Menús y barras de herramientas (1)

- **Menús**

Los menús han caído en desuso desde la aparición de *ActionBar*. De hecho, si utilizamos un menú en una API 11 o superior (Android 3.0) veremos que su ubicación ha dejado de ser la parte inferior de la pantalla para pasar al desplegable de la esquina superior derecha del *ActionBar*.

154

Menús y barras de herramientas (2)

- **Menús**

Existen tres tipos de menú:

- Menú principal: menú general asociado a una actividad. En las versiones de Android inferiores a 3.0 aparece abajo y en las superiores en la esquina superior derecha.
- Submenú: se despliega al pulsar sobre una opción de un menú principal.
- Menú contextual: aparecen al realizar una pulsación larga sobre algún elemento.

155

Menús y barras de herramientas (3)

- **Menú Principal / Submenú**

- Mediante archivo XML:

Bastará con crear un archivo XML dentro de la carpeta de recursos "menu" en el que podremos utilizar dos tipos de elementos:

- item: cada uno de los elementos del menú o submenú.
- menu: agrupación de elementos (menú principal o submenú).

156

Menús y barras de herramientas (4)

- **Menú Principal / Submenú**

- Mediante archivo XML:

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
  <item android:id="@+id/menuOp1"
        android:title="Opcion1" />
  <item android:id="@+id/menuOp2"
        android:title="Opcion2" />
  <item android:id="@+id/menuOp3"
        android:title="Opcion3" >
       <menu>
          <item android:id="@+id/subMenu3Op1"
                android:title="Opcion 3.1" />
          <item android:id="@+id/subMenu3Op2"
                android:title="Opcion 3.2" />
       </menu>
  </item>
</menu>
```

menu_main.xml

157

Menús y barras de herramientas (5)

- **Menú Principal / Submenú**

- Mediante archivo XML:

Una vez definido el menú, implementaremos el método *onCreateOptionsMenu* en la actividad en la que queramos que se muestre, en nuestro ejemplo "*MainActivity*".

```
public class MainActivity extends AppCompatActivity {
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
    }
    /* ... */
}
```

158

Menús y barras de herramientas (6)

- **Menú Principal / Submenú**

- Mediante código:

Dentro del método *onCreateOptionsMenu* añadiremos las opciones de menú mediante *add*, y los submenús mediante *addSubMenu*.

Ambos métodos reciben cuatro parámetros:

- ID del grupo asociado a la opción.
- ID único para la opción (suelen utilizarse ctes).
- Orden de la opción.
- Texto de la opción.

159

Menús y barras de herramientas (7)

- **Menú Principal / Submenú**

- Mediante código:

```
public class MainActivity extends AppCompatActivity {
  private static final int MNU_OPC1 = 1;
  /* Resto de constantes para las opciones... */
  public boolean onCreateOptionsMenu(Menu menu) {
    menu.add(Menu.NONE, MNU_OPC1, Menu.NONE, "Opcion1")
        .setIcon(android.R.drawable.ic_menu_preferences);
    menu.add(Menu.NONE, MNU_OPC2, Menu.NONE, "Opcion2")
        .setIcon(android.R.drawable.ic_menu_compass);
    SubMenu smnu =
      menu.addSubMenu(Menu.NONE, MNU_OPC3, Menu.NONE, "Opcion3")
          .setIcon(android.R.drawable.ic_menu_agenda);
    smnu.add(Menu.NONE, SMNU_OPC31, Menu.NONE, "Opcion 3.1");
    smnu.add(Menu.NONE, SMNU_OPC32, Menu.NONE, "Opcion 3.2");
    return true;
  }
```

Menús y barras de herramientas (8)

- **Menú Principal / Submenú**

Ya sólo nos faltaría procesar la opción seleccionada del menú implementando el método *onOptionsItemSelected* que recibirá como parámetro el ítem seleccionado.

A partir de ese objeto se puede obtener su id mediante el método *getItemId*, y luego estableceremos la operación a realizar en función de la opción seleccionada, utilizando R.id.opción si lo creamos mediante XML, y la constante asociada si fue mediante código.

161

Menús y barras de herramientas (9)

- **Menú Principal / Submenú**

- Mediante archivo XML:

