

Android: Controles

Entrada y salida de datos

Controls I

Alarm



Alarm

To

jay@gmail.com

Home

Home

Work

Other

Custom

Sync Browser

5/31/2012 4:58 PM



Sync Calendar

6/1/2012 11:15 AM



Sync Contacts

6/1/2012 3:50 PM



Off

On

OFF

ON

ATTENDING?

☒ Yes

☐ Maybe

☐ No

Set time



7



29



8

:

30

AM

9

31

PM



Cancel

Set

Set date



Sep



06



2010

Oct

07

2011

Nov

08

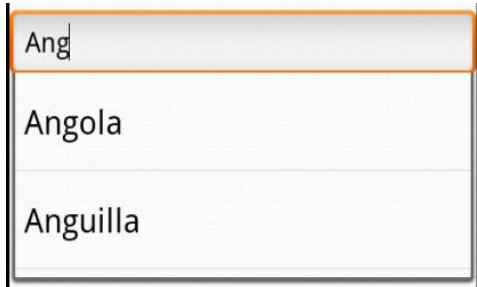
2012



Cancel

Set

Controles II

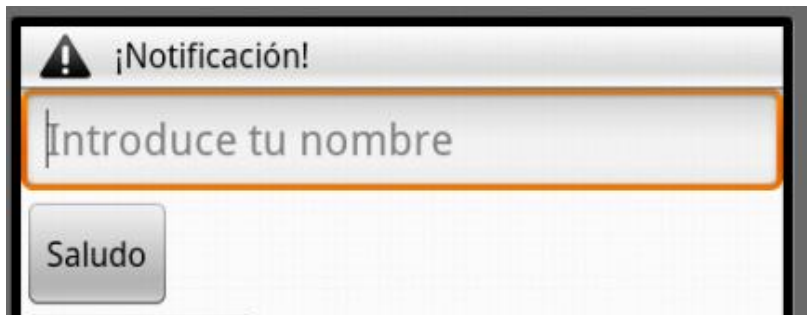
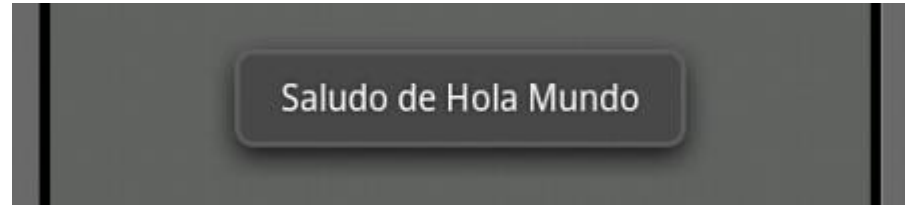


Ang

Angola

Anguilla

A list view with three items. The first item, 'Ang', is highlighted with an orange border, indicating it is the selected item.

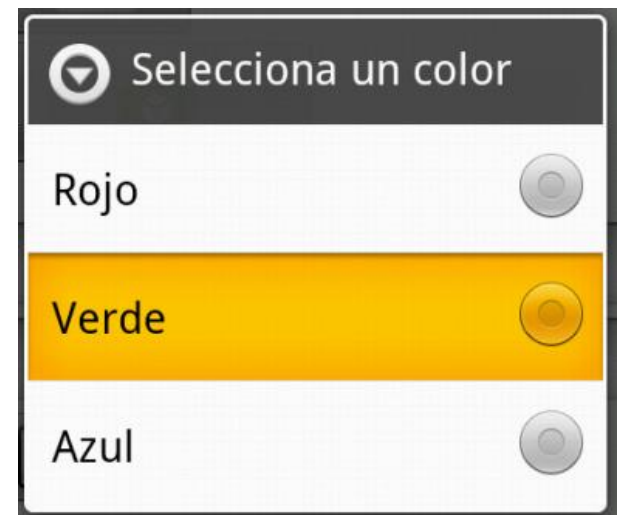
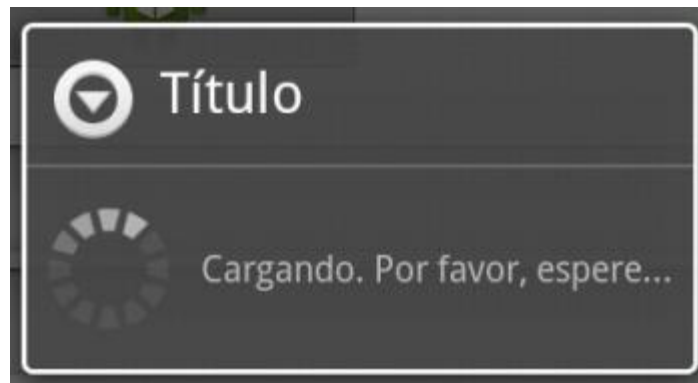
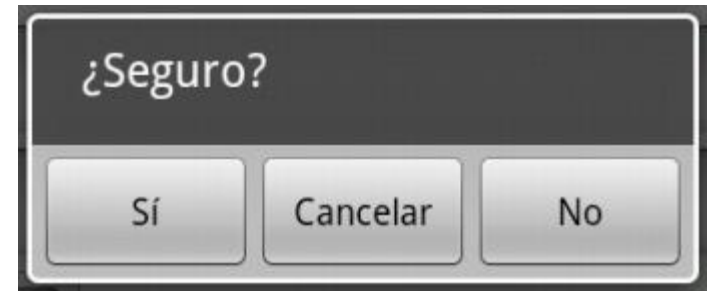


¡Notificación!

