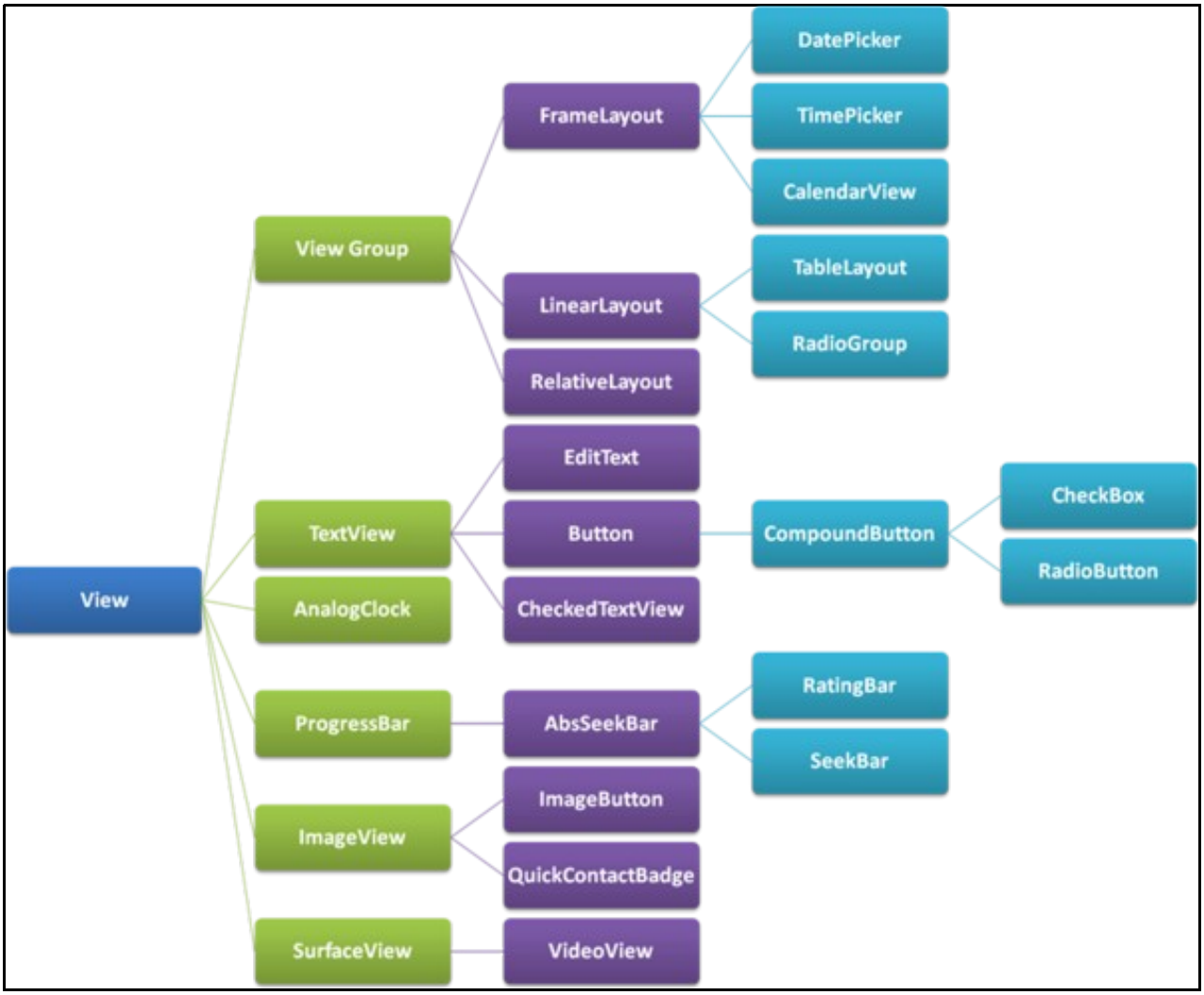
# EditText. Control de eventos I

## Sumario


- 1 Introdución
- 2 Casos prácticos
  - 2.1 Distintos tipos de datos: a propiedade android:inputType
  - 2.2 Control de eventos no teclado virtual. Distintas accións da tecla "Enter": android:imeOptions

