NOVEDADES EN HONEYCOMB

Bienvenidos.

GTUG MURCIA

NOVEDADES EN HONEYCOMB

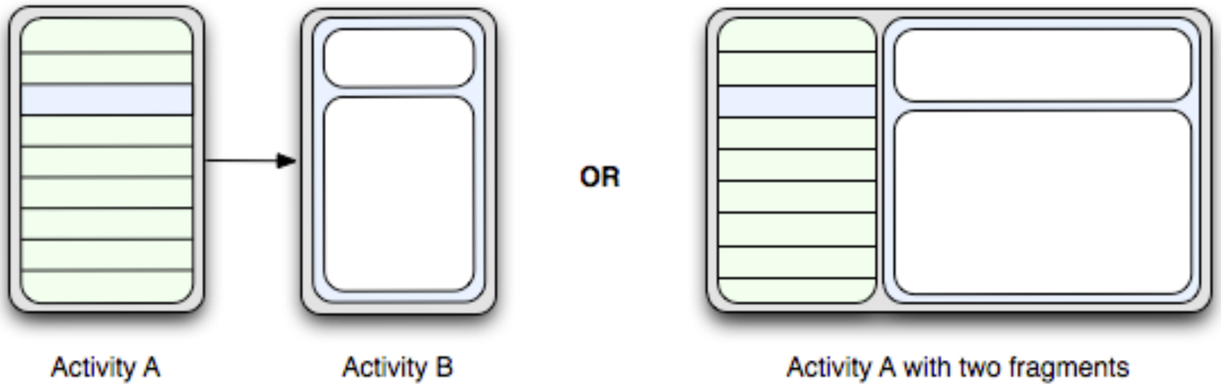
Lo que vamos a ver.

- ✧ Fragment.
- ✧ Loaders.
- ✧ ActionBar.
- ✧ Drag&Drop.
- ✧ Animaciones.
- ✧ Android Compability Package.

NOVEDADES EN HONEYCOMB

Fragments.

1 Nueva filosofía de diseño de la ACTIVITY.

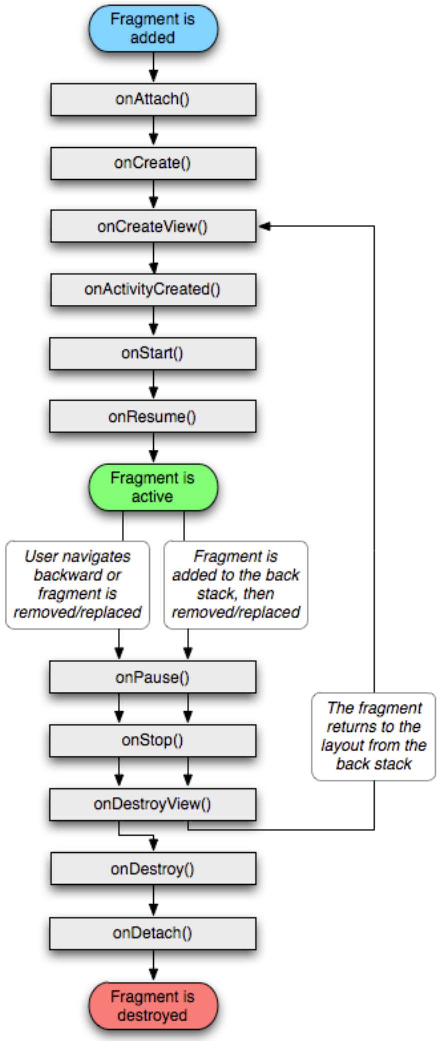


NOVEDADES EN HONEYCOMB

Fragments.

2 Ciclo de vida de un Fragment.

- ✧ onAttach()
- ✧ onCreateView()
- ✧ onActivityCreated()
- ✧ onDestroyView()
- ✧ onDetach()



NOVEDADES EN HONEYCOMB

Fragments.

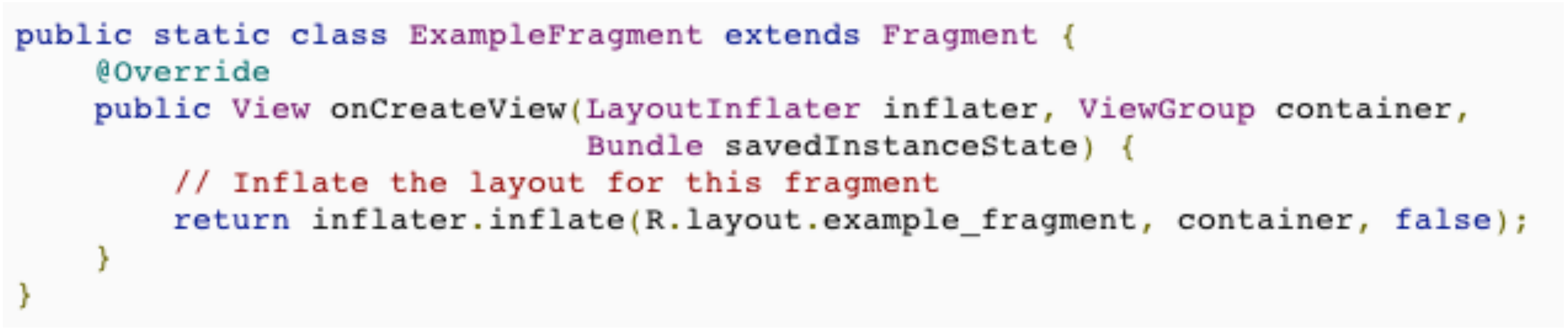
3 Principales Fragments.