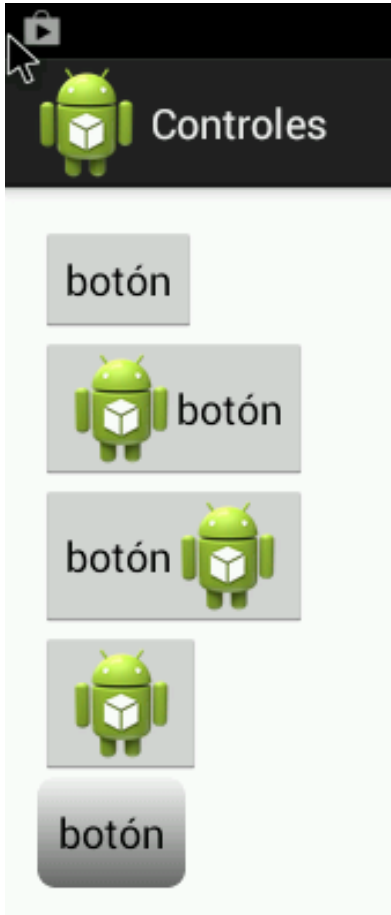
Introduce tu nombre

Saludo

A notification dialog box with a title bar containing a warning icon and the text '¡Notificación!'. Below the title bar is a text input field with the placeholder text 'Introduce tu nombre'. At the bottom left is a button labeled 'Saludo'.



Botones I



Los botones pueden ser de

- texto

```
<Button />
```

- texto e imagen

```
<Button android:drawableRight= "@drawable/imagen" />
```

- imagen

```
<ImageButton android:src= "@drawable/imagen" />
```

- usando estilos

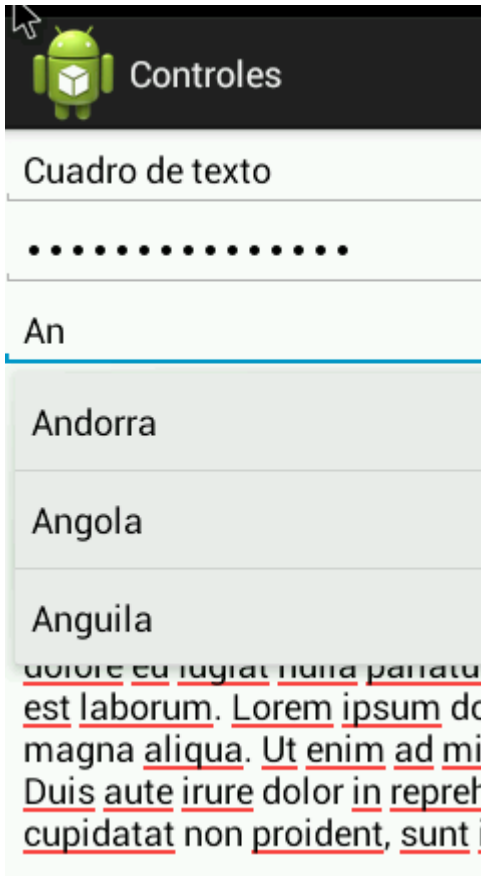
```
<Button style= "@style/botonGris" />
```

<http://developer.android.com/guide/topics/ui/themes.html>

Botones II

- ▶ `<Button`
 `android:id="@+id/boton"`
 `android:layout_width="wrap_content"`
 `android:layout_height="wrap_content"`
 `android:text="@string/boton" />`
- ▶ `onClick: android:onClick= "accion"`
- ▶ `public void accion(View view) {...}`
- ▶ `Button boton = (Button) findViewById(R.id.boton);`
 `boton.setOnClickListener(`
 `new View.OnClickListener() {`
 `public void onClick(View v) { ...`
 `}`
 `});`

Cuadros de texto



```
<EditText>  
    <requestFocus />  
</EditText>
```

```
<EditText android:inputType="textPassword" />
```

```
<AutoCompleteTextView/>
```

```
<EditText android:inputType="textMultiLine" />
```