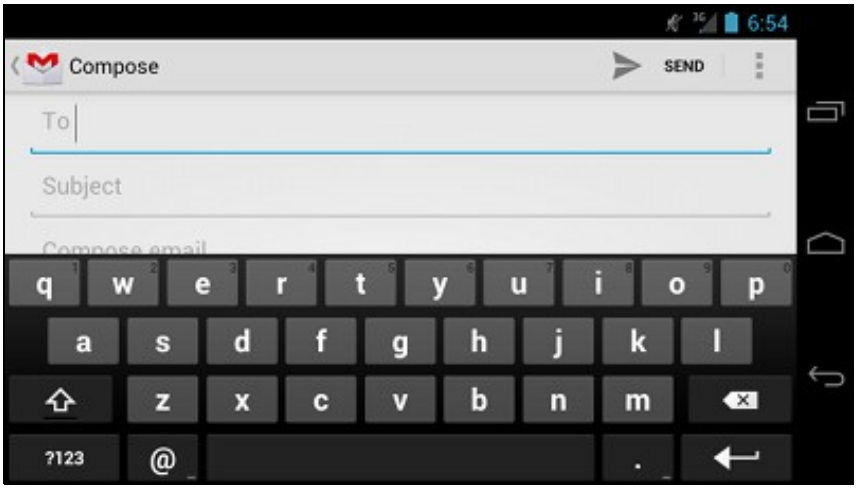

## Introdución

- O control **EditText** serve para introducir e editar texto por parte do usuario dunha aplicación Android.
- Este control é unha subclase da clase **TextView** que se viu anteriormente.



- Imaxe obtida de: http://www.itcsolutions.eu/2011/08/27/android-tutorial-4-procedural-vs-declarative-design-of-user-interfaces

- A forma de engadir un EditText en XML nun layout é a través do compoñente: <EditText/>
- Unha propiedade importante é: **android:inputType**
    - Permitirá que so se introduzan números, texto, passwords, teléfonos, etc.
    - Tamén indicará se o control é dunha soa liña ou multiliña.
    - Facer que non autocomplete as palabras, etc.
- Ademais permiten copiar, cortar e pegar.