- ✧ Fragment
- ✧ DialogFragment
- ✧ ListFragment
- ✧ PreferenceFragment

NOVEDADES EN HONEYCOMB

Fragments.

4 Creando un Fragment.



```
public static class ExampleFragment extends Fragment {
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.example_fragment, container, false);
    }
}
```

NOVEDADES EN HONEYCOMB

Fragments.

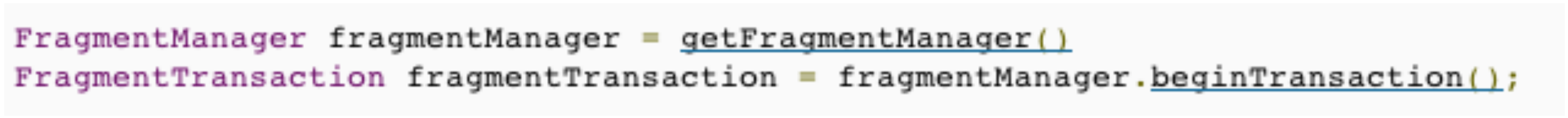
5 Añadiendo nuestro Fragment a una Activity I.

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <fragment android:name="com.example.news.ArticleListFragment"
            android:id="@+id/list"
            android:layout_weight="1"
            android:layout_width="0dp"
            android:layout_height="match_parent" />
    <fragment android:name="com.example.news.ArticleReaderFragment"
            android:id="@+id/viewer"
            android:layout_weight="2"
            android:layout_width="0dp"
            android:layout_height="match_parent" />
</LinearLayout>
```

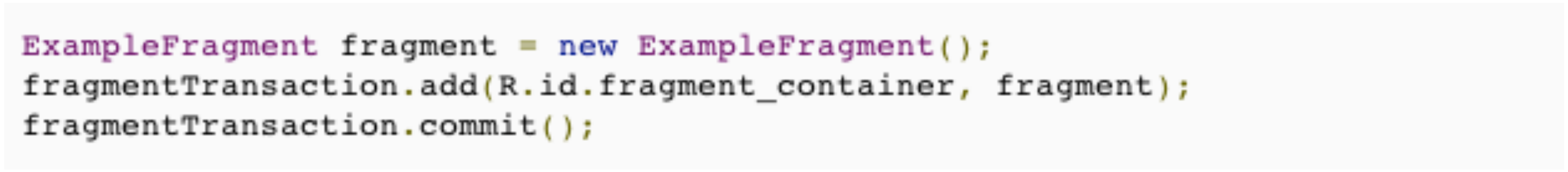
NOVEDADES EN HONEYCOMB

Fragments.

6 Añadiendo nuestro Fragment a una Activity II.



```
FragmentManager fragmentManager = getFragmentManager()
FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();
```



```
ExampleFragment fragment = new ExampleFragment();
fragmentTransaction.add(R.id.fragment_container, fragment);
fragmentTransaction.commit();
```

NOVEDADES EN HONEYCOMB

Fragments.

7 Manejando nuestros Fragments.

```
// Create new fragment and transaction
Fragment newFragment = new ExampleFragment();
FragmentTransaction transaction = getFragmentManager().beginTransaction();

// Replace whatever is in the fragment_container view with this fragment,
// and add the transaction to the back stack
transaction.replace(R.id.fragment_container, newFragment);
transaction.addToBackStack(null);

// Commit the transaction
transaction.commit();
```

NOVEDADES EN HONEYCOMB

Fragments.

8 Hablando con la Activity.

```
ExampleFragment fragment = (ExampleFragment) getFragmentManager().findFragmentById(R.id.example_fragment);
```