Obtener el contenido de un cuadro de texto:

```
et.getText().toString()
```

Para cambiar el texto de un cuadro de texto se usa:

```
et.setText(String).
```

AutoCompleteTextView

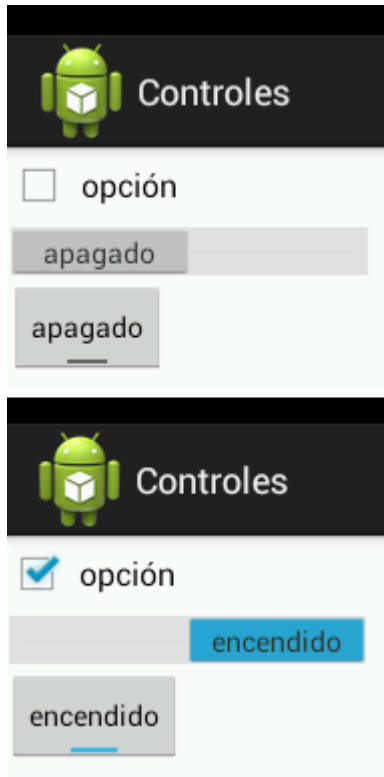
- ▶ Lista de valores para desplegar (strings.xml)

```
<string-array name="array_paises">  
    <item>Andorra</item>  
    <item>Angola</item>  
    <item>Anguila</item> ...  
</string-array>
```

- ▶ Asociar la lista al cuadro de texto: adaptador

```
actv = (AutoCompleteTextView) findViewById(R.id.actvPaises);  
String[] paises = getResources().getStringArray(R.array.array_paises);  
ArrayAdapter<String> adaptador = new ArrayAdapter<String>  
    (this, android.R.layout.simple_list_item_1, paises);  
actv.setAdapter(adaptador);
```

Casillas de verificación



```
<CheckBox android:text="@string/texto" />
```

```
<Switch  
    android:textOff="@string/textooff"  
    android:textOn="@string/textoon" />
```

```
<ToggleButton  
    android:textOff="@string/textooff"  
    android:textOn="@string/textoon" />
```

Para conocer y modificar sus estados se utilizan los métodos **isChecked()**, **setChecked(boolean)** y **toggle()**.

Para programar sus eventos se utiliza el atributo **android:onClick** o se les asocia un **OnClickListener** o un **OnCheckedChangeListener**.

ToggleButton

onClick	onClickListener	onCheckedChangeListener
<code><ToggleButton android:onClick="ver"/></code>	<code><ToggleButton /></code>	<code><ToggleButton /></code>
<pre>public void ver(View v){ if(tb.isChecked())... }</pre>	<pre>tb.setOnClickListener(new OnClickListener(){ @Override public void onClick (View arg0){ if(tb.isChecked()) ... } });</pre>	<pre>tb.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener(){ public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) { if(tb.isChecked()) ... } });</pre>
Ambas implementaciones son equivalentes. Estos eventos se ejecutan cada vez que pulsamos sobre el ToggleButton.		Este evento es diferente, se dispara cada vez que el valor del ToggleButton es modificado. Para obtener el valor del ToggleButton podemos usar <code>tb.isChecked()</code> o directamente el valor del parámetro <code>isChecked</code> .

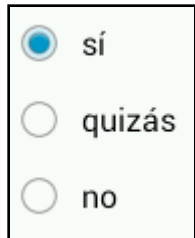
El CheckBox y el Switch se programan del mismo modo. El control de tipo Switch se ha introducido a partir de Android 4.0. Para poder usarlo se deberá modificar el manifiesto:

```
<uses-sdk  
    android:minSdkVersion="14"  
    android:targetSdkVersion="17" />
```

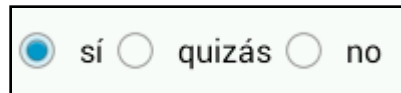
RadioButton y RadioGroup

- ▶ Los botones de radio permiten seleccionar una opción entre un grupo de opciones.

```
<RadioGroup>  
  <RadioButton android:text="..." />  
  <RadioButton .../>  
  ...  
</RadioGroup>
```



A vertical container with three radio buttons. The first button is selected (filled with blue) and is next to the text 'sí'. The second button is unselected (empty) and is next to the text 'quizás'. The third button is unselected (empty) and is next to the text 'no'.



A horizontal container with three radio buttons. The first button is selected (filled with blue) and is next to the text 'sí'. The second button is unselected (empty) and is next to the text 'quizás'. The third button is unselected (empty) and is next to the text 'no'.

Los métodos **isChecked()**, **toggle()** y **setChecked()** permiten comprobar y modificar el estado de un **RadioButton**.

Los métodos **getCheckedRadioButtonId()**, **check(int)** y **clearCheck()** permiten comprobar y modificar el estado de los botones de un **RadioGroup**.

Para programar los eventos de un **RadioButton** se utiliza el atributo **android:onClick** o se le asocia un **OnClickListener**.

