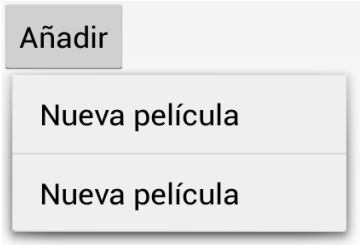
Menús

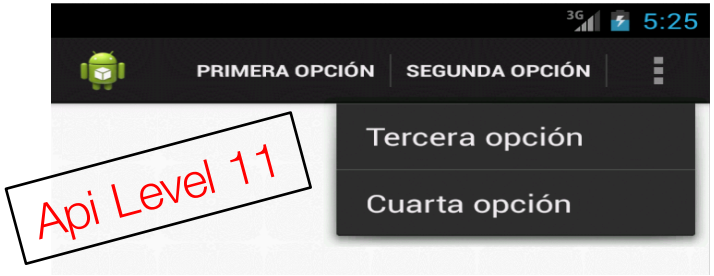
Options menu

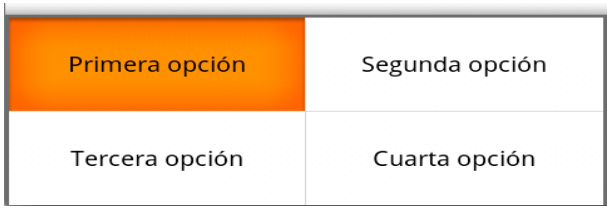
Context menu

Popup menu



Menús



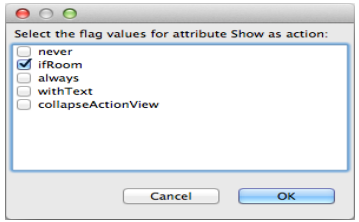


/res/menu/activity_main.xml

```
<menu xmlns:android="http://schemas.android.com...">
    <item android:id="@+id/menu_option_1"
          android:showAsAction="ifRoom"
          android:title="@string/menu_option_1"/>
    <item android:id="@+id/menu_option_2"
          android:showAsAction="always"
          android:title="@string/menu_option_2"/>
    <item android:id="@+id/menu_option_3"
          android:showAsAction="never"
          android:title="@string/menu_option_3"/>
    <item android:id="@+id/menu_option_4"
          android:showAsAction="never"
          android:title="@string/menu_option_4"/>
</menu>
```

Menús

```
<item android:id="@+id/menu_option_1"
    android:title="@string/menu_option_1_title"
    android:titleCondensed="@string/menu_op1_titleCondensed"
    android:icon="@android:drawable/ic_menu_xxxx"
    android:showAsAction="ifRoom"
    android:onClick="methodName"
    android:checkable="true"
    android:checked="false"
    android:visible="true"
    android:enabled="false"
    android:numericShortcut="8"
    android:alfabeticShortcut="b"/>
```



<item> Representa una opción del menú

android:id
Permite asignar un identificador al elemento del menú

android:title
Especifica el titulo del elemento del menú (obligatorio)

android:icon
Especifica el icono que representa el elemento del menú

android:showAsAction

ifRoom	sólo se muestra si hay sitio
always	siempre se muestra
never	nunca se muestra
withText	además del icono también muestra el title
collapseActionView	el <item> puede tener asociado un View que puede desaparecer

android:onClick="methodName"
Representa el nombre del método del Activity que se ejecutará al seleccionar este elemento del menú. La firma de este método tendrá que ser:

```
public void methodName(MenuItem menuItem)
```

Este método tendrá preferencia sobre el método

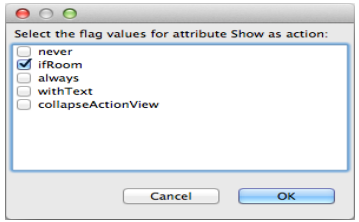
```
@Override
public boolean onOptionsItemSelected (MenuItem item)
```

Que se ejecuta por defecto cuando se selecciona cualquier elemento del menú.

