**Interfaz del usuario**

**En este documento**

1. [Ver Jerarquía](http://developer.android.com/guide/topics/ui/index.html#ViewHierarchy)
2. [Disposición](http://developer.android.com/guide/topics/ui/index.html#Layout)
3. [Widgets](http://developer.android.com/guide/topics/ui/index.html#Widgets)
4. [Sucesos de Entrada](http://developer.android.com/guide/topics/ui/index.html#Events)
5. [Menús](http://developer.android.com/guide/topics/ui/index.html#Menus)
6. [Temas avanzados](http://developer.android.com/guide/topics/ui/index.html#Advanced)
   1. [Adaptadores](http://developer.android.com/guide/topics/ui/index.html#Adapters)
   2. [Estilos y Temas](http://developer.android.com/guide/topics/ui/index.html#StylesAndThemes)

**Clases de clave**

1. [View](http://developer.android.com/reference/android/view/View.html)
2. [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html)
3. [Widget classes](http://developer.android.com/reference/android/widget/package-summary.html)

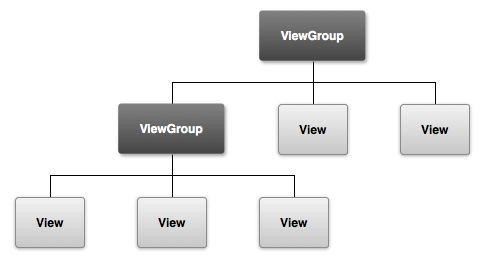
En una aplicación de Android, la interfaz de usuario se construye utilizando [View](http://developer.android.com/reference/android/view/View.html) y [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html) objetos. Hay muchos tipos de puntos de vista y los grupos de vista, cada uno de los cuales es un descendiente de la [View](http://developer.android.com/reference/android/view/View.html) de clases.

Ver objetos son las unidades básicas de la expresión de la interfaz de usuario en la plataforma Android. La clase View sirve como base para las subclases llamadas "widgets", que ofrecen la plena aplicación de objetos de interfaz de usuario, como los campos de texto y botones. La clase ViewGroup sirve de base para las subclases llamadas "disposiciones", que ofrecen diferentes tipos de arquitectura de diseño, como lineal, tabular y relativa.

Un objeto de vista es una estructura de datos cuyas propiedades almacenar los parámetros de diseño y el contenido de un área específica rectangular de la pantalla. Un objeto de vista se encarga de su propia medida, diseño, dibujo, cambio de foco, el desplazamiento y clave / gesto de las interacciones de la zona rectangular de la pantalla en la que reside. Como un objeto en la interfaz de usuario, una vista es también un punto de interacción para el usuario y el receptor de los eventos de interacción.

**Ver Jerarquía**

En la plataforma Android, se define la interfaz de usuario de una actividad utilizando una jerarquía de Vista y de los ganglios ViewGroup, como se muestra en el diagrama de abajo. Este árbol de la jerarquía puede ser tan simple o tan complejo como usted necesita que sea, y se puede construir utilizando el conjunto de widgets de Android y diseños predefinidos, o con vistas personalizadas que usted mismo crea.



Con el fin de colocar el árbol de jerarquía de la vista a la pantalla para la representación, su actividad debe llamar a la [setContentView()](http://developer.android.com/reference/android/app/Activity.html#setContentView%28int%29) el método y pasar una referencia al objeto nodo raíz. El sistema Android recibe esta referencia y la utiliza para invalidar, medir y dibujar el árbol. El nodo raíz de las peticiones de la jerarquía que sus nodos secundarios se dibujan - a su vez, cada nodo de grupo de opinión es el responsable de llamar a cada uno de sus puntos de vista propio hijo se dibujan. Los niños pueden solicitar un tamaño y ubicación dentro de la matriz, pero el objeto de los padres tiene la decisión final sobre qué tan grande que cada niño puede ser. Android analiza los elementos de su diseño en orden (desde la parte superior del árbol de jerarquía), instancias de las vistas y los agrega a su padre (s). Debido a que estos se dibujan en orden, si hay elementos que se superponen las posiciones, el último que se elaborará se encuentran en la parte superior de los demás establecidos previamente a dicho espacio.

Para una discusión más detallada sobre cómo se miden las jerarquías de vista y dibujado, lea [Cómo Android Dibuja Vistas](http://developer.android.com/guide/topics/ui/how-android-draws.html) .

**Disposición**

La manera más común para definir el diseño y expresar la jerarquía de la vista es con un archivo de formato XML. XML ofrece una estructura legible para el diseño, al igual que HTML. Cada elemento en el XML es un objeto View o ViewGroup (o sus descendientes). View objects are leaves in the tree, ViewGroup objects are branches in the tree (see the View Hierarchy figure above).

{0}·{/0} {1}       {/1} {2}El nombre de un elemento XML se respectivos a la clase Java que representa. {/2} Así que un <TextView> elemento crea una [TextView](http://developer.android.com/reference/android/widget/TextView.html) en su interfaz de usuario, y un <LinearLayout> elemento crea una [LinearLayout](http://developer.android.com/reference/android/widget/LinearLayout.html) grupo de vistas. Al cargar un recurso de diseño, el sistema Android inicializa estos objetos en tiempo de ejecución, correspondientes a los elementos en su diseño.

Por ejemplo, un simple diseño vertical, con una vista de texto y un botón tiene el siguiente aspecto:

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns: android = "http://schemas.android.com/apk/res/android"

android: layout\_width = "fill\_parent"

android: layout\_height = "fill\_parent"

android: orientation = "vertical">

<TextView android: id = "+ id @ / texto"

android: layout\_width = "wrap\_content"

android: layout\_height = "wrap\_content"

android: text = "Hola, soy un TextView" />

<Botón de android: id = "+ id @ / botón"

android: layout\_width = "wrap\_content"

android: layout\_height = "wrap\_content"

android: text = "Hola, soy un botón" />

</ LinearLayout>

Observe que el elemento LinearLayout contiene tanto el TextView y el botón. Puede anidar otra LinearLayout (u otro tipo de grupo de la vista) dentro de aquí, para alargar la jerarquía de la vista y crear un diseño más complejo.

Para más información sobre la construcción de un diseño de interfaz de usuario, lea [Layouts XML](http://developer.android.com/guide/topics/ui/declaring-layout.html) .

**Sugerencia:** También puede dibujar Ver y ViewGroups objetos en el código de Java, usando las [addView(View)](http://developer.android.com/reference/android/view/ViewGroup.html#addView%28android.view.View%29) los métodos para insertar de forma dinámica nueva vista y los objetos ViewGroup.