OnCheckedChangeListener

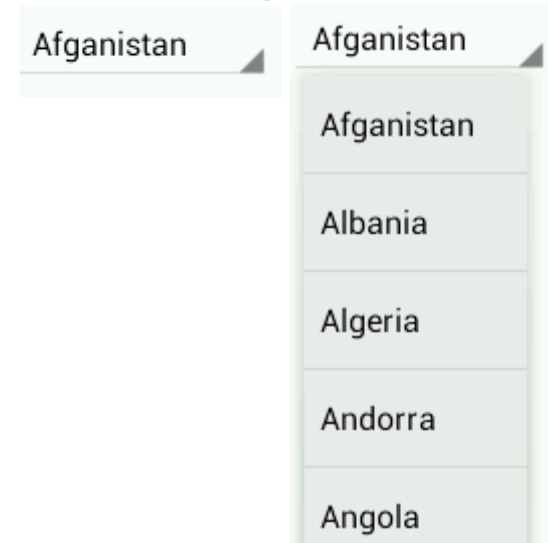
OnCheckedChangeListener //sobre un RadioGroup

```
rg.setOnCheckedChangeListener(new OnCheckedChangeListener() {  
    @Override  
    public void onCheckedChanged(RadioGroup group, int checkedId) {  
        switch (checkedId) {  
            case R.id.radio0:  
                ...  
                break;  
            ...  
            default:  
                ...  
        }  
    }  
});
```

Spinner I

► Lista de valores para desplegar (strings.xml)

```
<string-array name="array_paises">  
    <item>Andorra</item>  
    <item>Angola</item>  
    <item>Anguila</item> ...  
</string-array>
```



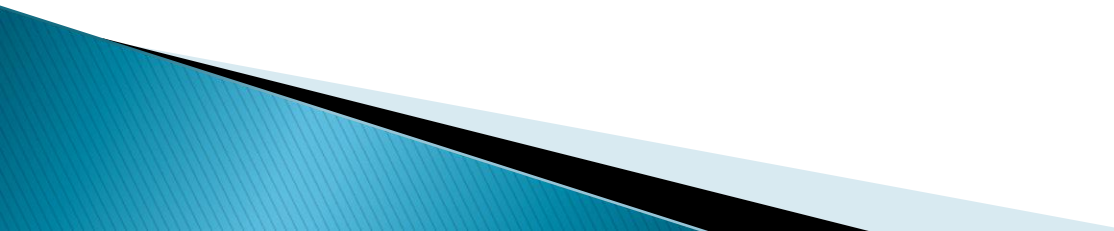
► Asociar la lista al spinner

```
sp = (Spinner) findViewById(R.id.spPaises);  
ArrayAdapter<CharSequence> adaptador =  
    ArrayAdapter.createFromResource(this,  
        R.array.array_paises, android.R.layout.simple_spinner_item);  
adaptador.setDropDownViewResource(android.R.layout.simple_spinner_dropdo  
wn_item);  
sp.setAdapter(adaptador);
```

Spinner II

- ▶ Responder a la selección de un elemento

```
sp.setOnItemSelectedListener(new  
    OnItemSelectedListener() {  
        @Override  
        public void onItemSelected(AdapterView<?> parent,  
            View view, int position, long id) { ...  
        }  
        @Override  
        public void onNothingSelected(  
            AdapterView<?> parent) { ...  
        }  
    });
```



TimePickerDialog

```
tpd = new TimePickerDialog(this,  
                           new ObtenerHora(), 23, 59, true);  
tpd.show();
```

```
public class ObtenerHora implements  
    OnTimeSetListener {  
    @Override  
    public void onTimeSet(TimePicker tp, int hour, int  
        minute){...  
    }  
}
```



DatePickerDialog

```
dpd = new DatePickerDialog(this,  
                           new ObtenerFecha(), 2013, 11, 31);  
dpd.show();  
public class ObtenerFecha implements  
                                   OnDateSetListener{  
    @Override  
    public void onDateSet(DatePicker view, int year,  
                           int monthOfYear,int dayOfMonth) {  
  
    }  
}
```



Fecha y hora

```
Calendar c = Calendar.getInstance();  
int hora = c.get(Calendar.HOUR_OF_DAY);  
int minuto = c.get(Calendar.MINUTE);  
int ano = c.get(Calendar.YEAR);  
int mes = c.get(Calendar.MONTH); // 0 a 11  
int dia = c.get(Calendar.DAY_OF_MONTH);
```


TextView. ImageView. Toast

```
<ImageView android:src="@drawable/imagen">
```

```
img = (ImageView)findViewById(R.id.ImgFoto);  
img.setImageResource(R.drawable.icono);
```

```
<TextView android:text="@string/idcadena" >
```