Menús

<item> Representa una opción del menú

```
<item android:id="@+id/menu_option_1"
    android:title="@string/menu_option_1_title"
    android:titleCondensed="@string/menu_op1_titleCondensed"
    android:icon="@android:drawable/ic_menu_xxxx"
    android:showAsAction="ifRoom"
    android:onClick="methodName"
    android:checkable="true"
    android:checked="false"
    android:visible="true"
    android:enabled="false"
    android:numericShortcut="8"
    android:alfabeticShortcut="b"/>
```



android:checkeable="true|false"
Indica que el elemento del menú es checkeable

android:checked="true|false"
Si el elemento del menú es checkeable, en este atributo se indica su estado

android:visible="true|false"
Indica si el elemento del menú se mostrará o no en la pantalla

android:enabled="true|false"
Aun cuando se muestre un elemento del menú, puede inhabilitarse lo que implica que el usuario no puede seleccionarlo

android:numericShortcut="*número*"
Representa la tecla numérica que actúa como atajo para acceder l elemento del menú

android:alfabeticShortcut="*carácter*"
Representa la tecla alfabética que actúa como atajo para acceder l elemento del menú

Menús

```
<menu>
    <group android:id="@+id/grp1"
           android:checkableBehavior="single"
           android:visible="true"
           android:enabled="false">

        <item android:id="@+id/menu_option_1"
              android:showAsAction="ifRoom"
              android:title="@string/menu_option_1"/>

        <item android:id="@+id/menu_option_2"
              android:showAsAction="always"
              android:title="@string/menu_option_2"/>

    </group>
</menu>
```

<group> Representa un conjunto de opciones del menú sobre las que se puede fijar propiedades en bloque

android:checkableBehavior

- none: ninguno de los elementos del menú es checkable
- all: Todos los elementos del menú son checkables choice mode multiple
- single: Todos los elementos del menú son checkables choice mode single

android:visible="true|false"

Se puede ocultar o mostrar en bloque todos los elementos del menú que engloba el grupo

android:enabled="true|false"

Se puede habilitar o deshabilitar la acción de todos los elementos del menú que engloba el grupo

```
menu.setGroupCheckable(int groupId, boolean checkable, boolean exclusive)
menu.setGroupEnabled(int groupId, boolean enabled)
menu.setGroupVisible(int groupId, boolean visible)

item.setCheckable(boolean checkable);
item.setChecked(boolean isChecked);
Item.isChecked();
item.setEnabled(boolean enabled);
item.setVisible(boolean visible);
```

Options menu

1 2 3

```
public class MyActivity  extends Activity
{
   : : : :
   @Override
   public boolean onCreateOptionsMenu(Menu menu)
   {
        MenuInflater menuInflater = getMenuInflater();
        menuInflater.inflate(R.menu.identificador, menu);
        return true;
   }
```

Crea OptionsMenu

Options menu

1 2 3

```
public class MyActivity extends Activity
{
   @Override
   public boolean onOptionsItemSelected(MenuItem menuItem)
   {
       switch ( menuItem.getId() )
       {
           case R.id.menuItemID :
                                  .......
                                  return true;

           default:   return super.onOptionsItemSelected(menuItem);
       }
   }
```

Ejecutado cuando el usuario elige un elemento del menú

Se ha completado la acción que representa el elemento del menú

No se ha tratado la acción que representa el elemento del menú

```
public void metodoDefinidoEnAtributoOnClick(MenuItem item)
{
   : : :
}
```

Ejecutado si así se indicó en el atributo onClick del elemento <item> del <menu>

Options menu

1 2 3

```
public class MyActivity  extends Activity
{
   @Override
   public boolean onPrepareOptionsMenu(Menu menu)
   {
      : : :
      if ( . . . )
      {
          MenuItem menuItem = menu.findItem(R.id.menuItemID);
          menuItem.setEnabled( false ); // .setCheckable()    .setVisible()
          return true;
      }
     : : :
   }
```

Es invocado por el sistema para poder configurar el menú.