Hay una variedad de maneras en que usted puede diseñar los puntos de vista. Uso de más y diferentes tipos de grupos de vista, se puede estructurar puntos de vista de niños y grupos de opinión en un número infinito de formas. Algunos grupos de vistas predefinidas ofrecidas por Android (llamada diseños) son LinearLayout, RelativeLayout, TableLayout, GridLayout y otros. Cada uno ofrece un conjunto único de parámetros de diseño que se utilizan para definir las posiciones de los puntos de vista del niño y la estructura de diseño.

Para obtener información sobre algunos de los diferentes tipos de grupos de vistas utilizadas para un diseño, lea [Objetos comunes de diseño](http://developer.android.com/guide/topics/ui/layout-objects.html) .

**Widgets**

Un widget es un objeto View que sirve como interfaz para la interacción con el usuario. Android proporciona un conjunto de widgets totalmente implementadas, como botones, casillas de verificación, y campos de entrada de texto, así usted puede construir su interfaz de usuario. Algunos widgets proporcionados por Android son más complejos, como un selector de fechas, un reloj, y los controles de zoom. Pero usted no está limitado a los tipos de widgets proporcionados por la plataforma Android. Si quieres hacer algo más personalizado y crear sus propios elementos accionables, se puede, mediante la definición de su objeto propio punto de vista o por ampliar e integrar los widgets existentes.

Lea más en el [de componentes personalizados](http://developer.android.com/guide/topics/ui/custom-components.html) guía para el desarrollador.

Para obtener una lista de los widgets proporcionados por Android, consulte la [android.widget](http://developer.android.com/reference/android/widget/package-summary.html) paquete.

**Sucesos de Entrada**

Una vez que hayas añadido algunos Vistas / widgets a la interfaz de usuario, usted probablemente querrá saber acerca de la interacción del usuario con ellos, para que pueda llevar a cabo acciones. Para estar informado de los eventos de entrada de usuario, que tiene que hacer una de dos cosas:

* **Definir un detector de eventos y registrarlo con la vista.** Más a menudo que no, esto es cómo va a escuchar los eventos. La clase View contiene una colección de interfaces anidadas con nombre en *<algo>* escucha, cada uno con un método de devolución de llamada On *<something>* () . Por ejemplo, [View.OnClickListener](http://developer.android.com/reference/android/view/View.OnClickListener.html) (para el manejo de 'clic' en una vista), [View.OnTouchListener](http://developer.android.com/reference/android/view/View.OnTouchListener.html) (para controlar los eventos de la pantalla táctil en una vista), y [View.OnKeyListener](http://developer.android.com/reference/android/view/View.OnKeyListener.html) (para el manejo de las prensas de dispositivos clave dentro de una vista). Así que si usted quiere que su Vista para ser notificado cuando se "hace clic" (como cuando se selecciona un botón), poner en práctica OnClickListener y definir su onClick() el método de devolución de llamada (donde se realiza la acción sobre click), y registrarlo en la Ver con [setOnClickListener()](http://developer.android.com/reference/android/view/View.html#setOnClickListener%28android.view.View.OnClickListener%29) .
* **Sustituir un método de devolución de llamada existente para la vista.** Esto es lo que debe hacer cuando se ha implementado la clase de vista propio y desea escuchar los eventos específicos que ocurren dentro de ella. Ejemplos de eventos que puede manejar incluir cuando se toca la pantalla ( [onTouchEvent()](http://developer.android.com/reference/android/view/View.html#onTouchEvent%28android.view.MotionEvent%29) ), cuando la bola se mueve ( [onTrackballEvent()](http://developer.android.com/reference/android/view/View.html#onTrackballEvent%28android.view.MotionEvent%29) ), o cuando una llave en el dispositivo se presiona ( [onKeyDown()](http://developer.android.com/reference/android/view/View.html#onKeyDown%28int,%20android.view.KeyEvent%29) ). Esto le permite definir el comportamiento por defecto para cada evento dentro de su vista personalizada y determinar si el caso debe pasar a algún otro niño Vista. Una vez más, se trata de devoluciones de llamada a la clase View, así que su única oportunidad para definirlas es cuando usted [crear un componente personalizado](http://developer.android.com/guide/topics/ui/custom-components.html) .

Seguir leyendo sobre el manejo de la interacción del usuario con el dictamen de la [entrada Eventos](http://developer.android.com/guide/topics/ui/ui-events.html) documento.

**Menús**

Menús de la aplicación son otra parte importante de la interfaz de usuario de una aplicación. Los menús ofrece una interfaz confiable que revela las funciones de aplicación y los ajustes. El menú de la aplicación más común se revela pulsando el botón *Menú* en el dispositivo. Sin embargo, también puede agregar menús contextuales, los cuales pueden ser revelados cuando el usuario presiona y sostiene sobre un elemento.

Los menús también están estructurados mediante una jerarquía de vistas, pero no se define esta estructura a ti mismo. En su lugar, se define el [onCreateOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#onCreateOptionsMenu%28android.view.Menu%29) o [onCreateContextMenu()](http://developer.android.com/reference/android/app/Activity.html#onCreateContextMenu%28android.view.ContextMenu,%20android.view.View,%20android.view.ContextMenu.ContextMenuInfo%29) los métodos de devolución de llamada para su actividad y declarar los elementos que desea incluir en su menú. En el momento apropiado, Android creará automáticamente la jerarquía de vistas necesario para el menú y dibujar cada uno de sus elementos de menú en el mismo.

Los menús también se ocupan de sus propios actos, así que no hay necesidad de registrar detectores de eventos en los elementos de su menú. Cuando un elemento en el menú se selecciona, el [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) o [onContextItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onContextItemSelected%28android.view.MenuItem%29) método será llamado por el marco.

Y al igual que el diseño de la aplicación, usted tiene la opción de declarar los artículos de su menú en un archivo XML.

Leer [menús](http://developer.android.com/guide/topics/ui/menus.html) para obtener más información.

**Temas avanzados**

Una vez que hayas lidiado los fundamentos de la creación de una interfaz de usuario, usted puede explorar algunas características avanzadas para la creación de una interfaz de aplicación más compleja.

**Adaptadores**

A veces se quiere llenar un grupo de vistas con un poco de información que no puede ser rígida, en cambio, desea vincular a su punto de vista de una fuente externa de datos. Para ello, se utiliza un AdapterView como su punto de vista del grupo y cada niño a ver se inicializa y se rellena con los datos del adaptador.

El objeto AdapterView es una implementación de ViewGroup que determina su punto de vista del niño sobre la base de un objeto adaptador determinado. El adaptador actúa como un mensajero entre el origen de datos (tal vez una matriz de cadenas externas) y el AdapterView, que lo muestra. Hay varias implementaciones de la clase adaptador, para tareas específicas, tales como los CursorAdapter para la lectura de los datos de base de datos de un cursor o un ArrayAdapter para la lectura de una matriz arbitraria.

Para obtener más información sobre el uso de un adaptador para llenar sus puntos de vista, leer [enlaces a datos con AdapterView](http://developer.android.com/guide/topics/ui/binding.html) .