```
public static class FragmentA extends ListFragment {
    OnArticleSelectedListener mListener;
    ...
    @Override
    public void onAttach(Activity activity) {
        super.onAttach(activity);
        try {
            mListener = (OnArticleSelectedListener) activity;
        } catch (ClassCastException e) {
            throw new ClassCastException(activity.toString() + " must implement OnArticleSelectedListener");
        }
    }
    ...
}
```

NOVEDADES EN HONEYCOMB

Loaders.

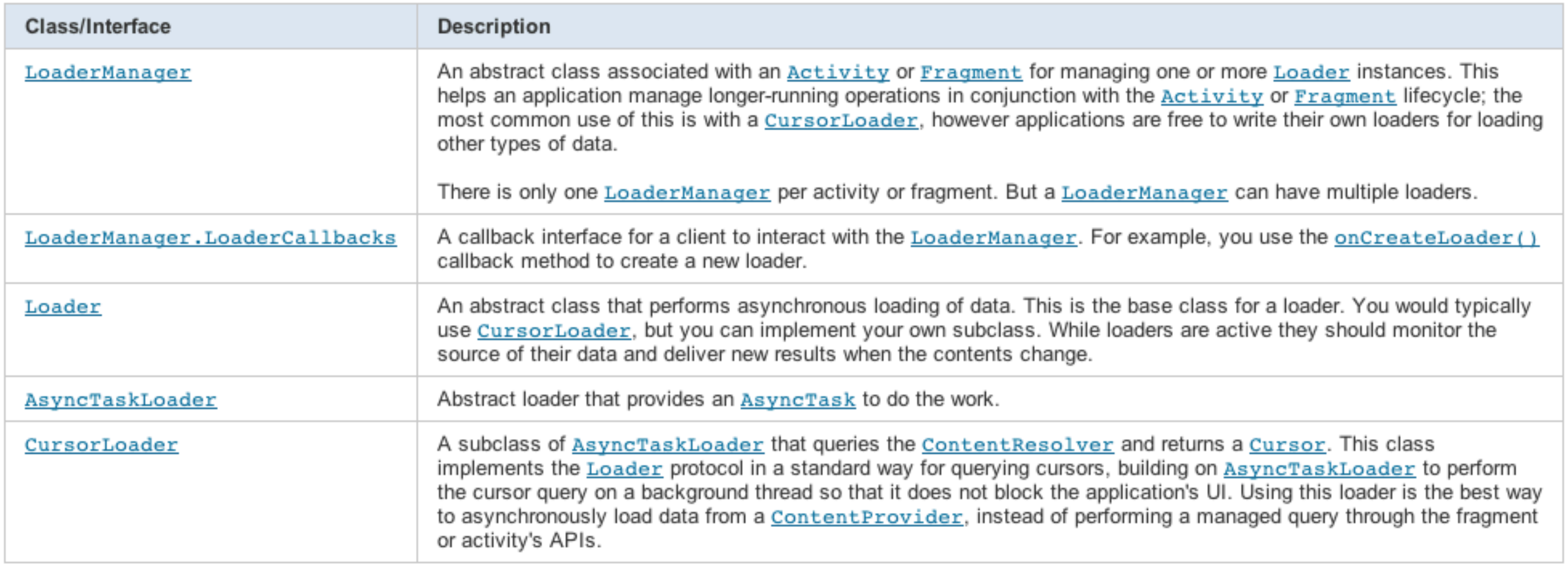
1 Funcionalidades.

- ✧ Obtener datos de forma asíncrona.
- ✧ Monitorizan el origen de datos.
- ✧ Reconectan de forma automática.
- ✧ No necesitamos volver a obtener los datos.

NOVEDADES EN HONEYCOMB

Loaders.

2 Clases.



Class/Interface	Description
LoaderManager	An abstract class associated with an Activity or Fragment for managing one or more Loader instances. This helps an application manage longer-running operations in conjunction with the Activity or Fragment lifecycle; the most common use of this is with a CursorLoader, however applications are free to write their own loaders for loading other types of data. There is only one LoaderManager per activity or fragment. But a LoaderManager can have multiple loaders.
LoaderManager.LoaderCallbacks	A callback interface for a client to interact with the LoaderManager. For example, you use the onCreateLoader() callback method to create a new loader.
Loader	An abstract class that performs asynchronous loading of data. This is the base class for a loader. You would typically use CursorLoader, but you can implement your own subclass. While loaders are active they should monitor the source of their data and deliver new results when the contents change.
AsyncTaskLoader	Abstract loader that provides an AsyncTask to do the work.
CursorLoader	A subclass of AsyncTaskLoader that queries the ContentResolver and returns a Cursor. This class implements the Loader protocol in a standard way for querying cursors, building on AsyncTaskLoader to perform the cursor query on a background thread so that it does not block the application's UI. Using this loader is the best way to asynchronously load data from a ContentProvider, instead of performing a managed query through the fragment or activity's APIs.

NOVEDADES EN HONEYCOMB

Loaders.

3 Cómo usar un Loader.

Iniciar el Loader.
getLoaderManager().initLoader(int id, bundle args, LoaderCallbacks<T> callbacks)

Reiniciar el Loader.
getLoaderManager().initLoader(int id, bundle args, LoaderCallbacks<T> callbacks)

Escuchar al Loader.

onCreateLoader(int id, bundle args)
onLoadFinished(Loader<T> loader, T data)
onLoadReset(Loader<T>)

NOVEDADES EN HONEYCOMB

Action Bar.

1 Usos del Action Bar.

- ✧ Sustituye a la barra de título.
- ✧ Muestra las acciones del menú.
- ✧ Proporciona TABs para navegar entre Fragments.
- ✧ Facilita la navegación mediante una lista de selección.
- ✧ Añade el concepto de “Action Views”.

NOVEDADES EN HONEYCOMB

Action Bar.

1 Mostrar el Action Bar.