```
public boolean onOptionsItemSelected(MenuItem item) {
   TextView txt = (TextView)findViewById(R.id.txtMensa);
   switch (item.getItemId()) {
      case R.id.menuOp1:
         txt.setText("Opcion 1 pulsada!"); return true;
      case R.id.menuOp2:
         txt.setText("Opcion 2 pulsada!"); return true;
      case R.id.subMenu3Op1:
         txt.setText("Opcion 3.1 pulsada!"); return true;
      case R.id.subMenu3Op2:
         txt.setText("Opcion 3.2 pulsada!"); return true;
      default:
         return super.onOptionsItemSelected(item);
    }
}
```

162

Menús y barras de herramientas (10)

- **Menú Principal / Submenú**

- Mediante código:

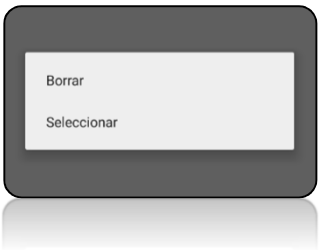
```
public boolean onOptionsItemSelected(MenuItem item) {
   TextView txt = (TextView)findViewById(R.id.txtMensa);
   switch (item.getItemId()) {
      case MNU_OPC1:
         txt.setText("Opcion 1 pulsada!"); return true;
      case MNU_OPC2:
         txt.setText("Opcion 2 pulsada!"); return true;
      case SMUOPC31:
         txt.setText("Opcion 3.1 pulsada!"); return true;
      case SMUOPC32:
         txt.setText("Opcion 3.2 pulsada!"); return true;
      default:
         return super.onOptionsItemSelected(item);
    }
}
```

163

Menús y barras de herramientas (11)

- **Menú Contextual**

El menú contextual se puede asociar a cualquier control de nuestra app: botón, etiqueta, elemento de una lista, etc. y accederemos a él mediante una pulsación larga sobre el control asociado.



164

Menús y barras de herramientas (12)

- **Menú Contextual**

Para ver como funciona este tipo de menú vamos a crear una app muy sencilla cuya actividad principal tendrá únicamente dos etiquetas:

- txtMenu: estará asociada a un menú contextual con dos opciones: "Escribir mensaje" y "Borrar".
- txtMensa: mostrará el mensaje o no en función de la opción seleccionada.

165

Menús y barras de herramientas (13)

- **Menú Contextual**

En primer lugar crearemos el menú.

Podemos hacerlo mediante archivo XML o mediante código igual que en los menús anteriores.

menu_eticontextual.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
  <item android:id="@+id/opcBorrar"
        android:title="Mostrar"/>
  <item android:id="@+id/opcEscribir"
        android:title="Borrar"/>
</menu>
```

166

Menús y barras de herramientas (14)

- **Menú Contextual**

A continuación asociamos la etiqueta al menú contextual mediante llamando al método *registerForContextMenu* dentro de *onCreate*.

```
public class MainActivity extends AppCompatActivity {
   protected TextView menu;

   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);

      //Asociamos los menús contextuales a los controles
      menu = (TextView)findViewById(R.id.txtMenu);
      registerForContextMenu(menu);
   }
```

167

Menús y barras de herramientas (15)

- **Menú Contextual**

Mediante *onCreateContextMenu* inflamos el menú tal y como hacíamos en los menús anteriores, pero con la diferencia de que este método se llamará con cada pulsación larga sobre el control asociado.

```
public void onCreateContextMenu(ContextMenu menu, View v,
  ContextMenu.ContextMenuInfo menuInfo) {
    super.onCreateContextMenu(menu, v, menuInfo);
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.menu_eticontextual, menu);
}
```

168

Menús y barras de herramientas (16)

- **Menú Contextual**

Por último establecemos la acción a realizar para cada opción del menú con el método *onContextItemSelected* que recibirá un objeto *MenuItem* con el ítem seleccionado.

```
public boolean onContextItemSelected(MenuItem item) {
   TextView texto = (TextView)findViewById(R.id.txtMensa);
   switch (item.getItemId()) {
      case R.id.opcBorrar:
         texto.setText("");   return true;
      case R.id.opcEscribir:
         texto.setText("Mensaje");  return true;
      default:
         return super.onContextItemSelected(item);
   }
}
```

Menús y barras de herramientas (17)

- **Menú Contextual**

En una misma actividad podemos tener varios componentes con un menú contextual asociado.

El proceso de creación de cada menú será idéntico al del ejemplo anterior, pero en el método *onCreateContextMenu* tendremos que identificar la vista que ha realizado la llamada al menú contextual para inflar el menú que le corresponde.

170

Menús y barras de herramientas (18)

- **Menú Contextual**

En primer lugar crearemos todos los menús que necesitemos (XML o Java) y registraremos los componentes que tengan un menú de contexto asociado.

```
protected ListView lista;

protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_main);
   // Asociamos los menús contextuales a los controles
   TextView titulo = (TextView)findViewById(R.id.txtMenu);
   registerForContextMenu(titulo);
   lista = (ListView)findViewById(R.id.listElem);
   registerForContextMenu(lista);
}
```

Menús y barras de herramientas (19)

- **Menú Contextual**

En el método *onCreateContextMenu*, inflaremos el menú que corresponda en función del componente seleccionado.

Además, como en el ejemplo se ha asociado un menú a los ítem de una lista, hemos recogido el elemento seleccionado para utilizar su título como cabecera del menú de contexto.