**Estilos y Temas**

Tal vez usted no está satisfecho con el aspecto de los widgets estándar. Para revisarlos, puede crear algunos de sus propios estilos y temas.

* Un estilo es un conjunto de uno o más atributos de formato que se pueden aplicar como una unidad a los elementos individuales en su diseño. Por ejemplo, puede definir un estilo que especifica un determinado tamaño del texto y el color, a continuación, aplicarla sólo a los elementos de vista específico.
* Un tema es un conjunto de uno o más atributos de formato que se pueden aplicar como una unidad a todas las actividades en una aplicación, o simplemente una sola actividad. Por ejemplo, se podría definir un tema que establece determinados colores para el marco de la ventana y el fondo del panel, y establece los tamaños de texto y los colores de los menús. Este tema puede ser aplicado a las actividades específicas o toda la aplicación.

Los estilos y los temas son los recursos. Android ofrece un estilo por defecto y los recursos temáticos que se pueden utilizar, o se puede declarar su propio estilo personalizado y recursos temáticos.

Más información sobre el uso de estilos y temas en los [estilos y temas](http://developer.android.com/guide/topics/ui/themes.html) del documento.

**Diseños de XML**

**En este documento**

1. [Escriba el código XML](http://developer.android.com/guide/topics/ui/declaring-layout.html#write)
2. [Cargar el recurso XML](http://developer.android.com/guide/topics/ui/declaring-layout.html#load)
3. [Atributos](http://developer.android.com/guide/topics/ui/declaring-layout.html#attributes)
   1. [DI](http://developer.android.com/guide/topics/ui/declaring-layout.html#id)
   2. [Los parámetros de diseño](http://developer.android.com/guide/topics/ui/declaring-layout.html#layout-params)
4. [posición.](http://developer.android.com/guide/topics/ui/declaring-layout.html#Position)
5. [Tamaño, el relleno y márgenes](http://developer.android.com/guide/topics/ui/declaring-layout.html#SizePaddingMargins)

**Clases de clave**

1. [View](http://developer.android.com/reference/android/view/View.html)
2. [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html)
3. [ViewGroup.LayoutParams](http://developer.android.com/reference/android/view/ViewGroup.LayoutParams.html)

Su distribución es la arquitectura de la interfaz de usuario en una actividad. Se define la estructura de diseño y cuenta con todos los elementos que aparecen para el usuario. Se puede declarar su diseño de dos maneras:

* **Declarar elementos de la interfaz en XML.** Android proporciona un vocabulario sencillo XML que corresponde a las clases y subclases Vista, como las de los widgets y diseños.
* **Crear instancias de elementos de diseño en tiempo de ejecución.** Su aplicación puede crear vistas y objetos ViewGroup (y manipular sus propiedades) mediante programación.

El marco de Android le da la flexibilidad para utilizar uno o ambos de estos métodos para la declaración y gestión de la interfaz de usuario de la aplicación. Por ejemplo, puede declarar los diseños de la aplicación por defecto en XML, incluyendo los elementos de pantalla que aparecen en ellas y sus propiedades. A continuación, puede agregar el código de la aplicación que se modifique el estado de los objetos en pantalla, incluyendo los declarados en XML, en tiempo de ejecución.

* El [ADT plugin para Eclipse](http://developer.android.com/sdk/eclipse-adt.html) ofrece una vista previa del diseño de su XML - con el archivo XML se abre, seleccione la ficha **Diseño.**
* Usted también debe tratar el [visor de la Jerarquía](http://developer.android.com/guide/developing/debugging/debugging-ui.html#hierarchyViewer) de herramientas, para la depuración de los diseños - que revela valores de diseño de la propiedad, se basa wireframes con indicadores de relleno / margen, y espectaculares vistas al prestados mientras se depura en el emulador o dispositivo.
* El [layoutopt](http://developer.android.com/guide/developing/debugging/debugging-ui.html#layoutopt) herramienta le permite analizar rápidamente los diseños y las jerarquías de las ineficiencias u otros problemas.

La ventaja a la declaración de la interfaz de usuario en XML es que permite separar mejor la presentación de su solicitud a partir del código que controla su comportamiento. Sus descripciones de la interfaz de usuario son externos a su código de aplicación, lo que significa que usted puede modificar o adaptar sin tener que modificar su código fuente y recompilar. Por ejemplo, puede crear diseños de XML para las orientaciones de pantalla diferentes, diferentes tamaños de pantalla del dispositivo, y en distintos idiomas. Además, se declara el diseño en XML hace que sea más fácil de visualizar la estructura de su interfaz de usuario, por lo que es más fácil de depurar problemas. Como tal, este documento se centra en la enseñanza de cómo declarar su diseño en formato XML. Si usted está interesado en crear instancias de objetos Vista en tiempo de ejecución se refieren a la [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html) y [View](http://developer.android.com/reference/android/view/View.html) referencias de clase.

En general, el vocabulario XML para elementos de la IU que declaran sigue de cerca la estructura y denominación de las clases y métodos, donde los nombres de elementos corresponden a los nombres de clase y los nombres de atributos corresponden a los métodos. De hecho, la correspondencia es a menudo tan directa que se puede adivinar lo que el atributo XML corresponde a un método de clase, o adivinar qué clase corresponde a un determinado elemento XML. Sin embargo, tenga en cuenta que no todo el vocabulario es el mismo. En algunos casos, hay ligeras diferencias de nomenclatura. Por ejemplo, el elemento EditarTexto tiene un text atributo que corresponde a EditText.setText() .

**Consejo:** Aprenda más acerca de los tipos de diseño diferentes en [objetos de diseño comunes](http://developer.android.com/guide/topics/ui/layout-objects.html) . También hay una colección de tutoriales sobre la construcción de varios diseños en la [Vistas Hola](http://developer.android.com/resources/tutorials/views/index.html) guía tutorial.

**Escriba el código XML**

Para su comodidad, la documentación de referencia de la API para las clases de interfaz de usuario se enumeran los atributos relacionados con XML disponibles que corresponden a los métodos de la clase, incluidos los atributos heredados.

Para obtener más información acerca de los elementos disponibles y atributos XML, así como el formato del archivo XML, vea [Recursos de diseño](http://developer.android.com/guide/topics/resources/available-resources.html#layoutresources) .

Uso de vocabulario XML de Android, puede diseñar rápidamente diseños de interfaz de usuario y los elementos de la pantalla que contienen, de la misma manera la creación de páginas web en HTML - con una serie de elementos anidados.