Antes del API Level 11 era ejecutado justo antes de mostrar el menú

A partir del API Level 11 sólo es ejecutado si previamente se ha requerido, para lo que es necesario invocar a ***invalidateOptionsMenu()***

Localiza un elemento del menú a través de su identificador

Context menu

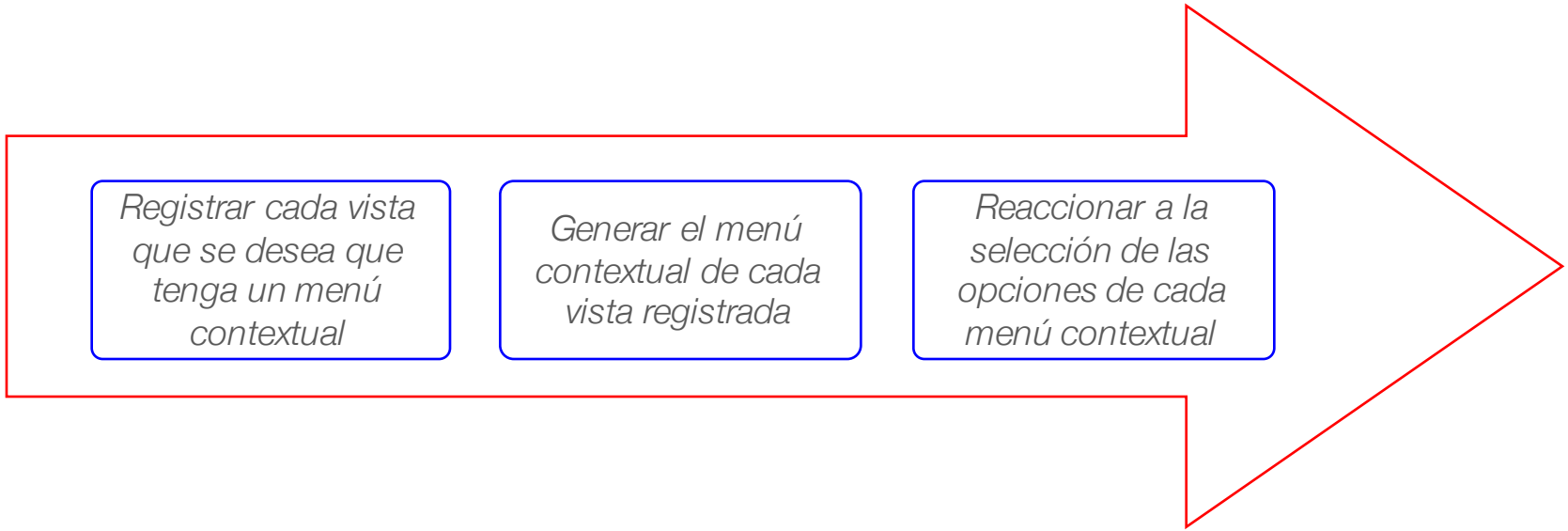
Floating context menu

Este menú aparece cuando se realiza una pulsación larga sobre una vista

Contextual action mode

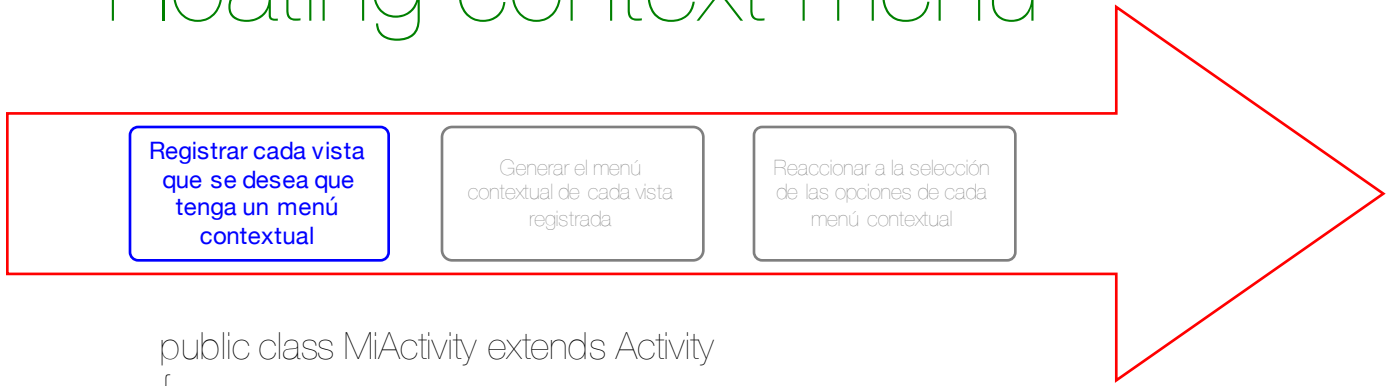
Este menú aparece cuando se desea realizar acciones con items seleccionados de una lista

Floating context menu



Floating context menu

1 2 3

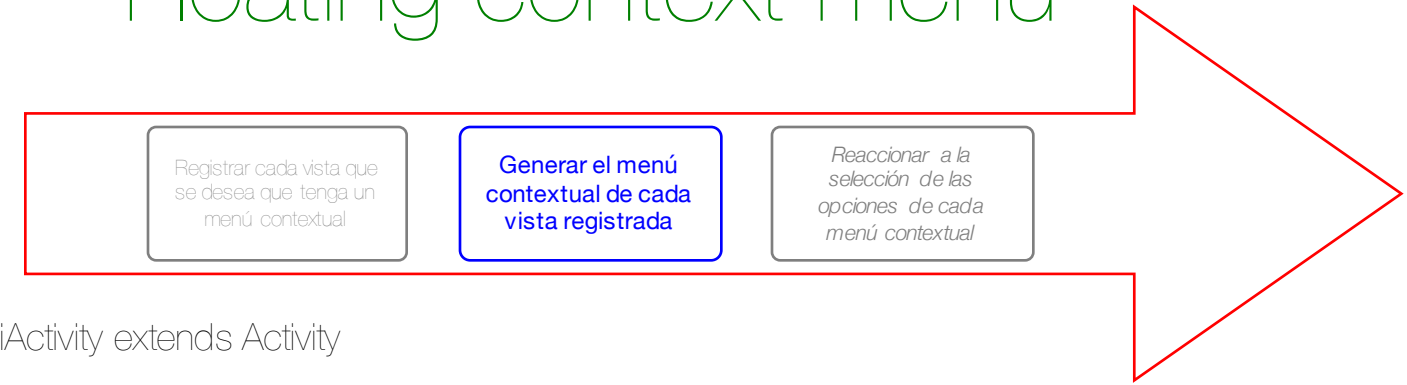



```
public class MiActivity extends Activity
{
  private View view;

  public boolean onCreate(Bundle savedInstanceState)
  {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.layout_id);
        view = (View) findViewById(R.id.view_id);
        registerForContextMenu(view);
        : : :
  }
}
```

Floating context menu

1 2 3




```
public class MiActivity extends Activity
{
 : : :
  @Override
  public boolean onCreateContextMenu(ContextMenu menu, View view, ContextMenuInfo menuInfo)
  {
    switch( view.getId() )
    {
      case R.id.view_id:   MenuInflater inflater = getMenuInflater();
                           inflater.inflate(R.menu.menu_id, menu);
                           return true;
       : : :
      default: return super.onCreateContextMenu(menu, view, menuInfo);
    }
  }
```