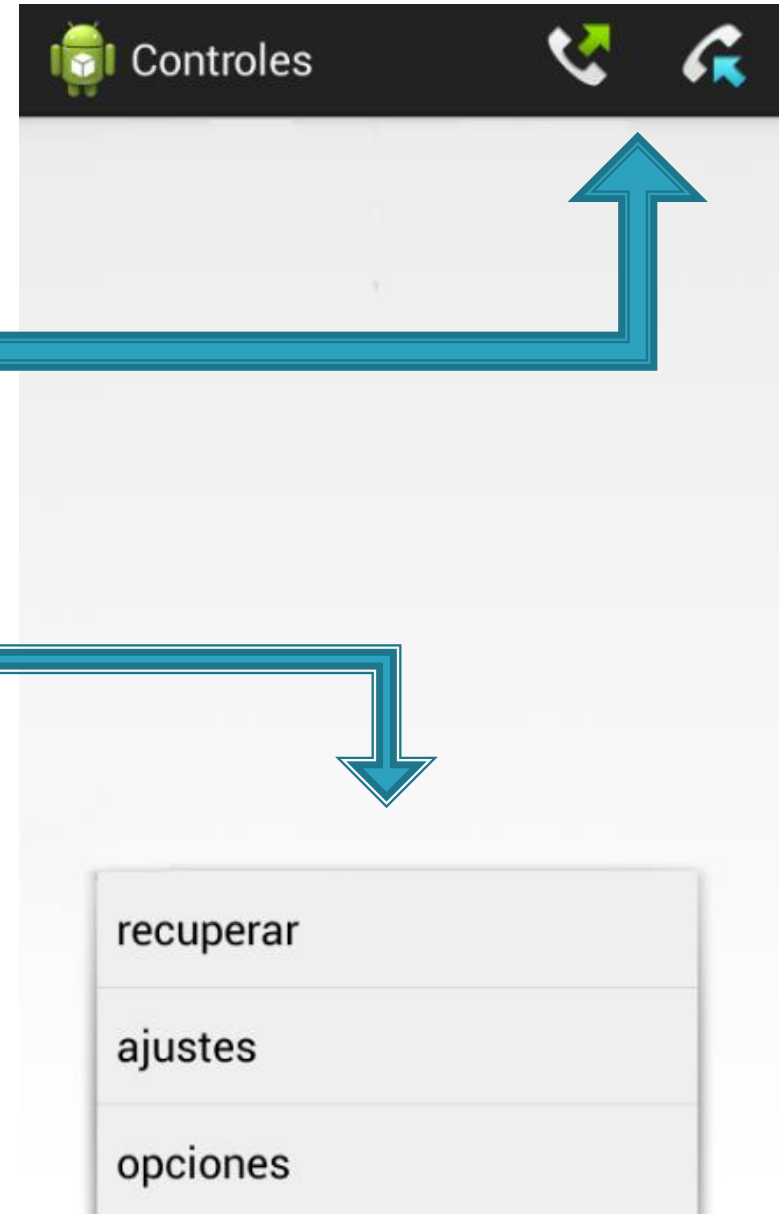
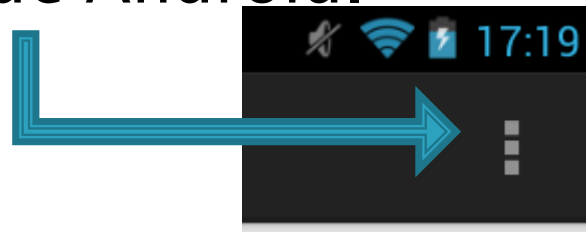
Métodos **getText().toString()** y **setText()**.

```
Toast.makeText(contexto, texto, duracion).show();
```



Menu I

- ▶ Los elementos gráficos de los menús se muestran sobre la barra de acción (ActionBar).
- ▶ Los elementos no gráficos de los menús se muestran en la parte inferior al pulsar la tecla menú o en la barra de acción, dependiendo de la versión de Android.



Menu II

- ▶ Los menús se definen en archivos XML de la carpeta /res/menu.
- ▶ Para crear un menú se utilizan las etiquetas <menu>, <item> y <group>.
- ▶ La apariencia final de los menús dependen de la versión de Android y del dispositivo.

```
<menu xmlns:android="http://schemas.android.com/apk/res/android" >  
  <item android:id="@+id/action_settings"  
    android:showAsAction="never"   
    android:title="@string/action_ajustes"/>  
  <item android:id="@+id/action_nuevo"  
    android:icon="@android:drawable/sym_call_outgoing"  
    android:title="@string/action_llamada"  
    android:showAsAction="ifRoom|withText"/>  
</menu>
```

No sale nunca en la barra de accion

Solo sale si hay espacio en la barra

Que aparezca el android:title

Always para que lo muestre siempre en la barra de

Menu III

Crear menú en una Activity.

