

Menus

Programació Multimèdia i Dispositius Mòbils

2n. DAM

Menus



- Que son? <http://developer.android.com/guide/topics/ui/menus.html>

Los menús son una parte importante de una actividad de una interfaz de usuario, que aporta a los usuarios una manera sencilla de realizar acciones.

Existen tres tipos de menús para aplicaciones:

- **Menú de opciones (Option Menu)**

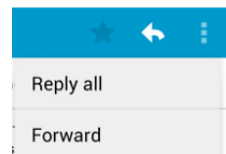
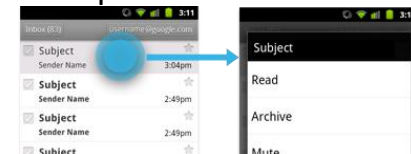
- Colección de items de menú para una actividad, que aparece cuando el usuario toca el botón de MENU. Cuando la aplicación se ejecuta en Android 3.0 o en una versión posterior, se pueden colocar los acceso rápidos para seleccionar los items de un menú en una Barra de Acción, como "items de acción."

- **Menú de Contexto (Context Menu)**

- Lista flotante de items de menú que aparece cuando el usuario hace click en un view que está registrado para aportar un menú de contexto.

- **Menú Emergente (pop-up Menu)**

- Lista flotante de items en un menú que aparece cuando el usuario hace click en un item de un menú que contiene un menú anidado.





Menus

- Para todos los tipos de menú, Android proporciona un formato estándar XML para definir los elementos del menú.
 - En lugar de construir un menú en el código del activity, debemos definir un menú y todos sus artículos en un recurso de menú XML.
 - A continuación, inflaremos el recurso de menú (cargarlo como un objeto Menu) en el activity o fragment.
- El uso de un recurso de menú es una buena práctica por varias razones:
 - Es más fácil de visualizar la estructura de menús en XML.
 - Se separa el contenido del menú a partir del código de conducta de la aplicación.
 - Esto le permite crear configuraciones de menú alternativos para diferentes versiones de la plataforma, tamaños de pantalla y otras configuraciones, aprovechando el marco de recursos de aplicaciones.
- Para definir el menú, crearemos un archivo XML dentro **res/menu/** del directorio del proyecto y crearemos el menú con los siguientes elementos:

<menú>

Define un menú, que es un contenedor para los elementos del menú. Un elemento <menu> debe ser el nodo raíz del archivo y puede contener uno o más elementos <item> o <grupo>.

<item>

Crea un MenuItem, que representa un solo elemento de un menú. Este elemento puede contener un elemento <menu> anidado con el fin de crear un submenú.

<grupo>

Un contenedor opcional, invisible para los elementos <item>. Esto le permite clasificar los elementos de menú por lo que comparten características tales como estado activo y la visibilidad. Para obtener más información, consultad la sección sobre Creación de Grupos de menús.

