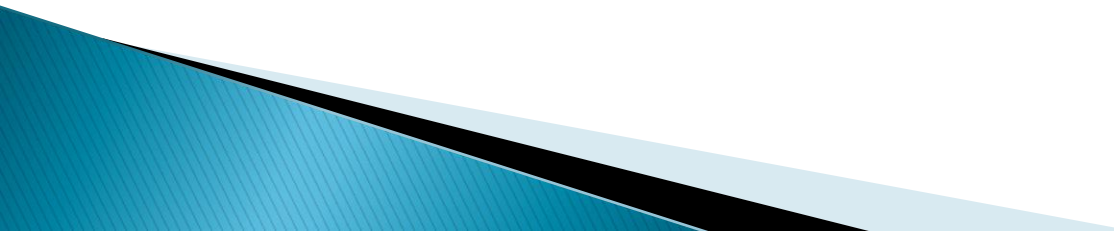


# Android: Fragment

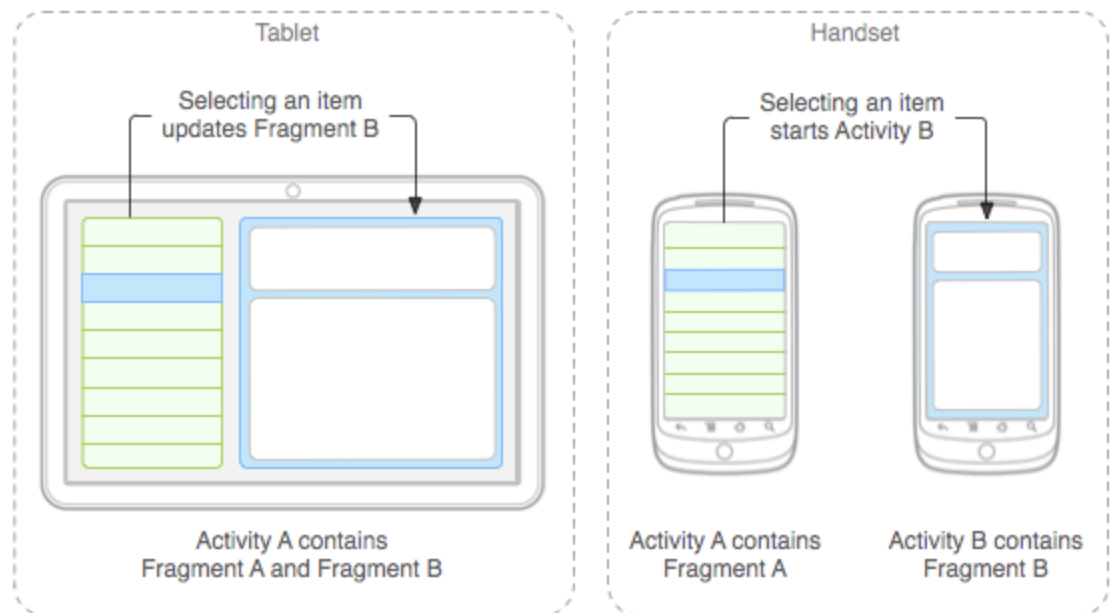
`android.app.Fragment`

# Fragmento

- ▶ Un fragmento representa un comportamiento o una parte de la interfaz de usuario en una Actividad. Se pueden combinar múltiples fragmentos en una sola actividad para construir una interfaz de usuario multipanel y reutilizar un fragmento en múltiples actividades. Un fragmento es como una sección modular de una actividad, que tiene su propio ciclo de vida y que recibe sus propios eventos.
- 

# Uso de fragmentos

- Los fragmentos existen a partir de Android 3.0, pero también es posible que funcione en versiones anteriores utilizando el paquete de compatibilidad.  
(`android.support.v4.app.FragmentActivity`)



# Creación de fragmentos I

- ▶ El diseño de un fragmento se crea en un layout XML del mismo modo que se crea el layout de una actividad.
- ▶ En el layout contenedor del fragmento se inserta una etiqueta `<fragment>`.
- ▶ `<fragment`  
    `android:id="@+id/idFragmento"`  
    `android:layout_width="..."`  
    `android:layout_height="..."`  
    `android:name="com.izv.app.ClaseFragmento">`  
`</fragment>`

# Creación de fragmentos II

- ▶ El atributo `android:name` hace referencia a la clase Java que implementa la funcionalidad del fragmento. Esta clase extiende la clase `android.app.Fragment` y debe implementar al menos el método `onCreateView()`. Este método es equivalente al método `onCreate()` de la clase `Activity`.