```
<uses-sdk android:minSdkVersion="4"
          android:targetSdkVersion="11" />
```

2 Ocultar el Action Bar.

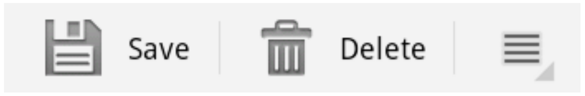
```
<activity android:theme="@android:style/Theme.Holo.NoActionBar">
```

```
ActionBar actionBar = getActionBar();
actionBar.hide();
```

NOVEDADES EN HONEYCOMB

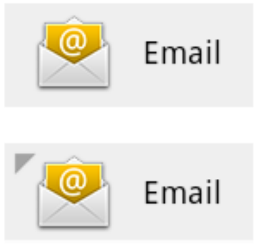
Action Bar.

3 Añadir "Action Items".



- ✧ setShowAsAction()
- ✧ SHOW_AS_ACTION_ALWAYS
- ✧ SHOW_AS_ACTION_IF_ROOM
- ✧ SHOW_AS_ACTION_NEVER
- ✧ SHOW_AS_ACTION_WITHTEXT

4 Usar el icono de la Activity.



- ✧ Android.R.id.home
- ✧ setDisplayHomeAsUpEnabled(true)

NOVEDADES EN HONEYCOMB

Action Bar.

4 Action Views.

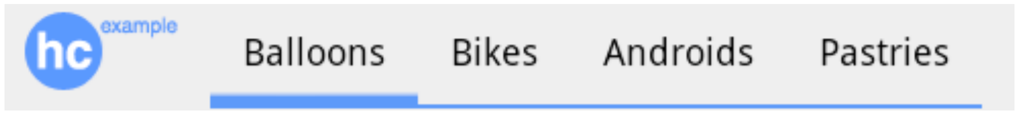
```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/menu_search"
        android:title="Search"
        android:icon="@drawable/ic_menu_search"
        android:showAsAction="ifRoom"
        android:actionLayout="@layout/searchview" />
</menu>
```

```
MenuItem item = menu.add("Buscar");
//Componente SearchView
item.setIcon(android.R.drawable.ic_menu_search);
//Si hay espacio en el menu muestras la view
item.setShowAsAction(MenuItem.SHOW_AS_ACTION_IF_ROOM);
SearchView sv = new SearchView(getActivity());
sv.setOnQueryTextListener(this);
item.setActionView(sv);
```

NOVEDADES EN HONEYCOMB

Action Bar.

5 Action Tabs.



1 ActionBar.TabListener
- ✧ onTabSelected
- ✧ onTabUnselected
- ✧ onTabReselected

2 setNavigationMode(NAVIGATION_MODE_TABS)

3 ActionBar.Tab -> newTab()
- ✧ setText(), setIcon()
- ✧ setTabListener()

4 addTab()

NOVEDADES EN HONEYCOMB

Action Bar.

6 Lista de Selección.

```
ActionBar actionBar = getActionBar();
actionBar.setNavigationMode(ActionBar.NAVIGATION_MODE_LIST);
```

```
SpinnerAdapter mSpinnerAdapter = ArrayAdapter.createFromResource(this, R.array.action_list,
          android.R.layout.simple_spinner_dropdown_item);
```