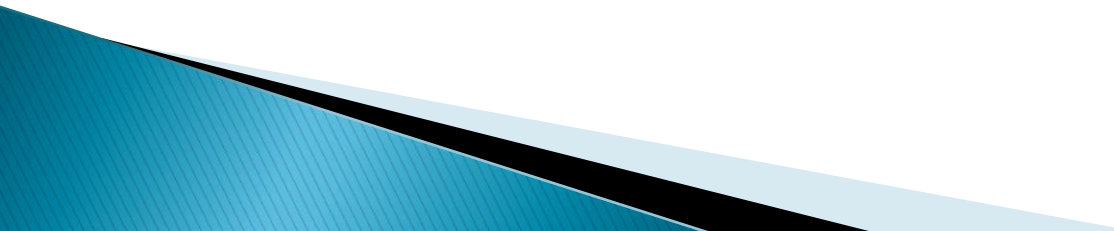
```
@Override
public boolean onCreateOptionsMenu
                        (Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.game_menu,
                        menu);

    return true;
}
```

Responder a la selección del menú.

```
@Override
public boolean onOptionsItemSelected
                        (MenuItem item) {
    switch (item.getItemId()) {
        case R.id.action_settings:
            ...
            return true;
        case R.id.action_nuevo:
            ...
            return true;
        default:
            return
                super.onOptionsItemSelected(item);
    }
}
```

ContextMenu

- ▶ Los menús contextuales son menús asociados a elementos de tipo vista de la aplicación.
 - ▶ Hay dos tipos de menús contextuales:
 1. Menús contextuales flotantes, asociados a un elemento, se muestra un menú central al hacer clic largo.
 2. Menús ActionMode, asociados a un elemento, se muestra una nueva ActionBar al hacer clic largo.
- 

ContextMenu

```
registerContextMenu(vista);
```

```
//se registra el control para poder recibir un menú contextual, onCreate
```

```
@Override //se sobrescribe este método para generar el menú  
public void onCreateContextMenu(ContextMenu menu, View v, ContextMenuInfo menuInfo) {  
    super.onCreateContextMenu(menu, v, menuInfo);  
    MenuInflater inflater = getMenuInflater();  
    inflater.inflate(R.menu.menu_contextual, menu);  
}
```

```
@Override //método para responder al item seleccionado  
public boolean onOptionsItemSelected(MenuItem item) {  
    AdapterContextMenuInfo info = (AdapterContextMenuInfo) item.getMenuInfo();  
    switch (item.getItemId()) {  
        case R.id.action_settings:  
            return true;  
        ...  
        default:  
            return super.onOptionsItemSelected(item);  
    }  
}
```

ActionMode (Api 11)

```
private ActionMode.Callback respuestaMenu = new ActionMode.Callback() {  
    @Override  
    public boolean onCreateActionMode(ActionMode mode, Menu menu) {  
        MenuInflater inflater = mode.getMenuInflater();  
        inflater.inflate(R.menu.menu_contextual, menu);  
        return true;  
    }  
    @Override  
    public boolean onPrepareActionMode(ActionMode mode, Menu menu) {  
        return false;  
    }  
    @Override  
    public boolean onActionItemClicked(ActionMode mode, MenuItem item) {  
        switch (item.getItemId()) {  
            case R.id.action_nuevo:  
                mode.finish();  
                return true;  
            default:  
                return false;  
        }  
    }  
    @Override  
    public void onDestroyActionMode(ActionMode mode) {  
        modoAccion = null;  
    }  
};
```

- ▶ El objeto respuestaMenu será el encargado de mostrar el menú y responder a las acciones que se produzcan sobre él.
- ▶ Al control ActionMode se le asocia un oyente LongClick que sustituye la barra de acción por una nueva.

```
private ActionMode modoAccion;  
  
vista.setOnLongClickListener(new View.OnLongClickListener() {  
    public boolean onLongClick(View view) {  
        if (modoAccion != null) {  
            return false;  
        }  
        modoAccion = ClaseActividad.this.startActionMode(respuestaMenu);  
        view.setSelected(true);  
        return true;  
    }  
});
```



El objeto modoAccion representa la nueva barra de acción.

PopupMenu (Api 11)

- ▶ Un menu Popup es un menú modal anclado sobre un control de tipo View.

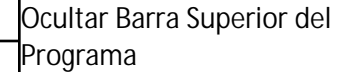
```
<ImageView  
    android:id="@+id/ivImagen"  
    android:onClick="popup"  
    android:src="@drawable/ic_icono" />
```

```
public void popup(View v) {  
    PopupMenu popup = new PopupMenu(this, v);  
    MenuInflater inflater = popup.getMenuInflater();  
    inflater.inflate(R.menu.menu_contextual, popup.getMenu());  
    popup.show();  
}
```



ActionBar (Api 11)

Ocultar Barra Superior del Programa



- ▶ Formas de ocultarla:
- ▶ Desde el manifiesto:

```
<activity  
  android:theme="@android:style/Theme.Holo.  
  NoActionBar">
```
- ▶ Desde la actividad:

```
ActionBar actionBar = getActionBar();  
actionBar.hide(); //show()
```
- ▶ Para usar el icono de la aplicación se debe referenciar: `android.R.id.home`.