# Creación de fragmentos III

[illegible]



onAttach()

onCreate()

onCreateView()

onActivityCreated()

onStart()

onResume()

Fragment is active



User navigates backward or fragment is removed/replaced

The fragment is added to the back stack, then removed/replaced

onPause()

onStop()

onDestroyView()

onDestroy()

onDetach()

Fragment is destroyed

The fragment returns to the layout from the back stack

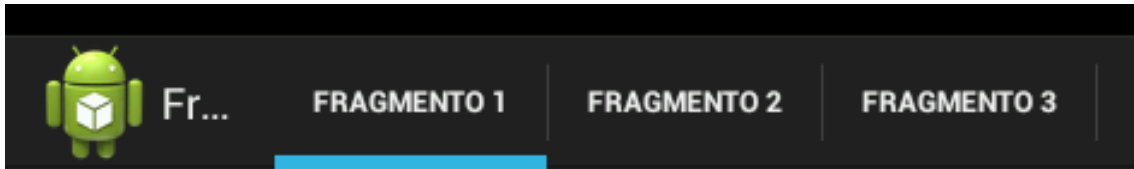
# OnClickListener

- ▶ Si el fragmento contiene algún botón para el que se vaya a implementar el evento `onClick` **no** se puede usar el atributo `android:onClick`.
- ▶ En su lugar se debe implementar una clase `Listener`.

```
Button bt = (Button) view.findViewById(R.id.boton);  
bt.setOnClickListener(new View.OnClickListener() {  
    public void onClick(View v) {  
        ...  
    }  
});
```



# Pestañas y fragmentos



- ▶ En la barra de acción, ActionBar, se pueden integrar pestañas de navegación y mostrar un fragmento diferente al pulsar sobre ellas.
- ▶ Los fragmentos se cargan de forma dinámica.



# Habilitar la ActionBar

- ▶ 

```
ActionBar ab = getActionBar();  
ab.setNavigationMode(  
    ActionBar.NAVIGATION_MODE_TABS);  
ab.show();
```
- ▶ La barra de acción la activamos en el método `onCreate()` del layout principal que va a contener los fragmentos.

# Contenedor para los fragmentos

En el layout principal se inserta un ViewGroup que va a hacer de contenedor del fragmento.

```
<Layout>
```

```
  <Layout android:id="@+id/frag_contenedor">
```

```
    </Layout>
```

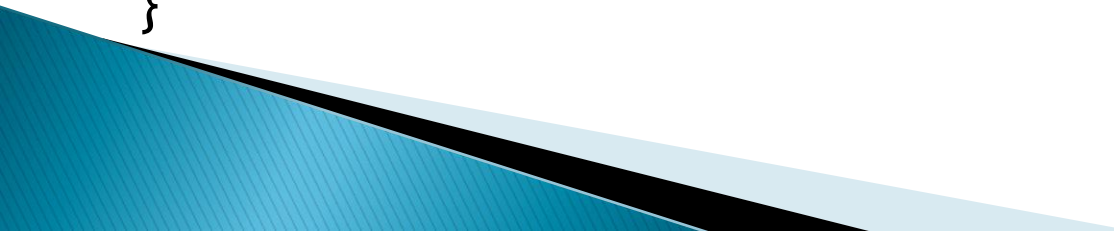
```
</Layout>
```



# Layout y clase Fragment

- ▶ Diseñamos los layouts de los fragmentos igual que los layouts de las actividades.
- ▶ Creamos la clase Fragment que visualice el layout.

