClickListener**. Dentro de este evento, hemos cerrado la ventana de diálogo mediante su método ***cancel()***, aunque podríamos haber realizado cualquier otra acción.
- Podemos probar esto con el **ejercicio 3**.
- De forma similar procederíamos si la ventana de diálogo tuviese dos o tres botones, empleando los métodos ***setNegativeButton()*** y ***setNeutralButton()***, respectivamente.

- Podemos probar esto en el **ejercicio 4**.

6. METODOS ONCREATEDIALOG() Y SHOWDIALOG()

- Aunque hasta ahora, por simplificar, hemos puesto todo el código en el método onCreate() (o bien en otros métodos creados por nosotros), **siempre se deberían crear los diálogos dentro del método *onCreateDialog()*** de la Activity a la que están asociados.

- Cuando se quiere mostrar un diálogo, hay que llamar al método ***showDialog(int)*** y pasarle un integer que identifica unívocamente al diálogo que se quiere mostrar.

```
// Pedimos que se muestre el dialogo asociado a la constante 1
showDialog(DIALOGO_MENSAJE);
```

- Para ello, debemos definir una constante entera para cada ventana de diálogo que queramos crear.

```
// constantes enteras para identificar cada ventana de diálogo
private static final int DIALOGO_MENSAJE = 1;
(…)
```

- Cuando se pide un diálogo por primera vez, Android llama al método ***onCreateDialog(int)*** de la Activity, que es donde se debe instanciar el diálogo. A este método se le pasa el mismo ID que se le había pasado a ***showDialog(int)***. Después de crear el Dialog, se devuelve el objeto al final del método.

```
@Override
protected Dialog onCreateDialog (int id){
    AlertDialog.Builder ventana = new AlertDialog.Builder(this);
    ventana.setTitle("Atención");
    ventana.setMessage("Esto es un mensaje. Pulsa ...");
    ventana.setIcon(android.R.drawable.ic_dialog_alert);
    return ventana.create();
}
```

- El método ***onCreateDialog(int)*** sólo es llamado la primera vez que se demanda el diálogo, después ya queda en la memoria residente y sólo es necesario realizar una llamada con ***showDialog(int)*** para que se muestre en pantalla.

- Podemos probar esto en el **ejercicio 5**.

7. VENTANAS DE DIALOGO CON ELEMENTOS DE SELECCION

- Cuando las opciones a seleccionar por el usuario son más de tres podemos utilizar los diálogos de selección para mostrar una lista de opciones entre las que el usuario pueda elegir.

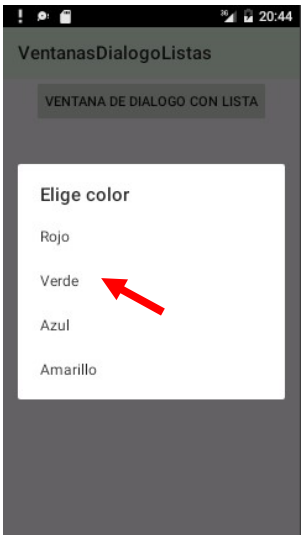
- También utilizaremos la clase ***AlertDialog***, pero sin asignar ningún mensaje ni botón, sino que directamente indicaremos la lista de opciones a mostrar, mediante el método ***setItems().***

- La lista de opciones puede ser un array tradicional.

- Mediante un listener de tipo ***DialogInterface.OnClickListener*** podemos implementar el evento ***onClick()*** correspondiente a la lista de opciones, para poder saber la opción seleccionada.

- Debido a que la lista aparece en la zona de ventana destinada al mensaje, la ventana de diálogo no puede mostrar al mismo tiempo un mensaje y una lista.

- Ejemplo:

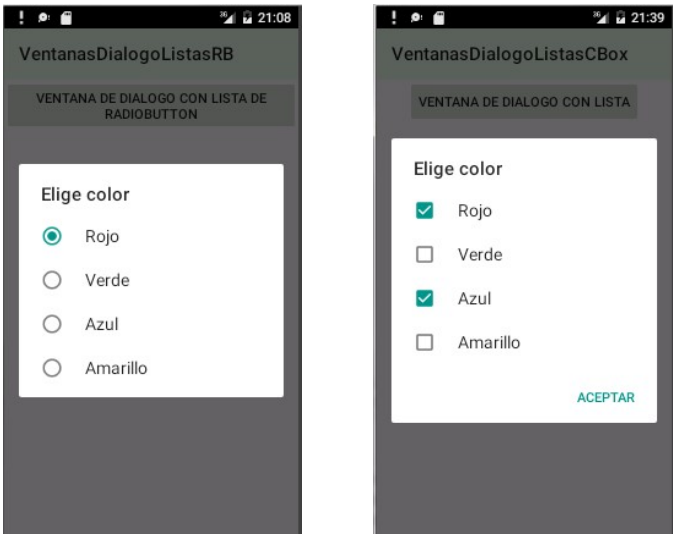


- Código:

```
@Override
public Dialog onCreateDialog(int id) {
    final String[] colores = {"Rojo", "Verde", "Azul", "Amarillo"};
    AlertDialog.Builder ventana = new AlertDialog.Builder(this);
    ventana.setTitle("Elige color");
    ventana.setItems(colores, new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            // el parametro "which" contiene la posicion del elemento seleccionado
            Toast.makeText(getApplicationContext(), "Opción elegida: " + colores[which],
                 Toast.LENGTH_LONG).show();
        }
    });
    return ventana.create();
}
```

- Podemos probar esto en el **ejercicio 6**.

- También es posible crear ventanas de diálogo con listas de selección múltiple o listas con botones de radio:



- Para ello, hay que utilizar los métodos ***setMultiChoiceItems()*** (selección múltiple) y ***setSingleChoiceItems()*** (selección simple con botones de radio).

- El método ***setSingleChoiceItems()*** es similar a ***setItems()***, pero recibe como segundo parámetro el índice de la opción marcada por defecto (o el valor -1, si no queremos tener ninguna de ellas marcada inicialmente).

```
ventana.setSingleChoiceItems(colores, 0, new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {
        (…)
    }
});
```

- El método ***setMultiChoiceItems()*** implementa un listener del tipo **OnMultiChoiceClickListener**. Y puede recibir como segundo parámetro el valor **null** para indicar que no debe aparecer ninguna opción seleccionada por defecto. Además, en este caso, el evento onClick recibe tanto la opción seleccionada (tipo int) como el estado en el que ha quedado dicha opción (tipo boolean).

```
ventana.setMultiChoiceItems(colores, null, new
DialogInterface.OnMultiChoiceClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which, boolean isChecked) {
        (…)
    }
});
```

- Podemos probar esto en el **ejercicio 7 y 8**
  (Ayuda en http://developer.android.com/guide/topics/ui/dialogs.html).