Cada archivo de formato debe contener exactamente un elemento raíz, que debe ser un Vista o un objeto ViewGroup. Una vez que haya definido el elemento raíz, puede agregar objetos de diseño adicionales o widgets como elementos secundarios para construir gradualmente una jerarquía de vista que define el diseño. Por ejemplo, aquí hay un diseño de XML que utiliza una línea vertical [LinearLayout](http://developer.android.com/reference/android/widget/LinearLayout.html) llevar a cabo una [TextView](http://developer.android.com/reference/android/widget/TextView.html) y un [Button](http://developer.android.com/reference/android/widget/Button.html) :

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns: android = "http://schemas.android.com/apk/res/android"

android: layout\_width = "fill\_parent"

android: layout\_height = "fill\_parent"

android: orientation = "vertical">

<TextView android: id = "+ id @ / texto"

android: layout\_width = "wrap\_content"

android: layout\_height = "wrap\_content"

android: text = "Hola, soy un TextView" />

<Botón de android: id = "+ id @ / botón"

android: layout\_width = "wrap\_content"

android: layout\_height = "wrap\_content"

android: text = "Hola, soy un botón" />

</ LinearLayout>

Después de haber declarado su disposición en formato XML, guarde el archivo con el .xml extensión, en su proyecto de Android res/layout/ directorio, por lo que se compilará correctamente.

Vamos a discutir cada uno de los atributos que se muestran aquí un poco más tarde.

**Cargar el recurso XML**

Al compilar la aplicación, cada archivo de formato XML se compila en una [View](http://developer.android.com/reference/android/view/View.html) de recursos. Usted debe cargar el recurso de diseño de su código de aplicación, en su [Activity.onCreate()](http://developer.android.com/reference/android/app/Activity.html#onCreate%28android.os.Bundle%29) la aplicación de devolución de llamada. Hacerlo llamando [setContentView()](http://developer.android.com/reference/android/app/Activity.html#setContentView%28int%29) , pasándole la referencia al recurso de diseño en forma de: R.layout. *layout\_file\_name* Por ejemplo, si la configuración de XML se guarda como main\_layout.xml , tendría que cargarlo para su actividad, así:

onCreate public void (paquete savedInstanceState) {

super.onCreate (savedInstanceState);

setContentView (R.layout.main\_layout);

}

El onCreate() el método de devolución de llamada en su actividad es llamado por el marco de Android cuando su actividad se puso en marcha (véase la discusión sobre los ciclos de vida, en la [Actividades](http://developer.android.com/guide/topics/fundamentals/activities.html#Lifecycle) de documento).

**Atributos**

Todos los objetos View y ViewGroup apoya su propia variedad de atributos XML. Algunos atributos son específicos de un objeto View (por ejemplo, apoya la TextView textSize atributo), pero estos atributos también se heredan por los objetos que pueden extenderse Ver esta clase. Algunos son comunes a todos los objetos de visualización, ya que se heredan de la clase View raíz (como la id atributo). Y, otros atributos son considerados "los parámetros de diseño", que son los atributos que describen ciertas orientaciones de diseño del objeto View, tal como se define por el padre objeto que objeto ViewGroup.

**DI**

Cualquier objeto View puede tener un identificador de número entero, asociado con él, para identificar de forma única la vista dentro del árbol. Cuando se compila la aplicación, esta identificación se hace referencia como un entero, pero la identificación se suele asignar en el archivo de formato XML como una cadena, en la id atributo. Se trata de un atributo XML común a todos los objetos de Vista (que se define por la [View](http://developer.android.com/reference/android/view/View.html) de clases) y lo va a usar muy a menudo. La sintaxis de una identificación, dentro de una etiqueta XML es:

android: id = "+ id @ / my\_button"

El AT-símbolo arroba (@) en el principio de la cadena indica que el analizador XML debe analizar y ampliar el resto de la cadena de identificación y lo identifican como un recurso de identificación. El plus-símbolo (+) significa que este es un nuevo nombre de recurso que debe ser creado y añadido a nuestros recursos (en el R.java archivo). Hay un número de recursos de identificación de otros que se ofrecen por el marco Android. Al hacer referencia a un identificador de recursos para Android, no es necesario el signo-símbolo, pero debe agregar el android paquete de espacio de nombres, así:

android: id = "@ android: id / vacío"

Con el android espacio de nombres del paquete en su lugar, ahora estamos haciendo referencia a un ID de la android.R recursos de la clase, en lugar de la clase de recursos locales.

A fin de crear puntos de vista y hacer referencia a ellos desde la aplicación, un patrón común es:

1. Definir un punto de vista / widget en el archivo de diseño y asignarle un identificador único:
2. <Botón de android: id = "+ id @ / my\_button"
3. android: layout\_width = "wrap\_content"
4. android: layout\_height = "wrap\_content"
5. android: text = "@ cadena / my\_button\_text" />
6. A continuación, cree una instancia del objeto de la vista y captura desde el diseño (por lo general en el [onCreate()](http://developer.android.com/reference/android/app/Activity.html#onCreate%28android.os.Bundle%29) método):
7. Button myButton = (Button) findViewById (R.id.my\_button);

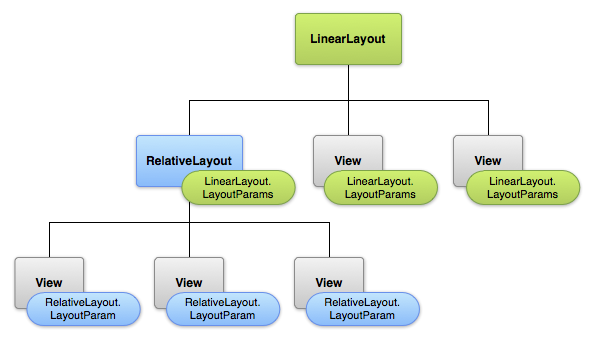
Definición de identificadores de objetos de vista es importante cuando se crea un [RelativeLayout](http://developer.android.com/reference/android/widget/RelativeLayout.html) . En una disposición relativa, puntos de vista entre hermanos puede definir su diseño en relación con otro punto de vista de hermanos, que hace referencia el identificador único.

Un ID no tiene que ser único en todo el árbol entero, pero debe ser único dentro de la parte del árbol que está buscando (que a menudo puede ser el árbol entero, así que es mejor ser completamente única cuando sea posible).

**Los parámetros de diseño**

Diseño de los atributos XML llamado layout\_ *something* definir los parámetros de diseño para la vista de que son apropiados para el ViewGroup en el que reside.

Cada clase ViewGroup implementa una clase anidada que se extiende [ViewGroup.LayoutParams](http://developer.android.com/reference/android/view/ViewGroup.LayoutParams.html) . Esta subclase contiene tipos de propiedades que definen el tamaño y la posición de cada punto de vista infantil, según sea apropiado para el grupo de opinión. Como se puede ver en la figura 1, el grupo de padres punto de vista define los parámetros de diseño para cada punto de vista infantil (incluyendo el grupo de vista del niño).