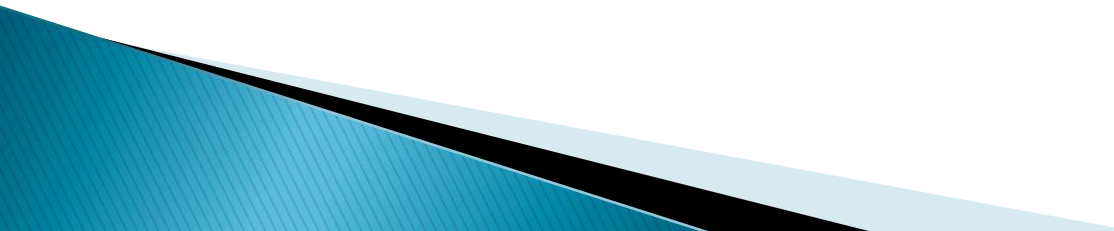
```
public class FragUno extends Fragment{
    @Override
    public View onCreateView(LayoutInflater inf,
                           ViewGroup contenedor, Bundle estado) {
        View v = inflater.inflate(R.layout.frag1, contenedor,
                                false);
        return v;
    }
}
```



# Agregar la pestaña al layout principal

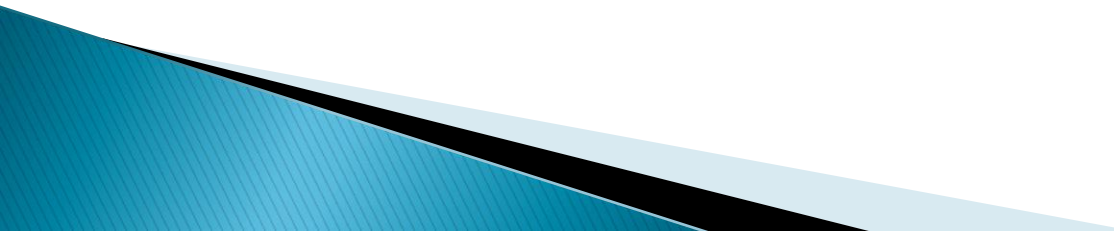
```
Tab pes1 = ab.newTab().setText("Frag 1");  
Fragment f1 = new FragUno ();  
pes1.setTabListener(new Oyente(f1));  
ab.addTab(pes1);
```

El Oyente es una clase Listener que es la que se encarga de mostrar el fragmento indicado al pulsar sobre la pestaña.