172

Menús y barras de herramientas (20)

- **Menú Contextual**

```
public void onCreateContextMenu(ContextMenu menu, View v,
ContextMenu.ContextMenuInfo menuInfo) {

   super.onCreateContextMenu(menu, v, menuInfo);
   MenuInflater inflater = getMenuInflater();

   if(v.getId() == R.id.txtTitulo) {
      inflater.inflate(R.menu.menu_eticontextual, menu);
   } else if(v.getId() == R.id.listElem) {
      AdapterView.AdapterContextMenuInfo info =
         (AdapterView.AdapterContextMenuInfo)menuInfo;
      String titulo = ((Titular)lista.getAdapter().
         getItem(info.position)).getTitulo();
      menu.setHeaderTitle(titulo);
      inflater.inflate(R.menu.menu_itemcontextual, menu);
   }
}
```

Menús y barras de herramientas (21)

- **Menú Contextual**

Para acabar establecemos la operación asociada a cada opción de cada uno de los menús de contexto en el método *onContextItemSelected*.

Para ello, podemos crear un único *switch* con el valor de *item.getItemId()*, y establecer el comportamiento de todas las opciones con independencia del menú de contexto al que pertenezcan.

174

Menús y barras de herramientas (22)

- **Menú Contextual**

```
public boolean onContextItemSelected(MenuItem item) {
   TextView texto = (TextView)findViewById(R.id.txtMensaje);
   AdapterView.AdapterContextMenuInfo info =
      (AdapterView.AdapterContextMenuInfo) item.getMenuInfo();
   switch (item.getItemId()) {
      case R.id.opcBorrar:        // Menu titulo
         texto.setText(""); return true;
      case R.id.opcEscribir:      // Menu titulo
      case R.id.opcOcultarItem:   // Menu lista
         texto.setText("Nada Seleccionado"); return true;
      case R.id.opcMostrarItem:   // Menu lista
         String titulo =
         ((Titular)lista.getAdapter().getItem(info.position)).
         getTitulo();
         texto.setText("Seleccionado: " + titulo);return true;
      default:
         return super.onContextItemSelected(item);
   }
}
```

Menús y barras de herramientas (23)

- ***Action Bar / App Bar***

La *Action Bar* o *App Bar* es la barra de título y herramientas que aparece en la parte superior de la mayoría de aplicaciones.

Normalmente muestra el título de la *app* o actividad, varios botones de acción con las opciones principales y un menú desplegable (*overflow*) con el resto de opciones.

176

Menús y barras de herramientas (24)

- ***Action Bar / App Bar***

Podemos utilizar la *Action Bar* de 2 formas:

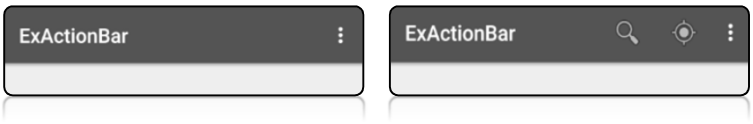
- *Action Bar* por defecto que se añade a las actividades al utilizar un tema (*theme*) en la aplicación.
- Componente *Toolbar* disponible desde Android 5.0, pero compatible con las versiones anteriores.

177

Menús y barras de herramientas (25)

- ***Action Bar / App Bar***

Si al crear nuestra aplicación en lugar de seleccionar "*Empty Activity*", seleccionamos "*Basic Activity*" obtendremos una aplicación que ya incorpora de forma automática una *Action Bar*, cuyas opciones vendrán definidas en el archivo: "res/menu/menu_main.xml"



178

Menús y barras de herramientas (26)

- ***Action Bar / App Bar***

El título que se muestra se define mediante la propiedad "label" en el manifiesto de la aplicación, y normalmente será un *string* definido en "res/values/strings.xml".

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/..."
    package="com.example.oscar.exactionbar">
    <application
        android:label="@string/app_name"
        ...
</application>
</manifest>
```

179

Menús y barras de herramientas (27)

- ***Action Bar / App Bar***

Los botones de acción y las opciones del menú *overflow* se definen agregando elementos ítem al menú y estableciendo sus propiedades:

- ***android:id***, identificador del ítem.
- ***android:title***, texto que se visualizará para la opción.
- ***android:icon***, icono asociado a la acción.

180

Menús y barras de herramientas (28)

- ***Action Bar / App Bar***
  - ***app:showAsAction***, en una Action Bar indica si el ítem se mostrará como botón de acción o como parte del menú de *overflow:*
    - *ifRoom*: se mostrará como botón de acción sólo si hay espacio disponible.
    - *withText*: en caso de mostrarse como botón de acción se añadirá al icono el texto de la opción.
    - *never*: se mostrará siempre en el menú *overflow*.
    - *always*: se mostrará siempre como botón de acción. Si no hay espacio suficiente puede provocar que los elementos se solapen.

181

Menús y barras de herramientas (29)

- ***Action Bar / App Bar***

```
<menu
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    tools:context="com.example.oscar.exactionbar.MainActivity">
    <item
        android:id="@+id/actionBuscar"
        android:icon="@android:drawable/ic_menu_search"
        app:showAsAction="always"
        android:title="Buscar" />
    <item
        android:id="@+id/actionLocalizacion"
        android:icon="@android:drawable/ic_menu_mylocation"
        app:showAsAction="always"
        android:title="Localizacion" />
    <item
        android:id="@+id/action_settings"
        android:orderInCategory="100"
        app:showAsAction="never"
        android:title="Ajustes" />
</menu>
```