**FIGURA I.** La visualización de una jerarquía de vistas con parámetros de diseño asociados con cada vista.

Tenga en cuenta que cada subclase LayoutParams tiene su propia sintaxis para establecer los valores. Cada elemento hijo debe definir LayoutParams que sean apropiados para su casa matriz, aunque también se puede definir LayoutParams diferentes para sus propios hijos.

Todos los grupos de vista incluyen una anchura y altura ( layout\_width y layout\_height ), y cada punto de vista es necesario para definirlos. LayoutParams Muchos también incluyen márgenes opcionales y las fronteras.

Usted puede especificar el ancho y la altura con las medidas exactas, aunque es probable que no quiero hacer esto a menudo. Más a menudo, se utiliza una de estas constantes para establecer la anchura o la altura:

* *wrap\_content* le dice a su punto de vista con el tamaño en sí de las dimensiones requeridas por su contenido
* *fill\_parent* (nombre *match\_parent* en el nivel de la API 8) le dice a su fin de llegar a ser tan grande como su grupo de vista permitirá a los padres.

En general, especificando un ancho de diseño y la altura usando unidades absolutas tales como pixeles no es recomendable. En su lugar, utilizando las mediciones relativas, tales como unidades independientes de la densidad de píxeles *(DP), wrap\_content* o *fill\_parent,* es un mejor enfoque, ya que ayuda a garantizar que la aplicación se mostrará correctamente en una gran variedad de tamaños de pantalla del dispositivo. Los tipos de medición aceptados se definen en el [recursos a disposición](http://developer.android.com/guide/topics/resources/available-resources.html#dimension) de documentos.

**Diseño de Posición**

La geometría de un punto de vista es la de un rectángulo. Una vista tiene una ubicación, expresada como un par de coordenadas *izquierda* y *superior,* y dos dimensiones, expresado como una anchura y una altura. La unidad para la localización y dimensiones es el píxel.

Es posible recuperar la ubicación de un punto de vista mediante la invocación de métodos de la [getLeft()](http://developer.android.com/reference/android/view/View.html#getLeft%28%29) y [getTop()](http://developer.android.com/reference/android/view/View.html#getTop%28%29) . El primero devuelve la izquierda, o X, coordinar del rectángulo que representa la vista. Este último regresa la parte superior, o Y, coordenada del rectángulo que representa el punto de vista. Estos métodos tanto devolver la ubicación de la vista con relación a su elemento principal. Por ejemplo, cuando GetLeft () devuelve 20, que significa el punto de vista se encuentra a 20 píxeles a la derecha del borde izquierdo de su matriz directa.

Además, varios métodos de conveniencia se ofrecen para evitar cálculos innecesarios, a saber, [getRight()](http://developer.android.com/reference/android/view/View.html#getRight%28%29) y [getBottom()](http://developer.android.com/reference/android/view/View.html#getBottom%28%29) . Estos métodos devuelven las coordenadas de los bordes derecho e inferior del rectángulo que representa el punto de vista. Por ejemplo, llamando [getRight()](http://developer.android.com/reference/android/view/View.html#getRight%28%29) es similar a la siguiente cálculo: getLeft() + getWidth() .

**Tamaño, el relleno y márgenes**

El tamaño de una vista se expresa con una anchura y una altura. Una vista realmente poseen dos pares de valores de anchura y altura.

El primer par se conoce como *ancho medido* y *altura medida.* Estas dimensiones definir lo grande que uno quiere ser vista dentro de su matriz. Las dimensiones de medición se pueden obtener llamando [getMeasuredWidth()](http://developer.android.com/reference/android/view/View.html#getMeasuredWidth%28%29) y [getMeasuredHeight()](http://developer.android.com/reference/android/view/View.html#getMeasuredHeight%28%29) .

El segundo par se conoce simplemente como *la anchura* y *la altura,* o la *anchura* y la *altura* a veces, *la elaboración de dibujo.* Estas dimensiones definir el tamaño real de la vista en la pantalla, en la elaboración de tiempo y después de la presentación. Estos valores pueden, pero no tiene que, ser diferente de la anchura medida y altura. La anchura y la altura se puede obtener llamando [getWidth()](http://developer.android.com/reference/android/view/View.html#getWidth%28%29) y [getHeight()](http://developer.android.com/reference/android/view/View.html#getHeight%28%29) .

Para medir sus dimensiones, un punto de vista toma en cuenta su relleno. El relleno se expresa en píxeles para la izquierda, las partes superiores, derecho e inferior de la vista. El relleno se puede utilizar para compensar el contenido de la vista por una cantidad específica de pixeles. Por ejemplo, un margen izquierdo de 2 empujará el contenido de la vista por 2 píxeles a la derecha del borde izquierdo. El relleno se puede configurar utilizando el [setPadding(int, int, int, int)](http://developer.android.com/reference/android/view/View.html#setPadding%28int,%20int,%20int,%20int%29) método y consultar llamando [getPaddingLeft()](http://developer.android.com/reference/android/view/View.html#getPaddingLeft%28%29) , [getPaddingTop()](http://developer.android.com/reference/android/view/View.html#getPaddingTop%28%29) , [getPaddingRight()](http://developer.android.com/reference/android/view/View.html#getPaddingRight%28%29) y [getPaddingBottom()](http://developer.android.com/reference/android/view/View.html#getPaddingBottom%28%29) .

Aunque una vista puede definir un relleno, que no proporciona ningún apoyo para los márgenes. Sin embargo, los grupos de vista proporcionar tal soporte. Consulte [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html) y [ViewGroup.MarginLayoutParams](http://developer.android.com/reference/android/view/ViewGroup.MarginLayoutParams.html) para más información.

**Sucesos de Entrada**

**En este documento**

1. [Oyentes de eventos](http://developer.android.com/guide/topics/ui/ui-events.html#EventListeners)
2. [Event Handlers](http://developer.android.com/guide/topics/ui/ui-events.html#EventHandlers)
3. [Toque el modo de](http://developer.android.com/guide/topics/ui/ui-events.html#TouchMode)
4. [Manejo de enfoque](http://developer.android.com/guide/topics/ui/ui-events.html#HandlingFocus)

**Tutoriales relacionados**

1. [Formulario de cosas](http://developer.android.com/resources/tutorials/views/hello-formstuff.html)

En Android, no es más que una manera de interceptar los eventos de la interacción del usuario con la aplicación. Al considerar los acontecimientos dentro de la interfaz de usuario, el enfoque es capturar los eventos del objeto View específico que el usuario interactúa. La clase View proporciona los medios para hacerlo.