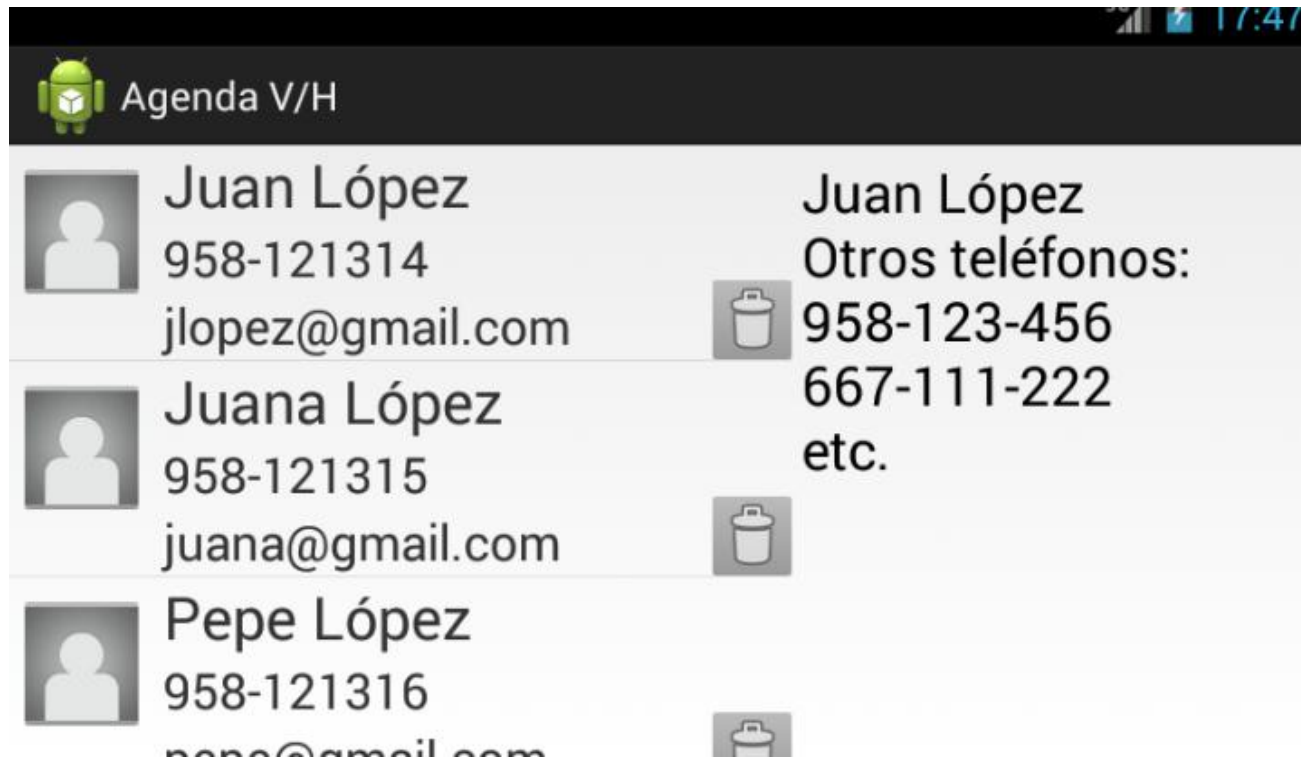


# Clase Listener

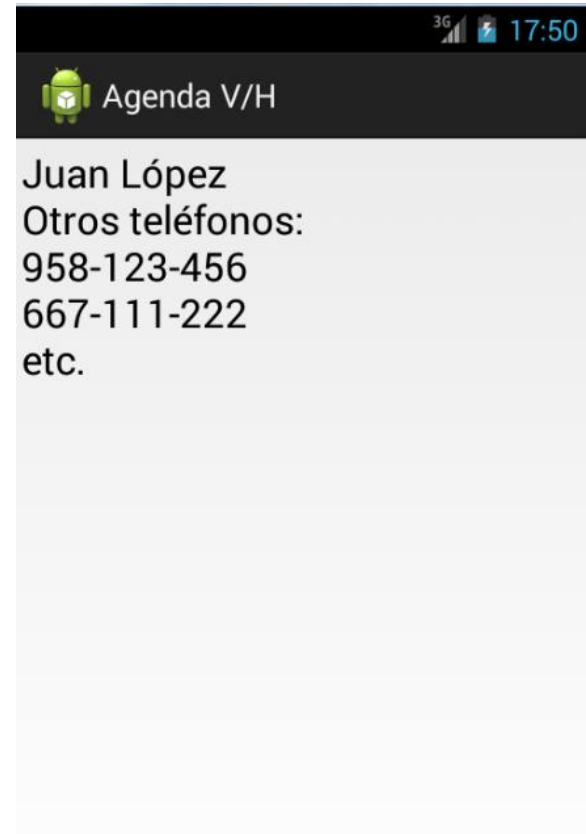
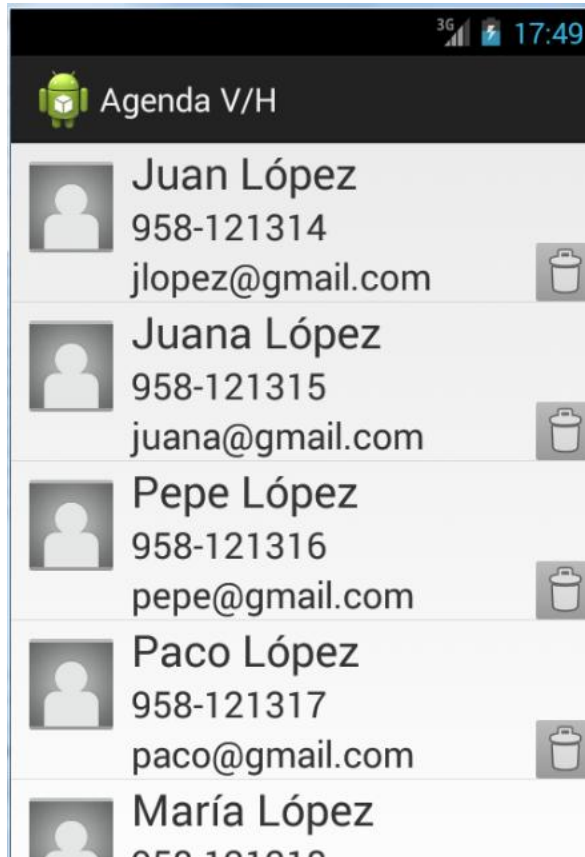
```
private class Oyente implements TabListener {
    private Fragment f;
    public Oyente(Fragment f) {
        super();
        this.f = f;
    }
    @Override
    public void onTabReselected(Tab tab, FragmentTransaction ft) {
    }
    @Override
    public void onTabSelected(Tab tab, FragmentTransaction ft) {
        ft.replace(R.id.frag_contenedor, f);
    }
    @Override
    public void onTabUnselected(Tab tab, FragmentTransaction ft) {
        ft.remove(f);
    }
}
```