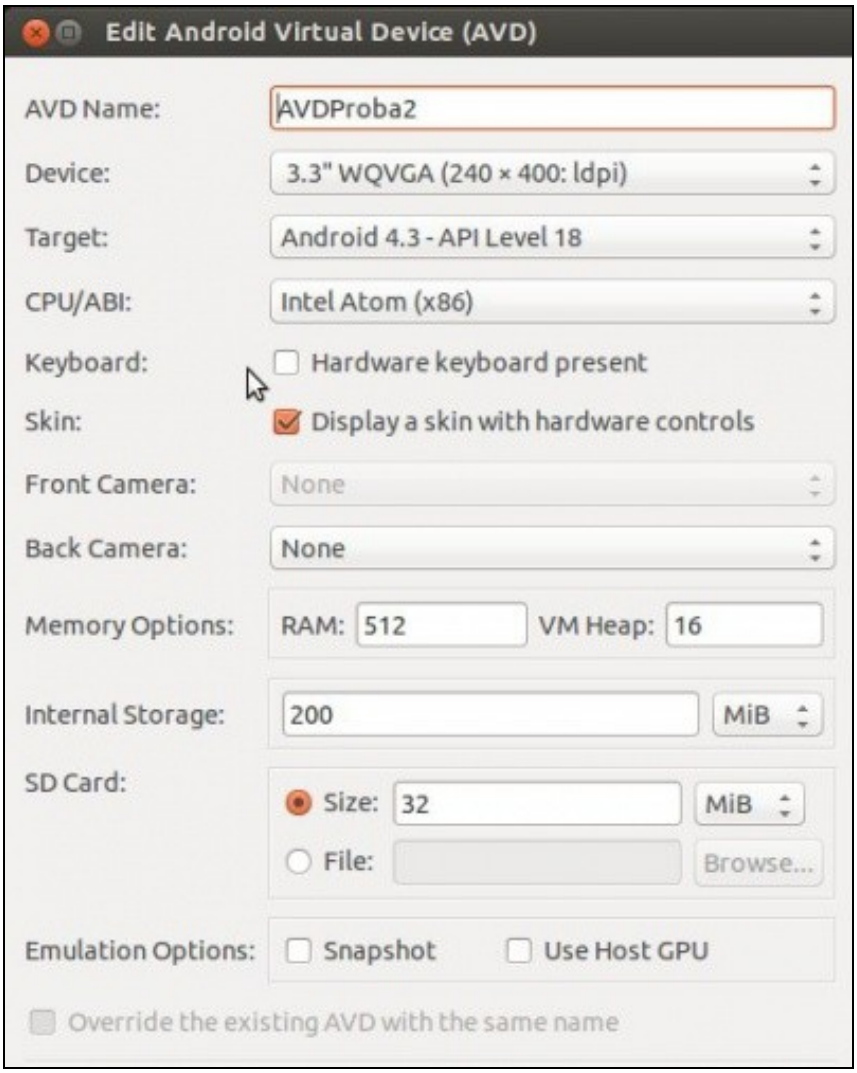

- Cando se entra nun cadro de texto, nun dispositivo que non teña teclado físico conectado, abrirase en pantalla un **teclado virtual** (tamén chamado lixeiro/soft) que se adaptará á propiedade **android:inputType** como veremos a continuación.

- Finalmente a tecla **Enter** no teclado virtual poder realizar distintas accións, ben por defecto ou ben explicitamente coa propiedade **android:imeOptions** (IME: Input Method Editor)

- **Referencias**:
    - O control EditText: http://developer.android.com/reference/android/widget/EditText.html
    - Campos de texto: http://developer.android.com/design/building-blocks/text-fields.html
    - Campos de texto para programadores: http://developer.android.com/guide/topics/ui/controls/text.html
    - Filtros de entrada (android:inputType): http://developer.android.com/reference/android/widget/TextView.html#attr_android:inputType
    - IME: http://developer.android.com/guide/topics/text/creating-input-method.html
    - android:imeOptions: http://developer.android.com/reference/android/widget/TextView.html#attr_android:imeOptions

- **IMPORTANTE**: Se se vai usar un AVD imos desactivar o teclado hardware para que cando premamos nun cadro de texto nos apareza o teclado virtual.

Lembrar que na imaxe aparece unha API 18 pero nos estamos traballando cunha API 16.


Edit Android Virtual Device (AVD)
AVD Name:
AVDProba2
Device:
3.3" WQVGA (240 × 400: ldpi)
Target:
Android 4.3 - API Level 18
CPU/ABI:
Intel Atom (x86)
Keyboard:
Hardware keyboard present
Skin:
Display a skin with hardware controls
Front Camera:
None
Back Camera:
None
Memory Options:
RAM: 512
VM Heap: 16
Internal Storage:
200
MiB
SD Card:
Size: 32
MiB
File:
Browse...
Emulation Options:
Snapshot
Use Host GPU
Override the existing AVD with the same name