```
actionBar.setListNavigationCallbacks(mSpinnerAdapter, mNavigationCallback);
```

NOVEDADES EN HONEYCOMB

Drag&Drop.

1. Procesos implicados.

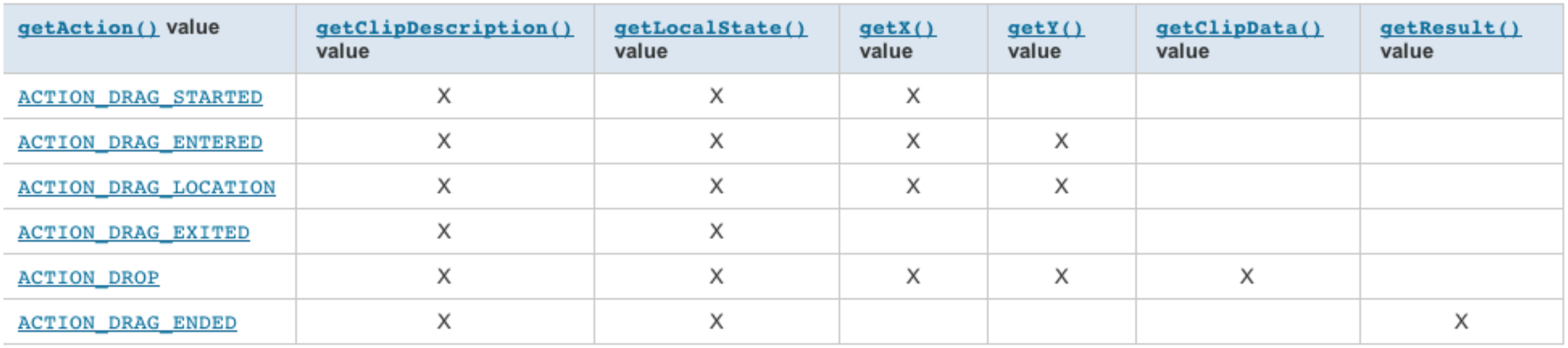
- ✧ Started
  - ➢ startDrag()
  - ➢ ACTION_DRAG_STARTED
- ✧ Continuing
  - ➢ ACTION_DRAG_ENTERED
- ✧ Dropped
  - ➢ ACTION_DROP
- ✧ Ended
  - ➢ ACTION_DRAG_ENDED

NOVEDADES EN HONEYCOMB

Drag&Drop.

2. Registrar Eventos.
   - onDragEvent()
   - setOnDragListener()

3.Eventos.



getAction() value	getClipDescription() value	getLocalState() value	getX() value	getY() value	getClipData() value	getResult() value
ACTION_DRAG_STARTED	X	X	X			
ACTION_DRAG_ENTERED	X	X	X	X		
ACTION_DRAG_LOCATION	X	X	X	X		
ACTION_DRAG_EXITED	X	X				
ACTION_DROP	X	X	X	X	X	
ACTION_DRAG_ENDED	X	X				X

NOVEDADES EN HONEYCOMB

Drag&Drop.

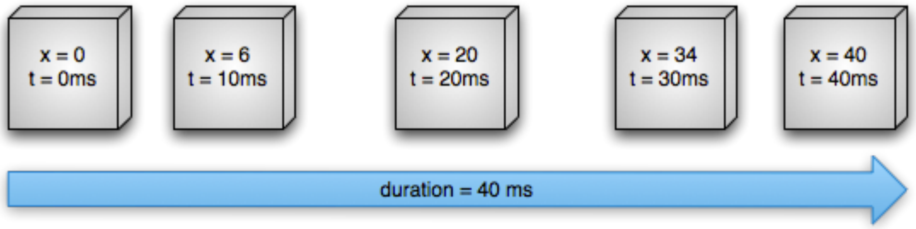
4. La sombra del arrastre.