# Orientaciones y fragmentos

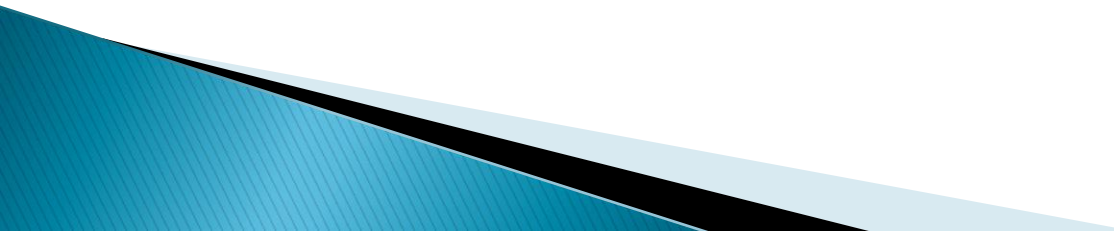


# Orientación vertical/horizontal



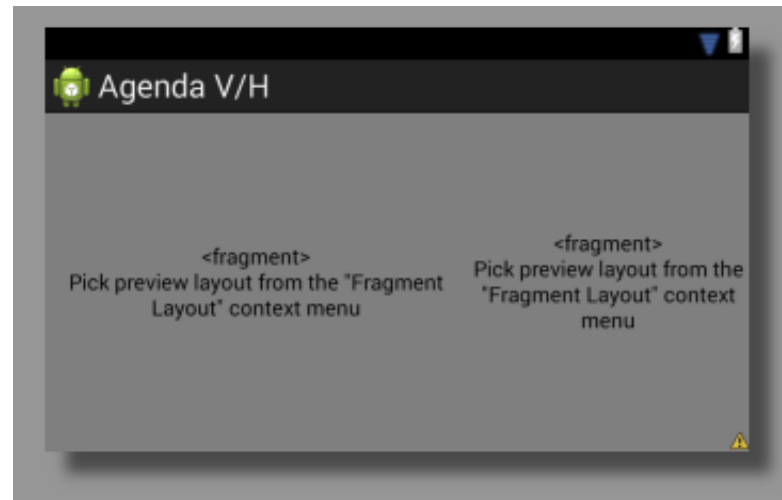
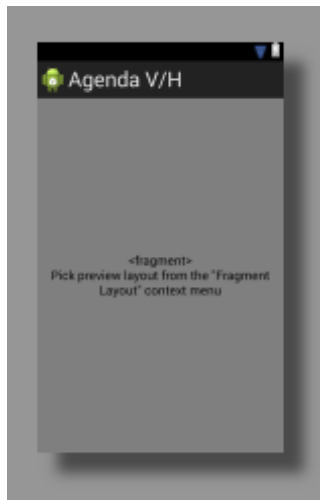