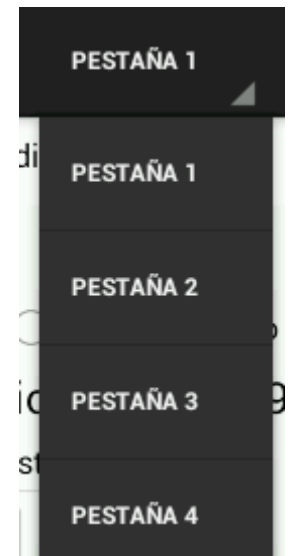
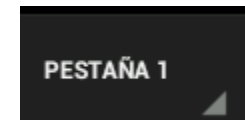
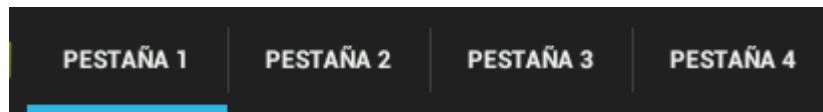
android.R.id.home

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case android.R.id.home:
            Intent intent = new Intent(this, ActividadInicial.class);
            intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
            startActivity(intent);
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}
```

De este modo se vuelve a la actividad de la aplicación, cerrando todas las actividades que hubiera abiertas. A partir de la API 14 se debe habilitar este icono: `actionbar.setHomeButtonEnabled(true)`.

Tab

- ▶ Las pestañas de navegación son un recurso que permite mostrar dentro de una actividad múltiples diseños.
- ▶ Este recurso se utiliza conjuntamente con los fragmentos.



Dialog: DialogFragment y AlertDialog

- ▶ Para crear cuadros de diálogo se utiliza AlertDialog.
- ▶ Se utiliza DialogFragment como contenedor de un AlertDialog.
- ▶ De forma predeterminada los diálogos tienen un título, un mensaje y hasta tres botones.
- ▶ Se puede convertir en modal usando el método `setCancelable(false)`.
- ▶ El uso de ProgressDialog está desaconsejado, es preferible usar ProgressBar.

`ProgressDialog.show(this, "título",
"mensaje"); // desaconsejado`

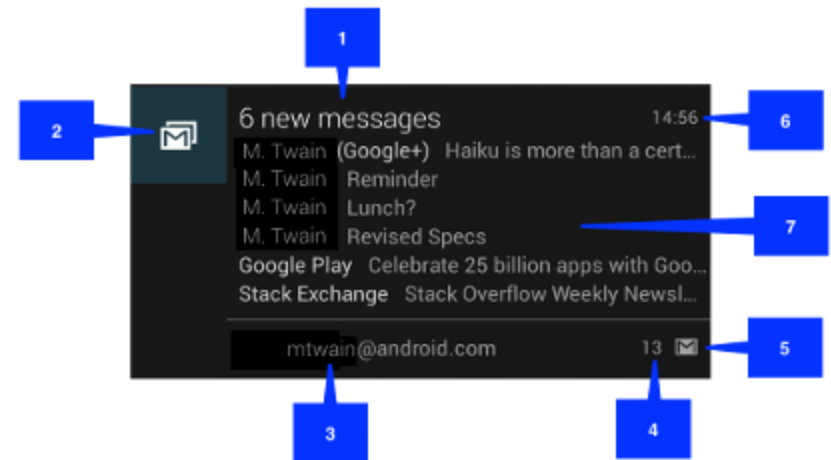
```
public class Dialogo extends DialogFragment {
    @Override
    public Dialog onCreateDialog(Bundle savedInstanceState) {
        AlertDialog.Builder adb = new AlertDialog.Builder(getActivity());
        adb.setMessage("mensaje")
            .setPositiveButton("positivo", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                    dismiss();
                }
            })
            .setNeutralButton("neutral", null)
            .setNegativeButton("negativo", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {}
            });
        return adb.create();
    }
}
```

```
Dialogo d=new Dialogo();
FragmentManager fm = this.getFragmentManager();
// this.getSupportFragmentManager()
d.show(fm, "dialogo"); //tag
```

← Compatibilidad

Notification I

- ▶ 1 título
- ▶ 2 icono grande
- ▶ 3 texto
- ▶ 4 información sobre el contenido
- ▶ 5 icono pequeño
- ▶ 6 hora de la notificación
- ▶ 7 área detallada



Notification II

```
Notification.Builder constructorNotificacion = new  
Notification.Builder(this)  
    .setSmallIcon(R.drawable.icono)  
    .setContentTitle("notificación")  
    .setContentText("texto")  
    .setContentIntent(PendingIntent.getActivity(getApplicationContext(),  
        0, new Intent(), 0));  
NotificationManager gestorNotificacion = (NotificationManager)  
    getSystemService(Context.NOTIFICATION_SERVICE);  
gestorNotificacion.notify(1, constructorNotificacion.build());
```

Esta es la forma más simple de crear una notificación. Lo normal es que una notificación contenga además un contenido que se muestra al desplegar la notificación y que al hacer clic sobre ella se abra una actividad.

No se debe olvidar que este recurso **no** se debe usar desde la aplicación activa, está pensado para que sea usado por aplicaciones que se ejecutan en segundo plano.

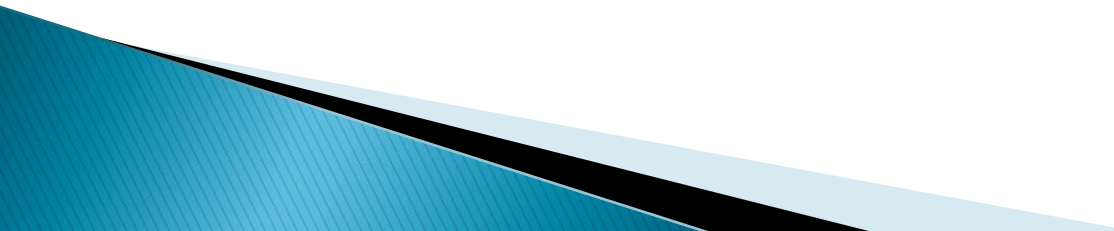
Log

- ▶ ADB (android debug bridge) proporciona un sistema de mensajes que ayuda a depurar una aplicación.