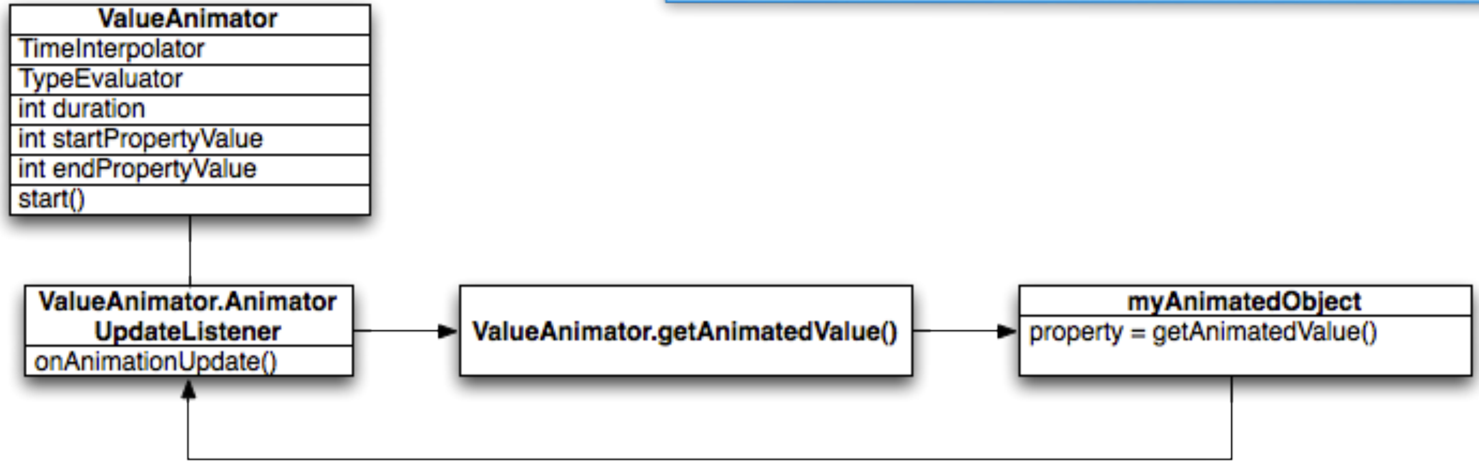
   - ✧ DragShadowBuilder()
   - ✧ DragShadowBuilder(View v)
   - ✧ onProvideShadowMetrics()
   - ✧ onDrawShadow()

NOVEDADES EN HONEYCOMB

Animación basada en propiedades.

1. Nuevo comportamiento.

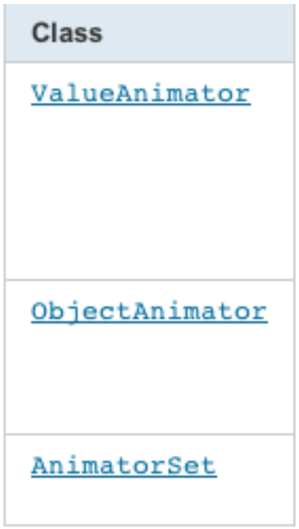




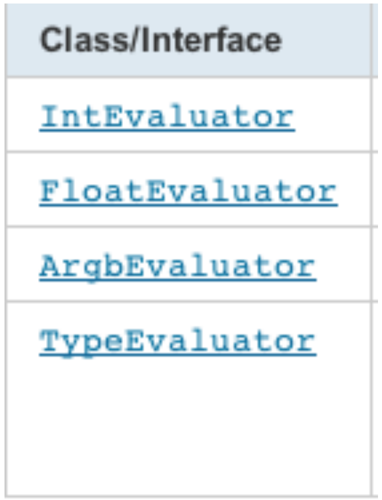
NOVEDADES EN HONEYCOMB

Animación basada en propiedades.

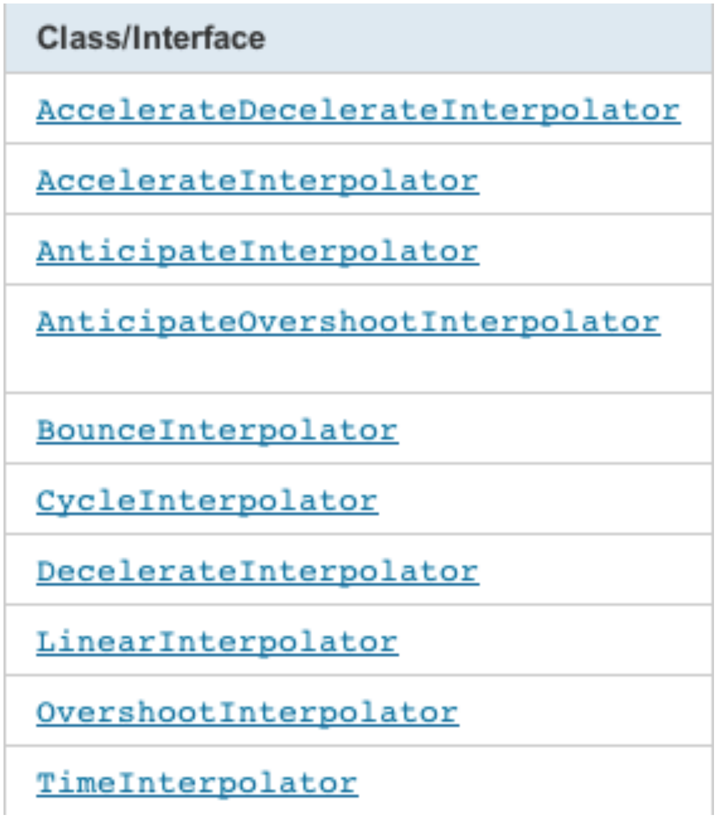
2. Clases involucradas.



Class
ValueAnimator
ObjectAnimator
AnimatorSet



Class/Interface
IntEvaluator
FloatEvaluator
ArgbEvaluator
TypeEvaluator



Class/Interface
AccelerateDecelerateInterpolator
AccelerateInterpolator
AnticipateInterpolator
AnticipateOvershootInterpolator
BounceInterpolator
CycleInterpolator
DecelerateInterpolator
LinearInterpolator
OvershootInterpolator
TimeInterpolator

NOVEDADES EN HONEYCOMB

Animación basada en propiedades.

3. ValueAnimator.