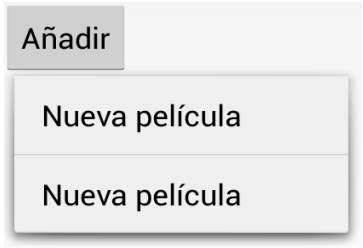
Floating context menu

1 2 3


```
public class MiActivity extends Activity
{
  : : :
  public boolean onContextItemSelected(MenuItem menuItem)
  {
       switch( menuItem.getItemId() )
       {
          case R.id.view_id:  ...
                                  return true;

          default: return super.onContextItemSelected(menuItem);
       }
```

Popup menu



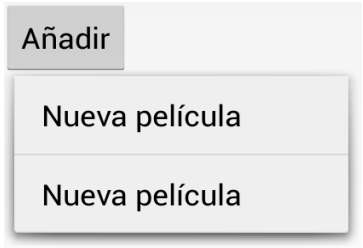
1

```
{
    PopupMenu popup = new PopupMenu( (Context) this, view);
    popup.setOnMenuItemClickListener(listener);
    popup.getMenuInflater().inflate(R.menu.actions);
    popup.show();
}

private OnMenuItemClickListener listener =    new OnMenuItemClickListener ()
                                              {
                                                    @Override
                                                    public boolean onMenuItemClick(MenuItem item)
                                                    {
                                                        switch ( item.getItemId() )
                                                        {
                                                            case R.id.option_1 :
                                                                                return true;

                                                            default:                    return false;
                                                        }
                                                    };
```

Popup menu



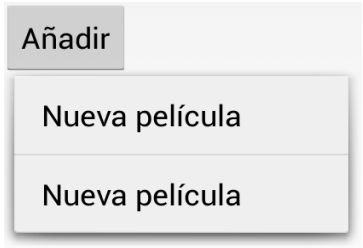
2

```
{
    PopupMenu popup = new PopupMenu( (Context) this, view);
    popup.setOnMenuItemClickListener(listener);
    popup.inflate(R.menu.actions);
    popup.show();
}

private PopupMenu.OnMenuItemClickListener listener = new PopupMenu.OnMenuItemClickListener ()
                                                    {
                                                        @Override
                                                        public boolean onMenuItemClick(MenuItem item)
                                                        {
                                                            switch ( item.getItemId() )
                                                            {
                                                                case R.id.option_1 : ....
                                                                                     return true;

                                                                default:             return false;
                                                            }
                                                        };
```

Popup menu



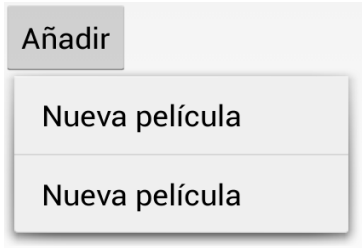
3

```
{
    PopupMenu popup = new PopupMenu( (Context) this, view);
    popup.setOnMenuItemClickListener(listener);
    popup.inflate(R.menu.popup_menu_id);
    popup.show();
}

private PopupMenu.OnMenuItemClickListener listener = new PopupMenu.OnMenuItemClickListener ()
                                                    {
                                                        @Override
                                                        public boolean onMenuItemClick(MenuItem item)
                                                        {
                                                            switch ( item.getItemId() )
                                                            {
                                                                case R.id.option_1 :
                                                                                    return true;

                                                                default:                    return false;
                                                            }
                                                        };
```

Popup menu



4

```
{
    PopupMenu popup = new PopupMenu( (Context) this, view);
    popup.setOnMenuItemClickListener(listener);
    popup.inflate(R.menu.popup_menu_id);
    popup.show();
}

private PopupMenu.OnMenuItemClickListener listener = new PopupMenu.OnMenuItemClickListener ()
                                                    {
                                                        @Override
                                                        public boolean onMenuItemClick(MenuItem item)
                                                        {
                                                            switch ( item.getItemId() )
                                                            {
                                                                case R.id.option_1 :
                                                                                    return true;

                                                                default:                    return false;
                                                            }
                                                        };
```