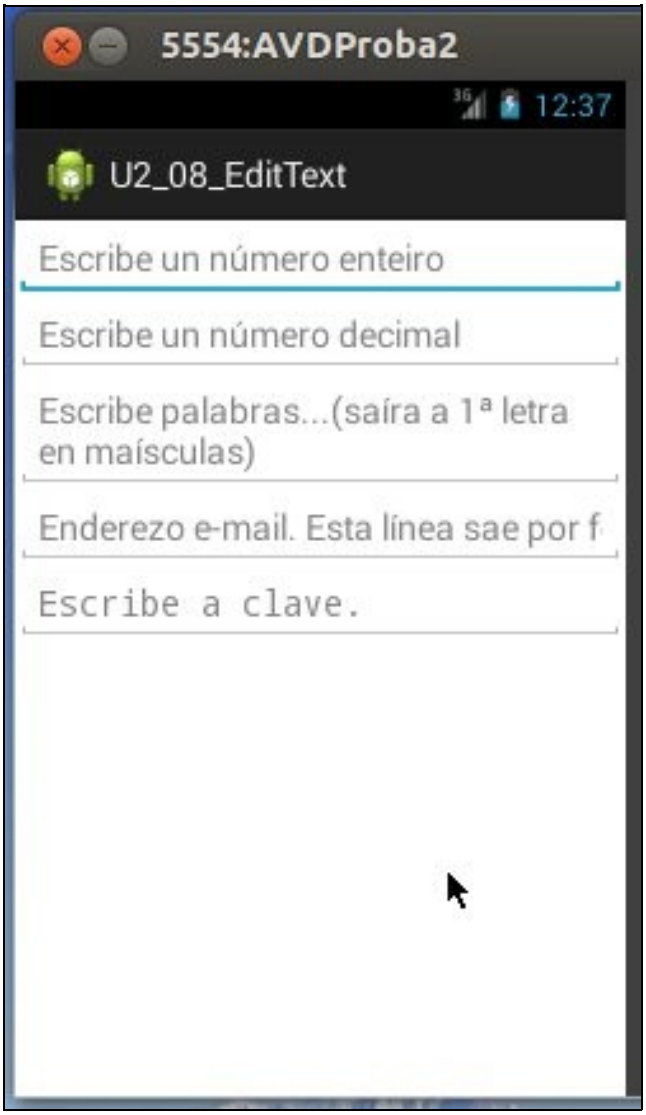
- Desactivar o teclado por hardware

## Casos prácticos

- Comezamos creando un proxecto no que realizar as probas: **U2_08_EditText**

### Distintos tipos de datos: a propiedade android:inputType

- A imaxe amosa un AVD con distintos tipos de campos, e cunha lenda interior indicando que tipo de dato hai que introducir nese campo.



- O layout xml que define esa pantalla é o seguinte. Observar como sempre as liñas marcadas.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Escribe un número enteiro"
        android:inputType="numberSigned" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Escribe un número decimal"
        android:inputType="numberSigned|numberDecimal" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Escribe palabras...(saíra a 1ª letra en maísculas)"
        android:inputType="textCapWords|textMultiLine" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enderezo e-mail. Esta línea sae por fóra da pantalla"
        android:inputType="textEmailAddress" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Escribe a clave."
        android:inputType="textPassword" />

</LinearLayout>
```

- Propiedade **android:hint**: indica o texto que se vai amosar no control cando este está baleiro.
- Propiedade **android:inputType**: indica o tipo de datos que se poden introducir no campo.
    - ♦ Se non se pon a propiedade, o campo acepta toda combinación de caracteres.
    - ♦ Pódense combinar os filtros facendo uso de **"|"**.
- Obviamente os valores dos Hints é mellor telos declarados en constantes en recursos XML.

- Lanzamos a aplicación nun AVD sen botonera hardware.
- '*NOTA Edición 2015****: No caso de usar un AVD con API 21, o botón*** *Next* ***é substituído por*** *">", o botón* *Done* (Feito) por unha marca de verificación.

- Distintos teclados

- 


Cando nos posicionamos no primeiro campo, como é para enteiros con signo (inputType), ofrece un teclado con só números, signo "-" e "." ou ",".
Probar a introducir un número decimal.
Observar tamén que a tecla **Enter** foi substituída por **Next**, para pasar ao seguinte campo. Pronto estudaremos esta tecla.

- 

O segundo campo xa admite números decimais e con signo. A tecla **Enter** segue marcando **Next**.

- 


O teclado xa contempla as letras e demais. Observar como de xeito automático pon en maiúsculas a primeira letra de cada palabra.

- 

Como o terceiro campo é multiliña a tecla de **Enter** é agora un retorno de carro para poder seguir escribindo varias liñas no campo.

- 


Situámonos co dedo (ou rato) no seguinte campo. Agora o teclado xa ofrece a @ para o mail. A tecla **Enter** segue sendo **Next**.

- 

E agora é o momento de introducir a password. Observar como a tecla **Enter** é agora **Done** (Feito). Porque este campo é último e xa non hai máis a continuación.

- 


Unha vez que se preme na tecla **Done** desaparece o teclado.

### Control de eventos no teclado virtual. Distintas accións da tecla "Enter": android:imeOptions

- A tecla **Enter** no teclado virtual pode realizar unha serie de accións:
  - ♦ Por defecto:
    - ◊ o sistema determina se existe outro elemento ao que poida ir o foco, nese caso a función da tecla é **Next**.
    - ◊ Se non o atopa a acción da tecla é **Done**.
    - ◊ Salvo nalgúns **inputType** como por exemplo o multiliña.
  - ♦ Podemos asignar explicitamente que acción se desexa que se leve a cabo cada caixa de texto. Para iso está o atributo **android:imeOptions**.
    - ◊ Que dun campo vaia ao anterior, que envíe o texto, que busque o texto, etc. En referencias está un enlace ás posibles accións.

- O seguinte XML vai e a mesma pantalla anterior, só que as accións da tecla **Enter** para algúns **EditText** foi cambiada. Observar as accións definidas nas liñas marcadas.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Escribe un número enteiro"
        android:imeOptions="actionDone"
        android:inputType="numberSigned" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Escribe un número decimal"
        android:imeOptions="actionPrevious"
        android:inputType="numberSigned|numberDecimal" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Escribe palabras...(saíra a 1ª letra en maísculas)"
        android:inputType="textCapWords|textMultiLine" />
```

```
    <EditText
        android:id="@+id/et_mail"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enderezo e-mail. Esta línea sae por fóra da pantalla"
        android:imeOptions="actionSend"
        android:inputType="textEmailAddress" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Escribe a clave."
        android:inputType="textPassword" />

</LinearLayout>
```