```
ValueAnimator animation = ValueAnimator.ofFloat(0f, 1f);
animation.setDuration(1000);
animation.start();
```

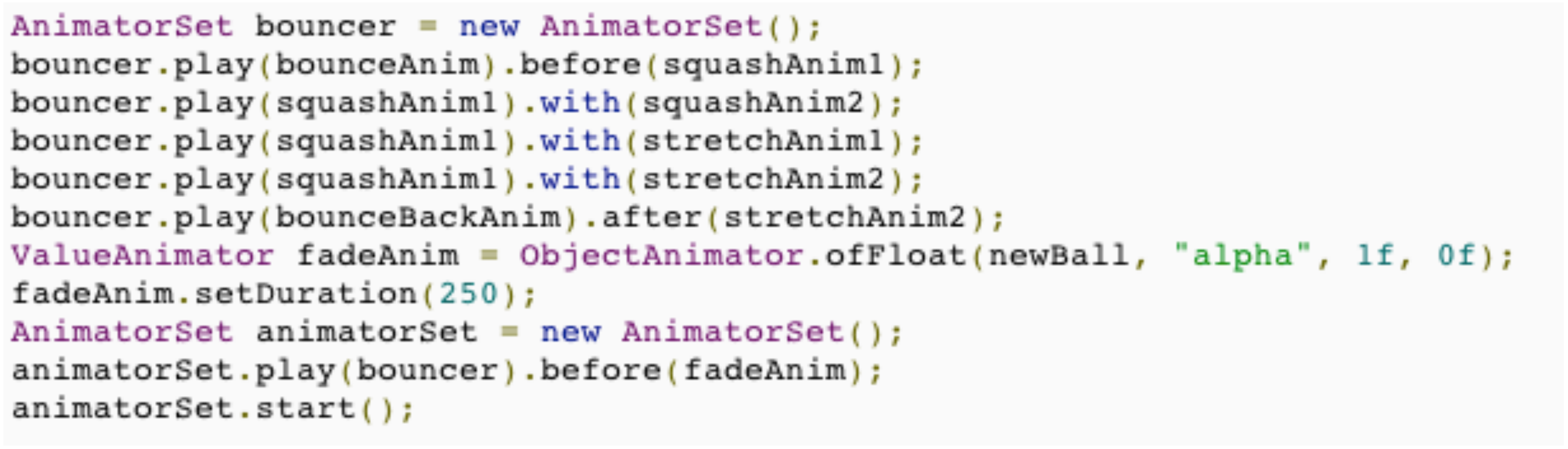
4. ObjectAnimator.