```
String LOGTAG = "HOLAMUNDO";  
Log.e(LOGTAG, "Mensaje de error");  
Log.w(LOGTAG, "Mensaje de warning");  
Log.i(LOGTAG, "Mensaje de información");  
Log.d(LOGTAG, "Mensaje de depuración");  
Log.v(LOGTAG, "Mensaje verbose");
```

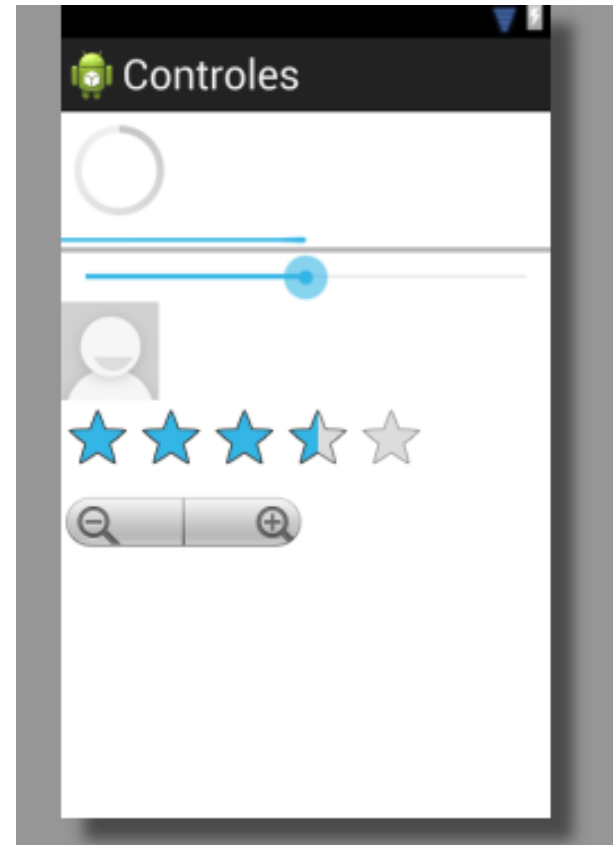
W	10-02 13:58:4...	629	629	es.izv.android	KeyCharact...	Using default keymap: /system/usr/keychars/qwer
E	10-02 13:59:0...	629	629	es.izv.android	HOLAMUNDO	Mensaje de error
W	10-02 13:59:0...	629	629	es.izv.android	HOLAMUNDO	Mensaje de warning
I	10-02 13:59:0...	629	629	es.izv.android	HOLAMUNDO	Mensaje de información
D	10-02 13:59:0...	629	629	es.izv.android	HOLAMUNDO	Mensaje de depuración
V	10-02 13:59:0...	629	629	es.izv.android	HOLAMUNDO	Mensaje verbose

Eventos de entrada

- ▶ `onClick`
 - ▶ `onLongClick`
 - ▶ `onCheckedChanged`
 - ▶ `onItemSelected`
 - ▶ `onOptionsItemSelected`
 - ▶ `onContextItemSelected`
 - ▶ `onActionItemClicked`
 - ▶ `onDateSet`, `onTimeSet`
 - ▶ `onFocusChange`, `onKey()`, `onTouch()`, etc.
- 

Etcetera

- ▶ ProgressBar
- ▶ SeekBar
- ▶ QuickContactBadge
- ▶ RatingBar
- ▶ ZoomControls
- ▶ Etc.



Enlaces

- ▶ <http://developer.android.com/guide/topics/ui/controls.html>
- ▶ Pestañas:
 - <http://developer.android.com/design/building-blocks/tabs.html>
 - <http://developer.android.com/training/implementing-navigation/lateral.html>
 - <http://stackoverflow.com/questions/14272125/android-tabhost-deprecated>
- ▶ Fragmentos:<http://developer.android.com/guide/components/fragments.html>
- ▶ Loaders:<http://developer.android.com/guide/components/loaders.html>
- ▶ Tasks:<http://developer.android.com/guide/components/tasks-and-back-stack.html>
- ▶ <http://developer.android.com/guide/components/processes-and-threads.html>