# Reutilizar recursos


- ▶ Vertical: la aplicación está basada en dos layouts, en el primero se muestra un fragmento y en el segundo el otro fragmento.
  - ▶ Horizontal: la aplicación se basa en un layout (landscape) que muestra dos fragmentos.
  - ▶ Se utiliza para ambas orientaciones los mismos recursos.
  - ▶ Se aprovecha también la mayor parte de la programación.
- 

# Layouts de la aplicación

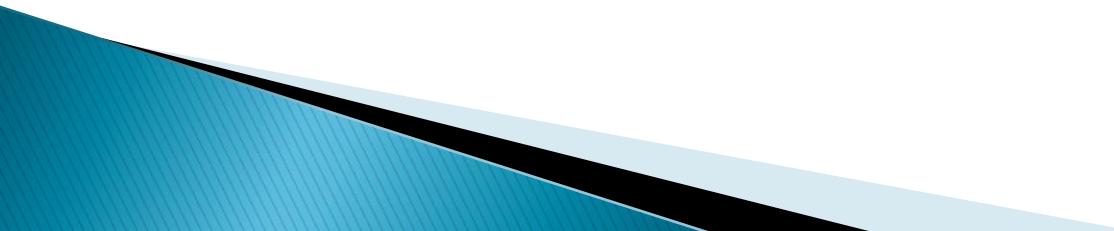
- ▶ Layout principal vertical
- ▶ Layout secundario vertical
- ▶ Layout principal horizontal (landscape)



# Fragmentos de la aplicación

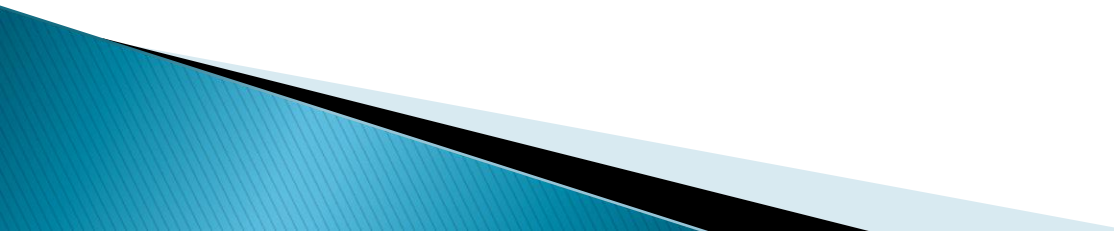
- ▶ Fragmento principal, contiene el ListView.
  - ▶ Fragmento secundario, contiene el TextView para mostrar la información adicional de cada uno de los elementos.
  - ▶ Puesto que el fragmento principal tiene un ListView, habrá que crear un layout adicional para mostrar cada uno de los elementos del ListView.
  - ▶ Total: 4 layouts y 2 fragmentos.
- 

# Programación

- ▶ Una sola Activity para los dos layouts principales.
  - ▶ Un Activity para el layout secundario.
  - ▶ Dos clases de tipo Fragment para los dos fragmentos.
  - ▶ La clase Contacto y la clase Adaptador.
- 

# Implementación de los fragmentos

```
public class Fragmento extends Fragment {  
    @Override  
    public View onCreateView(LayoutInflater i,  
                             ViewGroup vg, Bundle b) {  
        View v = i.inflate(R.layout.fragmento, vg,  
                           false);  
        return v;  
    }  
}
```