Dentro de las clases de vista diferentes que usted va a utilizar para componer su diseño, es posible que observe varios métodos de devolución de llamada públicas que parecen útiles para los eventos de la interfaz de usuario. Estos métodos son llamados por el marco de Android cuando la acción correspondiente en dicho objeto. Por ejemplo, cuando una vista (como un botón) se toca, el onTouchEvent() se llama al método en ese objeto. Sin embargo, con el fin de interceptar a esto, se debe extender la clase y reemplazar el método. Sin embargo, se extiende cada objeto Vista para manejar un evento no sería práctico. Esta es la razón por la clase View también contiene una colección de interfaces anidadas con devoluciones de llamada que se puede definir con mucha más facilidad. Estas interfaces, llamados [detectores de eventos](http://developer.android.com/guide/topics/ui/ui-events.html#EventListeners) , son su boleto a la captura de la interacción del usuario con la interfaz de usuario.

Mientras que usted con más frecuencia utilizan los detectores de eventos para escuchar la interacción del usuario, puede llegar un momento en el que se desea extender una clase de vista, con el fin de crear un componente personalizado. Tal vez usted desea extender el [Button](http://developer.android.com/reference/android/widget/Button.html) clase para hacer algo más elegante. En este caso, usted será capaz de definir los comportamientos de eventos predeterminado para la clase con los de la clase [de controladores de eventos](http://developer.android.com/guide/topics/ui/ui-events.html#EventHandlers) .

**Oyentes de eventos**

Un detector de eventos es una interfaz en la [View](http://developer.android.com/reference/android/view/View.html) de clases que contiene un método de devolución de llamada única. Estos métodos serán llamados por el marco de Android cuando la vista a la que ha sido el detector registrado es provocado por la interacción del usuario con el elemento de la interfaz de usuario.

Incluido en las interfaces de detector de eventos son los métodos de devolución de llamada siguientes:

onClick()

Desde [View.OnClickListener](http://developer.android.com/reference/android/view/View.OnClickListener.html) . Esto se llama cuando el usuario toca el tema (en el modo de contacto), o se centra en el tema con las de navegación trackball o las teclas y presiona la adecuada tecla "enter" o presiona hacia abajo en la rueda de desplazamiento.

onLongClick()

Desde [View.OnLongClickListener](http://developer.android.com/reference/android/view/View.OnLongClickListener.html) . Esto se llama cuando el usuario toca y mantiene el tema (en el modo de contacto), o se centra en el tema con las de navegación trackball o las teclas y presiona y mantiene el adecuado tecla "enter" o presiona y mantiene abajo en la rueda de desplazamiento ( para un segundo).

onFocusChange()

Desde [View.OnFocusChangeListener](http://developer.android.com/reference/android/view/View.OnFocusChangeListener.html) . Esto se llama cuando el usuario navega hacia o lejos del tema, utilizando las teclas de navegación o el trackball.

onKey()

Desde [View.OnKeyListener](http://developer.android.com/reference/android/view/View.OnKeyListener.html) . Esto se llama cuando el usuario se centra en el tema y presiona o suelta una tecla en el dispositivo.

onTouch()

Desde [View.OnTouchListener](http://developer.android.com/reference/android/view/View.OnTouchListener.html) . Esto se llama cuando el usuario realiza una acción calificada como un evento de contacto, incluyendo una prensa, la liberación, o cualquier gesto de movimiento en la pantalla (dentro de los límites del elemento).

onCreateContextMenu()

Desde [View.OnCreateContextMenuListener](http://developer.android.com/reference/android/view/View.OnCreateContextMenuListener.html) . Esto se llama cuando un menú contextual se está construyendo (como el resultado de una sostenida "pulsación larga"). Véase la discusión en los menús de contexto en el [Menús](http://developer.android.com/guide/topics/ui/menus.html#context-menu) guía para el desarrollador.

Estos métodos son los únicos habitantes de la interfaz correspondiente. Para definir uno de estos métodos y manejar sus eventos, implemente la interfaz anidado en su actividad o la definen como una clase anónima. A continuación, se pasa una instancia de su implementación a los respectivos View.set...Listener() método. (Por ejemplo, llamar a [setOnClickListener()](http://developer.android.com/reference/android/view/View.html#setOnClickListener%28android.view.View.OnClickListener%29) y pasarle la implementación de la [OnClickListener](http://developer.android.com/reference/android/view/View.OnClickListener.html) .)

El siguiente ejemplo muestra cómo registrar un detector en el clic de un botón.

/ / Crear una aplicación anónima de OnClickListener

mCorkyListener OnClickListener privada OnClickListener = new () {

public void onClick (View v) {

/ / Hacer algo cuando se pulsa el botón

}

};

protected void onCreate (savedValues ​​Bundle) {

...

/ / Captura de nuestro botón de la disposición

= Botón (Button) findViewById (R.id.corky);

/ / Registrar el detector onClick con la aplicación anterior

button.setOnClickListener (mCorkyListener);

...

}

También es posible que resulte más conveniente para poner en práctica OnClickListener como parte de su actividad. Esto evitará que la carga de la clase extra y asignación de objetos. Por ejemplo:

ExampleActivity public class Actividad OnClickListener implementos {

protected void onCreate (savedValues ​​Bundle) {

...

= Botón (Button) findViewById (R.id.corky);

button.setOnClickListener (este);

}

/ / Poner en práctica la devolución de llamada OnClickListener

public void onClick (View v) {

/ / Hacer algo cuando se pulsa el botón

}

...

}

Observe que el onClick() de devolución de llamada en el ejemplo anterior no tiene valor de retorno, pero algunos otros métodos de evento oyente debe devolver un valor lógico. La razón depende del evento. Para los pocos que lo hacen, aquí está el porqué:

* [onLongClick()](http://developer.android.com/reference/android/view/View.OnLongClickListener.html#onLongClick%28android.view.View%29) - Esta función devuelve un booleano para indicar si usted ha consumido el evento y no debe ser llevado más allá. Es decir, *devolver* true para indicar que se ha manejado el caso y debe parar aquí; devolver *false* si no lo has manipulado y / o el evento debe continuar a cualquier otro de clic en los oyentes.
* [onKey()](http://developer.android.com/reference/android/view/View.OnKeyListener.html#onKey%28android.view.View,%20int,%20android.view.KeyEvent%29) - Esta función devuelve un booleano para indicar si usted ha consumido el evento y no debe ser llevado más allá. Es decir, *devolver* true para indicar que se ha manejado el caso y debe parar aquí; devolver *false* si no lo has manipulado y / o el evento debe seguir a cualquier otra en clave de oyentes.
* [onTouch()](http://developer.android.com/reference/android/view/View.OnTouchListener.html#onTouch%28android.view.View,%20android.view.MotionEvent%29) - Esta función devuelve un booleano para indicar si su oyente consume este evento. Lo importante es que este evento puede tener múltiples acciones que se suceden. Por lo tanto, si usted vuelve *falsa* cuando el evento de acción hasta que se reciba, se indica que no han consumido el evento y tampoco están interesados ​​en las acciones posteriores de este evento. Por lo tanto, no se le pidió que cualquier otra acción dentro del evento, como un gesto con el dedo, o el caso de una eventual acción de seguimiento.