- Introducir datos
- 

O primeiro campo ten definida a acción **Done** para a tecla **Enter**.
- 

O segundo campo a acción ir ao anterior campo.

- 


O campo do mail, ten definida a acción **Send** que vai ser capturada en Java e realizar unha acción en consecuencia.

- Codificación en Java que captura a acción **actionSend** dun campo de texto (email) e o que fai e recoller o texto desa caixa de texto, o e-mail, e volver a imprimir unha mensaxe nesa mesma caixa de texto.

```
package com.example.u2_08_edittext;

import android.app.Activity;
import android.os.Bundle;
import android.view.KeyEvent;
import android.view.Menu;
import android.view.inputmethod.EditorInfo;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.TextView.OnEditorActionListener;

public class U2_08_EditText extends Activity {

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_u2_08_edit_text);


final EditText etMail = (EditText) findViewById(R.id.et_mail);

etMail.setOnEditorActionListener(new OnEditorActionListener() {
    @Override
    public boolean onEditorAction(TextView v, int actionId, KeyEvent event) {
        boolean handled = false;
        if (actionId == EditorInfo.IME_ACTION_SEND) {
            v.setText("Mensaxe enviada a: "+v.getText());
                 //etMail.setText("Mensaxe enviada a: " + etMail.getText());
            handled = true;
        }

        return handled;
    }
});

}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
// Inflate the menu; this adds items to the action bar if it is present.
```

```
getMenuInflater().inflate(R.menu.u2_08__edit_text, menu);
return true;
}

}
```

- **Liña 20**: creamos un obxecto (etMail) que apunta ao EditText no que se introduce o mail.
- **Liña 22 - 33**: chamamos a método Listener (*escoitador*) (que veremos máis adiante) do obxecto etMail: **setOnEditorActionListener ()**
    - Ese método será chamado cando se realice unha acción nun TextView
    - Lembrar que un EditText é unha subclase de TextView
    - Por exemplo, cando se prema unha tecla ou cando haxa unha acción IME seleccionada polo usuario.
    - Como parámetro váiselle pasar a creación dunha clase anónima que implementa unha interface (OnEditorActionListener) para a cal hai que sobreescribir o único método que ten a interface (onEditorAction()).

- **Liñas 24-29**: sobreescritura do método OnEditorAction() no que se reciben 3 parámetros:
    - O TextView que xerou o evento (v). Neste caso a caixa de texto do mail
    - O ID da acción enviada
    - E se o evento foi xerado pola tecla **Enter** ou non (event).
    - Comprobamos se o ID da acción é o SEND, nese caso modificamos o contido da caixa de texto do mail.

- **Liña 28**: realiza a mesma función ca 27. Lembrar que **v** é un TextView que apunta á caixa de edición do mail e que etMail é un EditText que apunta ao mesmo elemento.

- Para que o IDE nos cre a **clase anónima**:
    - Escribir a chamada ao método (ollo ; final incluído): **etMail.setOnEditorActionListener( );**
    - Escribir **new** entre os paréntesis: **etMail.setOnEditorActionListener(new );**
    - Premer CTRL+Barra espaciadora e xa o sistema completa todo o demais.
    - Logo só queda poñer o noso código no método a sobreescribir.

- Lembrar premer **SHIFT+CTRL+O** para importar os paquetes correspondentes.

- Acción Send
- 

Ao premer en **Send**

- 


O sistema captura o evento e envía unha mensaxe a o mesma caixa de texto.

-- Ángel D. Fernández González e Carlos Carrión Álvarez -- (2015).