```
ObjectAnimator anim = ObjectAnimator.ofFloat(foo, "alpha", 0f, 1f);
anim.setDuration(1000);
anim.start();
```

NOVEDADES EN HONEYCOMB

Animación basada en propiedades.

5. AnimatorSet.



```
AnimatorSet bouncer = new AnimatorSet();
bouncer.play(bounceAnim).before(squashAnim1);
bouncer.play(squashAnim1).with(squashAnim2);
bouncer.play(squashAnim1).with(stretchAnim1);
bouncer.play(squashAnim1).with(stretchAnim2);
bouncer.play(bounceBackAnim).after(stretchAnim2);
ValueAnimator fadeAnim = ObjectAnimator.ofFloat(newBall, "alpha", 1f, 0f);
fadeAnim.setDuration(250);
AnimatorSet animatorSet = new AnimatorSet();
animatorSet.play(bouncer).before(fadeAnim);
animatorSet.start();
```

NOVEDADES EN HONEYCOMB

Animación basada en propiedades.

6. Eventos.

AnimatorListener

- ✧ onAnimationStart().
- ✧ onAnimationEnd().
- ✧ onAnimationRepeat().
- ✧ onAnimationCancel().
- ✧ onAnimationEnd().

NOVEDADES EN HONEYCOMB

Android Compability Package.

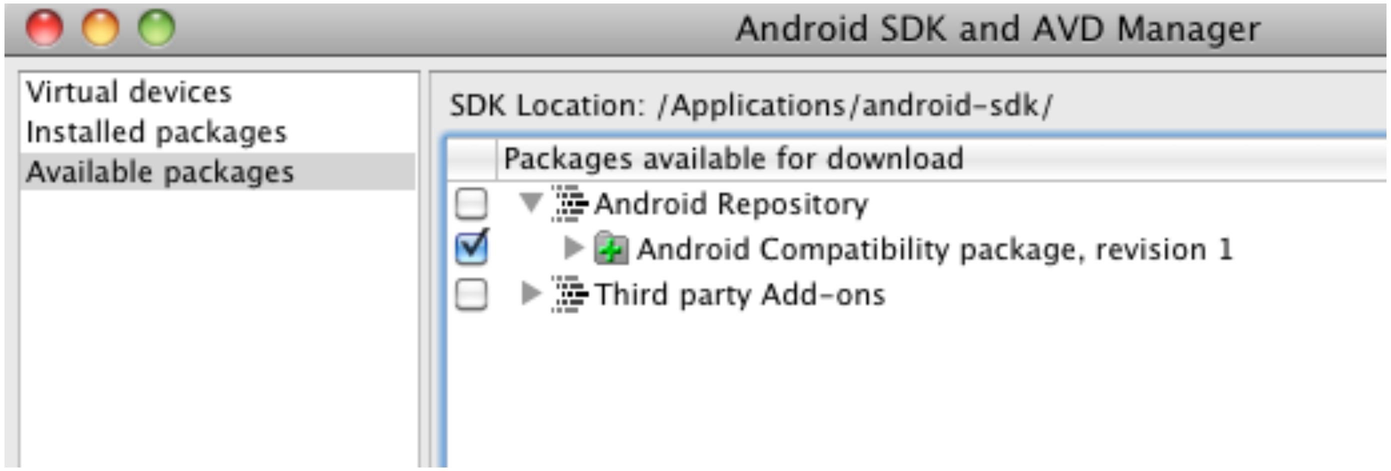
1 Contenido de android-support-v4 :

- ✧ Fragment API
- ✧ Loader API.
- ✧ CursorAdapter
- ✧ ResourceCursorAdapter
- ✧ SimpleCursorAdapter
- ✧ MenuCompat

NOVEDADES EN HONEYCOMB

Android Compability Package.

2 Configurar de android-support-v4 I:


Android SDK and AVD Manager

- Virtual devices
- Installed packages
- Available packages

SDK Location: /Applications/android-sdk/

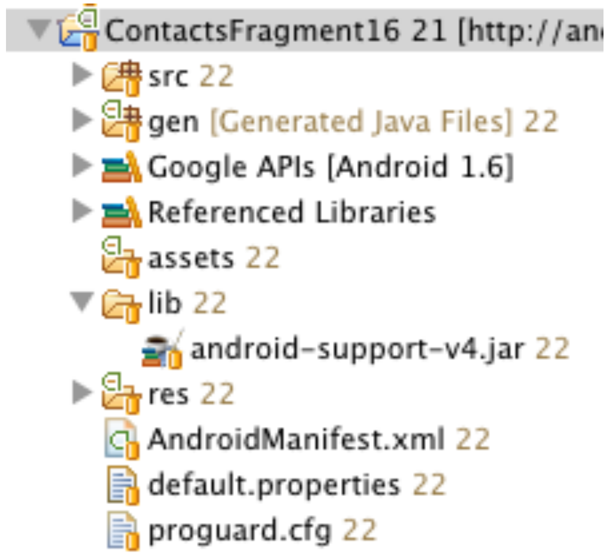
Packages available for download

- [ ] Android Repository
  - [✓] Android Compatibility package, revision 1
- [ ] Third party Add-ons

NOVEDADES EN HONEYCOMB

Android Compability Package.

3 Configurar de android-support-v4 II:

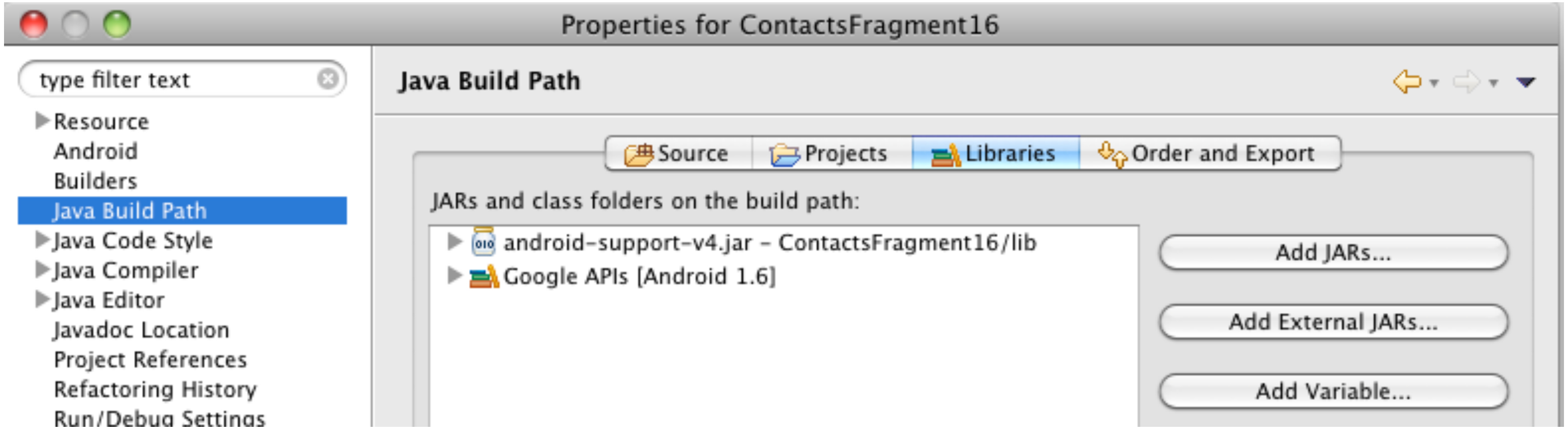


```
▼ ContactsFragment16 21 [http://an
  ▶ src 22
  ▶ gen [Generated Java Files] 22
  ▶ Google APIs [Android 1.6]
  ▶ Referenced Libraries
    assets 22
  ▼ lib 22
      android-support-v4.jar 22
  ▶ res 22
    AndroidManifest.xml 22
    default.properties 22
    proguard.cfg 22
```

NOVEDADES EN HONEYCOMB

Android Compability Package.

4 Configurar de android-support-v4 III:



Properties for ContactsFragment16

Java Build Path

JARs and class folders on the build path:

- android-support-v4.jar - ContactsFragment16/lib
- Google APIs [Android 1.6]

NOVEDADES EN HONEYCOMB

Gracias por vuestra asistencia.

Información, imágenes y recursos obtenidos de
http://android.developer.com