<http://developer.android.com/guide/topics/resources/menu-resource.html>

Fichero: game_menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu
xmlns:android="http://schemas.android.com/apk/res/android">
  <item android:id="@+id/new_game"
        android:icon="@drawable/ic_new_game"
        android:title="@string/new_game"/>
  <item android:id="@+id/help"
        android:icon="@drawable/ic_help"
        android:title="@string/help" />
</menu>
```



Menus

Inflar el recurso de Menu:

- Desde el código de la aplicación, se puede inflar un recurso de menú (convertir un recurso XML en un objeto programable) utilizando `MenuInflater.inflate()`. Como ejemplo, tenemos el siguiente código que infla el archivo `game_menu.xml` definido anteriormente, en el método callback `onCreateOptionsMenu()`, para utilizar el menú como la actividad de Menú de Opciones:


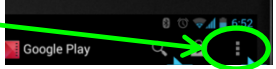
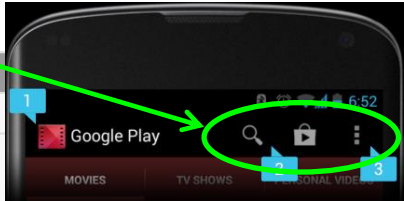
```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.game_menu, menu);
    return true;
}
```

- El método `getMenuInflater()` devuelve un objeto *MenuInflater* a la actividad. Con este objeto, se puede llamar al método `inflate()`, que infla un recurso de menú y lo convierte a un objeto *Menu*. En este ejemplo, el recurso de menú definido por `game_menu.xml` es inflado al `Menu` que se ha pasado al método `onCreateOptionsMenu()`.



Menu de Opciones

Creando un Menú de Opciones

- El menú de opciones es donde debemos incluir acciones y otras opciones que son relevantes para el contexto de la actividad actual, como "Buscar", "Redactar correo" y "Configuración".
- Cuando los elementos del menú de opciones aparecen en la pantalla depende de la versión para la que ha desarrollado su aplicación:
 - Aplicación para **Android 2.3.x (nivel de API 10) o menos**, el contenido del menú de opciones aparece en la parte inferior de la pantalla cuando el usuario pulsa el botón Menú.
 - Cuando se abre, la primera parte que se ve es el icono de menu, que contiene hasta seis opciones de menu.
 - Si el menu incluye más de seis artículos, Android coloca el sexto punto y el resto en el menú de desbordamiento, que el usuario puede abrir seleccionando más.
 - Aplicación para **Android 3.0 (nivel de API 11) y superiores**, los elementos del menú de opciones están disponibles en la barra de acción. Por defecto, el sistema coloca todos los elementos en el desbordamiento de la acción, que el usuario puede mostrar el icono de desbordamiento de la acción en el lado derecho de la barra de acción (o pulsando el botón de menú del dispositivo, si está disponible).
 - Para habilitar el acceso rápido a acciones importantes, se puede hacer que algunos elementos que aparezcan en la barra de acciones añadiendo android: showAsAction = "ifRoom" para los elementos <item> correspondientes.

Valor	Descripción
ifRoom	Sólo sitúa el item en el ActionBar "si hay suficiente espacio".
withText	También incluye el texto del atributo título (android: título) al'ActionBar. Se puede incluir este atributo junto a otros, separándolos por el caracter (pipe) ' '. <i>(A green arrow points from this text to the 'ifRoom' attribute in the code snippet above.)</i>
never	No sitúa nunca este item al Action Bar
always	Sitúa siempre este item a la Action Bar



Menu de Opciones

Cuando el sistema Android crea por primera vez el menú de opciones, llama al método `onCreateOptionsMenu()` del activity.

Respuesta a la acción del usuario

- Cuando el usuario selecciona un item del menu del menu de opciones (incluyendo los items de acción de la barra de acción), el sistema llama al método `onOptionsItemSelected()` de la actividad.
- Este método pasa el `MenuItem` elegido por el usuario.
- Se puede identificar el item del menú llamando al método `getItemId()`, que devuelve el ID único para ese item del menú (definido por el atributo `android:id` en el recurso del menú o mediante un integer dado al método `add()`).
- Se puede comparar este ID con los items del menú conocidos y realizar la acción apropiada.

Por ejemplo:

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    // Tractem la selecció d'Items
    switch (item.getItemId()) {
        case R.id.new_game: newGame(); return true;
        case R.id.help: showHelp(); return true;
        default: return super.onOptionsItemSelected(item);
    }
}
```



Menu de Opciones

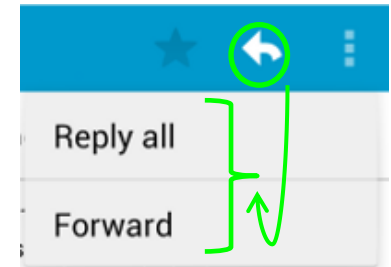
Cambiar los items del menú en tiempo de ejecución

- Una vez que la actividad se crea, se llama al método `onCreateOptionsMenu()` una sola vez, tal y como se describe anteriormente. El sistema guarda y reutiliza el Menu que se ha definido en este método hasta que la actividad se destruya. Si se quiere cambiar el menú de opciones en cualquier momento después de creado, se debe sobrescribir el método `onPrepareOptionsMenu()`. Este método pasa el objeto Menu tal y como existe en este momento. Esto es útil si se quiere eliminar, añadir, deshabilitar o habilitar los items del menú dependiendo del estado en curso de la aplicación.
- En Android 2.3 y versiones anteriores, el sistema llama al método **`onPrepareOptionsMenu()`** cada vez que el usuario abre el menú de opciones.
- En Android 3.0 y superiores, se debe llamar al método **`invalidateOptionsMenu()`** cuando se quiere actualizar el menú, ya que el menú siempre está abierto. El sistema llamará al método `onPrepareOptionsMenu()` para que se puede actualizar los items del menú.



Menus Emergentes

- Un menu emergente es un submenú que el usuario puede abrir si selecciona un item en otro menú. Se puede añadir un menu emergente a cualquier menú (exceptuando un Menu emergente).



- Los submenús son útiles cuando la aplicación tiene muchas funciones que pueden ser organizados en temas, como por ejemplo los items en una barra de menú en un PC (File, Edit, View, etc.)
- Cuando se crea el recurso de menu, se puede crear un menu emergente añadiendo un elemento <menu> como hijo de un <item>.