Recuerde que los acontecimientos clave, siempre estamos entregados a la vista de la actualidad en el foco. Ellos son enviados a partir de la parte superior de la jerarquía de vistas, y luego hacia abajo, hasta llegar al destino apropiado. Si la vista (o un hijo de la vista) que actualmente tiene el foco, entonces usted puede ver el recorrido hecho a través de la [dispatchKeyEvent()](http://developer.android.com/reference/android/view/View.html#dispatchKeyEvent%28android.view.KeyEvent%29) método. Como alternativa a la captura de los eventos clave a través de la Vista, también puede recibir todos los eventos dentro de su actividad con [onKeyDown()](http://developer.android.com/reference/android/app/Activity.html#onKeyDown%28int,%20android.view.KeyEvent%29) y [onKeyUp()](http://developer.android.com/reference/android/app/Activity.html#onKeyUp%28int,%20android.view.KeyEvent%29) .

**Nota:** Android se llaman controladores de eventos y luego los controladores predeterminados de las de la segunda definición de la clase. Como tal, regresando *verdadero* de estos detectores de eventos se detiene la propagación del evento para los detectores de eventos de otros y también bloquear la devolución de llamada al controlador de eventos predeterminado en la vista. Así que estar seguro de que desea terminar el evento cuando vuelva *realidad.*

**Event Handlers**

Si usted está construyendo un componente personalizado de la vista, entonces usted será capaz de definir varios métodos de devolución de llamada se utilizan como controladores de eventos predeterminados. En el documento acerca de [componentes personalizados](http://developer.android.com/guide/topics/ui/custom-components.html) , usted aprenderá ver algunos de las devoluciones de llamada comunes utilizados para la gestión de eventos, incluyendo:

* [onKeyDown(int, KeyEvent)](http://developer.android.com/reference/android/view/View.html#onKeyDown%28int,%20android.view.KeyEvent%29) - Se le llama cuando un evento de la llave se produce.
* [onKeyUp(int, KeyEvent)](http://developer.android.com/reference/android/view/View.html#onKeyUp%28int,%20android.view.KeyEvent%29) - Se le llama cuando un evento de tecla de arriba se produce.
* [onTrackballEvent(MotionEvent)](http://developer.android.com/reference/android/view/View.html#onTrackballEvent%28android.view.MotionEvent%29) - Se le llama cuando se mueve el trackball se produce.
* [onTouchEvent(MotionEvent)](http://developer.android.com/reference/android/view/View.html#onTouchEvent%28android.view.MotionEvent%29) - Se le llama cuando una pantalla táctil de eventos de movimiento se produce.
* [onFocusChanged(boolean, int, Rect)](http://developer.android.com/reference/android/view/View.html#onFocusChanged%28boolean,%20int,%20android.graphics.Rect%29) - Se le llama cuando las ganancias de vista o pierde el foco.

Hay otros métodos que usted debe tener en cuenta, que no forman parte de la clase Vista, pero puede afectar directamente la forma en que son capaces de controlar los eventos. Así que, cuando la gestión de eventos más complejos dentro de un diseño, tenga en cuenta estos otros métodos:

* [Activity.dispatchTouchEvent(MotionEvent)](http://developer.android.com/reference/android/app/Activity.html#dispatchTouchEvent%28android.view.MotionEvent%29) - Esto permite que su [Activity](http://developer.android.com/reference/android/app/Activity.html) para interceptar todos los eventos de toque antes de que sean enviados a la ventana.
* [ViewGroup.onInterceptTouchEvent(MotionEvent)](http://developer.android.com/reference/android/view/ViewGroup.html#onInterceptTouchEvent%28android.view.MotionEvent%29) - Esto permite que un [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html) para ver los eventos, ya que se envían a las observaciones del niño.
* [ViewParent.requestDisallowInterceptTouchEvent(boolean)](http://developer.android.com/reference/android/view/ViewParent.html#requestDisallowInterceptTouchEvent%28boolean%29) - Llame a este en una vista de nivel superior para indicar que no debe interceptar eventos de contacto con [onInterceptTouchEvent(MotionEvent)](http://developer.android.com/reference/android/view/ViewGroup.html#onInterceptTouchEvent%28android.view.MotionEvent%29) .

**Toque el modo de**

Cuando un usuario está navegando por una interfaz de usuario con las teclas direccionales, un trackball, es necesario dar una orientación a los elementos de recurso (como los botones) para que el usuario puede ver lo que va a aceptar la entrada. Si el dispositivo tiene capacidades táctiles, sin embargo, y el usuario comienza a interactuar con la interfaz por tocarla, entonces ya no es necesario para resaltar los elementos, o dar énfasis a una vista en particular. Así, hay un modo de interacción llamado "modo de contacto".

Para que un dispositivo táctil capaz, una vez que el usuario toca la pantalla, el dispositivo entra en modo de contacto. A partir de este punto en adelante, dictamen sólo para los que [isFocusableInTouchMode()](http://developer.android.com/reference/android/view/View.html#isFocusableInTouchMode%28%29) es cierto será recibir el foco, tales como reproductores de edición de texto. Otras vistas que son los botones táctiles, como, no se enfoque al tacto, sino que simplemente va a despedir a sus oyentes en el clic con el botón cuando se presiona.

Cada vez que un usuario pulsa una tecla de dirección o se desplaza con una rueda de desplazamiento, el dispositivo saldrá del modo de tocar y buscar el fin de tener el foco. Ahora, el usuario puede volver a interactuar con la interfaz de usuario sin tocar la pantalla.

El estado de modo de contacto se mantuvo a lo largo de todo el sistema (todas las ventanas y las actividades). Para consultar el estado actual, puede llamar a [isInTouchMode()](http://developer.android.com/reference/android/view/View.html#isInTouchMode%28%29) para ver si el dispositivo está en modo de contacto.

**Manejo de enfoque**

El marco se encargará de movimiento de foco de rutina en respuesta a la entrada del usuario. Esto incluye cambiar el foco como dictamen se eliminan o se ocultan, o como nuevos puntos de vista disponibles. Visitas al indicar su voluntad de asumir el enfoque a través de la [isFocusable()](http://developer.android.com/reference/android/view/View.html#isFocusable%28%29) método. Para cambiar de una vista si se puede tomar el foco, llame [setFocusable()](http://developer.android.com/reference/android/view/View.html#setFocusable%28boolean%29) . En el modo de contacto, puede consultar si una View permite enfocar con [isFocusableInTouchMode()](http://developer.android.com/reference/android/view/View.html#isFocusableInTouchMode%28%29) . Usted puede cambiar esto con [setFocusableInTouchMode()](http://developer.android.com/reference/android/view/View.html#setFocusableInTouchMode%28boolean%29) .