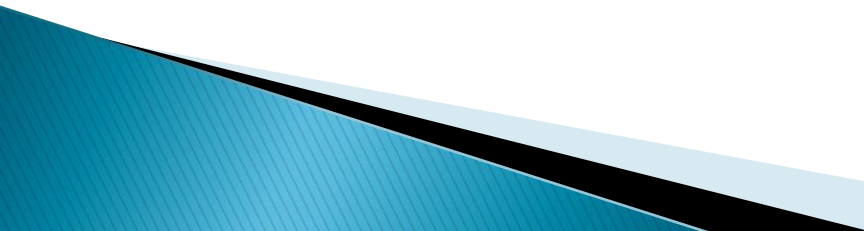
# Fragmento secundario

- ▶ En el fragmento secundario implementamos un método para el TextView.

```
public void setText(String s){  
    TextView tv = (TextView) getView().  
                    findViewById(R.id.tvTexto);  
    tv.setText(s);  
}
```

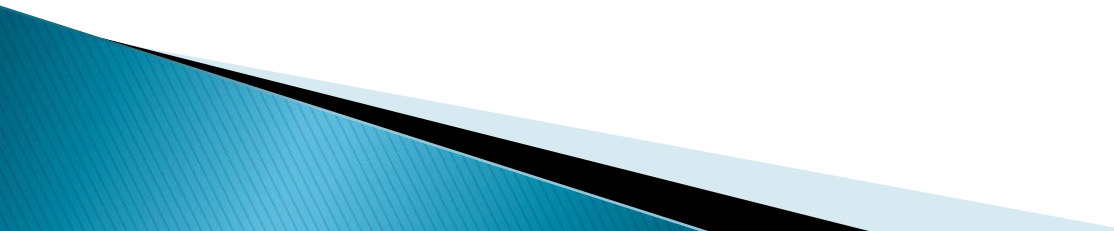
# Actividad principal

```
public class Actividad extends Activity{  
    private FragmentoDetalle fd;  
    @Override  
    protected void onCreate(Bundle b) {  
        super.onCreate(b);  
        setContentView(R.layout.actividad_primera);  
        fd = (FragmentoDetalle)  
            getSupportFragmentManager().  
            findFragmentById(R.id.fragmento);  
    }  
}
```



# OnItemClickListener del ListView

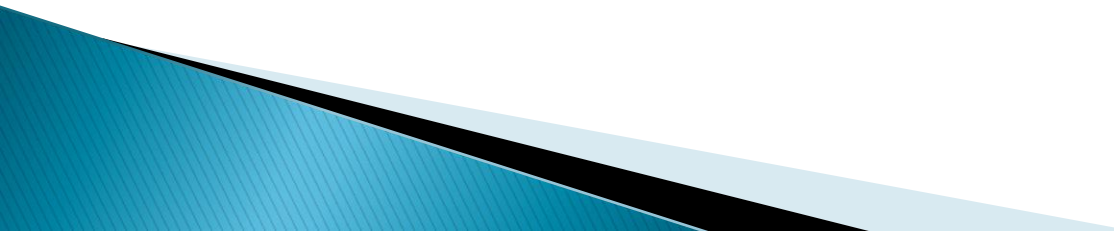
```
if (fd != null && fd.isInLayout()) {  
    fd.setText(texto);  
}  
else{  
    Intent i = new Intent(getApplicationContext(),  
                                Actividad2.class);  
    i.putExtra("valor", texto);  
    startActivity(i);  
}
```





# Actividad secundaria

```
public class Actividad2 extends Activity{
@Override
    protected void onCreate(Bundle b) {
        super.onCreate(b);
        setContentView(R.layout.actividad_segunda);
        Bundle e = getIntent().getExtras();
        String s = e.getString("datos");
        TextView tv = (TextView) findViewById
                                (R.id.tvTexto);
        tv.setText(s);
    }
}
```



- ▶ <http://developer.android.com/guide/components/fragments.html>
  - ▶ <http://developer.android.com/guide/topics/ui/actionbar.html>
  - ▶ <http://developer.android.com/training/implementing-navigation/lateral.html>
- 