Menus Emergentes

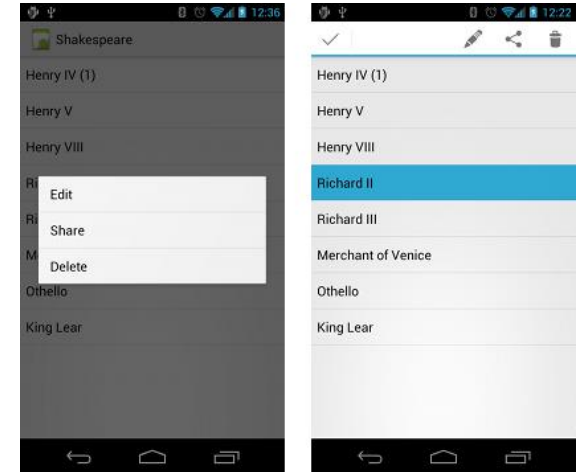
```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
  <item android:id="@+id/file" android:icon="@drawable/file" android:title="@string/file" >
    <!-- "file" pop-up menu-->
    <menu>
      <item android:id="@+id/create_new" android:title="@string/create_new" />
      <item android:id="@+id/open" android:title="@string/open" />
    </menu>
  </item>
</menu>
```

Cuando el usuario selecciona un item de un menu emergente, el método callback del item seleccionado del correspondiente menú padre, recibe el evento. Si, por ejemplo, el menú anterior se utiliza como un menú de opciones, cuando un item del submenú se selecciona se llama al método `onOptionsItemSelected()`.



Menus de Contexto

- Es conceptualmente similar al menú mostrado cuando el usuario realiza "click en el botón derecho" en un PC.
- Permite darle acceso al usuario a acciones que pertenecen a un item específico en la interfaz de usuario.
- Se muestra cuando el usuario realiza un "long click" (pulsar y mantener) en un item.
- Se puede crear un context menu para cualquier View.





Menus de Contexto

- Para que un View suministre un context menu, se debe "registrar" el view para un context menu. Hay que llamar al método `registerForContextMenu()` y pasarle el View que se quiere dar al context menu. Cuando este View recibe el largo click, muestra el context menu.
- Para definir el aspecto y el comportamiento del context menu, hay que sobrescribir los métodos `onCreateContextMenu()` y `onContextItemSelected()`



Menus de Contexto

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

MenuInflater es utilizado para inflar el context menu desde un recurso de menú. (También se puede utilizar el método `add()` para añadir items de menú.) Los parámetros del método callback incluyen el **View** que el usuario ha seleccionado y un objeto **ContextMenu**. **ContextMenuInfo** aporta información adicional sobre el item seleccionado.

Se pueden utilizar estos parámetros para determinar que context menu crear, aunque en este ejemplo, todos los context menus para la actividad son iguales.

Cuando el usuario selecciona un item del context menu, el sistema llama al método `onContextItemSelected()`.



Menus de Contexto

- Ejemplo de cómo manejar los items seleccionados:

```
@Override
public boolean onContextItemSelected(Menuitem item) {
    AdapterContextMenuInfo info = (AdapterContextMenuInfo) item.getMenuInfo();
    switch (item.getItemId()) {
        case R.id.edit: editNote(info.id);
                        return true;
        case R.id.delete: deleteNote(info.id);
                        return true;
        default: return super.onContextItemSelected(item);
    }
}
```

- getItemId() obtiene el ID de los item seleccionados del menú y un switch comprueba uno a uno los IDs que están definidos en el recurso del menú.
- La sentencia default llama al mismo método de la clase padre por si ésta puede manejar los items del menú que no son gestionados aquí.
- En este ejemplo, el item seleccionado es un item del ListView. Para realizar una acción en el item seleccionado, la aplicación tiene que saber el ID de la lista para ese item seleccionado (su posición en la ListView). Para obtener este ID, la aplicación llama al método getMenuInfo(), que devuelve un objeto AdapterView.AdapterContextMenuInfo que incluye el ID de la lista para el item seleccionado en el campo id. Los métodos locales editNote() y deleteNote() aceptan que este ID de la lista realice una acción con los datos especificados por el ID de la lista.
- **Nota:** Los items en un context menu no soportan iconos o accesos directos



Grupos de opciones

- Los *grupos de opciones* nos permiten agrupar varios elementos de un menú de forma que podamos aplicarles ciertas acciones o asignarles determinadas características o funcionalidades de forma conjunta.
 - Mostrar u ocultar todos los ítems con el método `setGroupVisible()`
 - Habilitar o deshabilitar todos los ítems con el método `setGroupEnabled()`
 - Especificar si todos los ítems son chequeables con `setGroupCheckable()`

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
  <item android:id="@+id/item1" android:icon="@drawable/item1"
        android:title="@string/item1" />
  <!-- menu group -->
  <group android:id="@+id/group1">
    <item android:id="@+id/groupItem1" android:title="@string/groupItem1" />
    <item android:id="@+id/groupItem2" android:title="@string/groupItem2" />
  </group>
</menu>
```

- Los ítems que están en el grupo se muestran igual que el primer ítem que no está en el grupo



Grupos de opciones

- En cambio, se pueden modificar las características de los dos ítems del grupo referenciando el ID del grupo y utilizando los métodos listados anteriormente.

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
  <item android:id="@+id/item1" android:icon="@drawable/item1"
        android:title="@string/item1" />
  <!-- menu group -->
  <group android:id="@+id/group1">
    <item android:id="@+id/groupItem1" android:title="@string/groupItem1" />
    <item android:id="@+id/groupItem2" android:title="@string/groupItem2" />
  </group>
</menu>
```