El movimiento de enfoque se basa en un algoritmo que encuentra el vecino más cercano en una dirección dada. En casos raros, el algoritmo por defecto pueden no coincidir con el comportamiento esperado de los desarrolladores. En estas situaciones, puede proporcionar reemplazos explícitos con los siguientes atributos XML en el archivo de diseño: *nextFocusDown, nextFocusLeft, nextFocusRight* y *nextFocusUp.* Añadir uno de estos atributos a la vista *del* cual el enfoque se está yendo. Defina el valor del atributo a ser el identificador de la vista *a la* que se debe dar atención. Por ejemplo:

<LinearLayout

android: orientation = "vertical"

... >

<Botón de android: id = "+ id @ / arriba"

android: nextFocusUp = "@ + id / abajo"

... />

<Botón de android: id = "+ id @ / abajo"

android: nextFocusDown = "@ + id / arriba"

... />

</ LinearLayout>

Por lo general, en este diseño vertical, la navegación desde el primer botón no ir a ninguna parte, ni la navegación desde el segundo botón. Ahora que el botón de la parte superior se ha definido la parte inferior como la *nextFocusUp* (y viceversa), el foco de navegación hará un ciclo de arriba a abajo y de abajo hacia arriba-.

Si desea declarar una vista que pueden recibir el foco en su interfaz de usuario (cuando tradicionalmente no lo es), agregue el android:focusable atributo XML a la vista, en la declaración de diseño. Establecer el valor *real.* También puede declarar una vista como al mismo tiempo recibir el foco en el modo táctil con android:focusableInTouchMode .

Para solicitar una vista particular para tener en foco, llame [requestFocus()](http://developer.android.com/reference/android/view/View.html#requestFocus%28%29) .

Para detectar los eventos de foco (ser notificado cuando una Vista recibe o pierde el foco), utilice [onFocusChange()](http://developer.android.com/reference/android/view/View.OnFocusChangeListener.html#onFocusChange%28android.view.View,%20boolean%29) , como se discutió en el [evento oyentes](http://developer.android.com/guide/topics/ui/ui-events.html#EventListeners) la sección anterior.

**Menús**

**En este documento**

1. [Definición de un menú en XML](http://developer.android.com/guide/topics/ui/menus.html#xml)
2. [Creación de un menú de opciones](http://developer.android.com/guide/topics/ui/menus.html#options-menu)
   1. [Gestión de eventos de clic](http://developer.android.com/guide/topics/ui/menus.html#RespondingOptionsMenu)
   2. [Cambio de los elementos de menú en tiempo de ejecución](http://developer.android.com/guide/topics/ui/menus.html#ChangingTheMenu)
3. [Crear menús contextuales](http://developer.android.com/guide/topics/ui/menus.html#context-menu)
   1. [Creación de un menú contextual flotante](http://developer.android.com/guide/topics/ui/menus.html#FloatingContextMenu)
   2. [Utilizando el modo de acción contextual](http://developer.android.com/guide/topics/ui/menus.html#CAB)
4. [Creación de un menú emergente](http://developer.android.com/guide/topics/ui/menus.html#PopupMenu)
   1. [Gestión de eventos de clic](http://developer.android.com/guide/topics/ui/menus.html#PopupEvents)
5. [Creación de grupos de menú](http://developer.android.com/guide/topics/ui/menus.html#groups)
   1. [Utilización de los elementos del menú checkable](http://developer.android.com/guide/topics/ui/menus.html#checkable)
6. [Agregar elementos de menú, según una intención](http://developer.android.com/guide/topics/ui/menus.html#intents)
   1. [Permitir que la actividad que se añade a otros menús](http://developer.android.com/guide/topics/ui/menus.html#AllowingToAdd)

**Clases de clave**

1. [Menu](http://developer.android.com/reference/android/view/Menu.html)
2. [MenuItem](http://developer.android.com/reference/android/view/MenuItem.html)
3. [ContextMenu](http://developer.android.com/reference/android/view/ContextMenu.html)
4. [ActionMode](http://developer.android.com/reference/android/view/ActionMode.html)

**Véase también**

1. [Barra de acción](http://developer.android.com/guide/topics/ui/actionbar.html)
2. [Menú de Recursos](http://developer.android.com/guide/topics/resources/menu-resource.html)
3. [Diga adiós al botón de menú](http://www.google.com/url?q=http%3A%2F%2Fandroid-developers.blogspot.com%2F2012%2F01%2Fsay-goodbye-to-menu-button.html&sa=D&sntz=1&usg=AFQjCNE1bd-qOTa0JviVxY2h19UKp3uPsQ)

Los menús son un componente de interfaz de usuario común en muchos tipos de aplicaciones. Para proporcionar una experiencia de usuario familiar y consistente, se debe utilizar el [Menu](http://developer.android.com/reference/android/view/Menu.html) las API para presentar las acciones del usuario y otras opciones en sus actividades.

Empezando con Android 3.0 (nivel de la API 11), con Android ya no son necesarios para proporcionar un botón de *menú* dedicado. Con este cambio, las aplicaciones de Android deben emigrar lejos de la dependencia en el panel del menú tradicional de 6 tema y en lugar de proporcionar una barra de acción para presentar las acciones comunes de los usuarios.

Aunque el diseño y la experiencia del usuario para algunos elementos de menú han cambiado, la semántica para definir un conjunto de acciones y opciones se sigue basando en el [Menu](http://developer.android.com/reference/android/view/Menu.html) las API. Esta guía muestra cómo crear los tres tipos fundamentales de los menús o las presentaciones de acción en todas las versiones de Android:

**Opciones de menú y barra de acción**

El [menú de opciones](http://developer.android.com/guide/topics/ui/menus.html#options-menu) es la colección principal de los elementos de menú para una actividad. Es el lugar donde usted debe colocar las acciones que tienen un impacto global en la aplicación, tales como "Buscar", "Redactar correo electrónico" y "Configuración".

Si va a desarrollar para Android 2.3 o inferior, los usuarios pueden abrir el panel de menú de opciones pulsando el botón *Menú.*

En Android 3.0 y superior, los elementos del menú de opciones se presentan por la [barra de acción](http://developer.android.com/guide/topics/ui/actionbar.html) como una combinación de elementos de acción en la pantalla y las opciones de desbordamiento. A partir de Android 3.0, el botón de *menú* está en desuso (algunos dispositivos no tienen uno), por lo que deben migrar hacia el uso de la barra de acción para facilitar el acceso a las acciones y otras opciones.

Vea la sección sobre la [Creación de un menú de opciones](http://developer.android.com/guide/topics/ui/menus.html#options-menu) .

**El menú contextual y el modo de acción contextual**

Un menú contextual es un [menú flotante](http://developer.android.com/guide/topics/ui/menus.html#FloatingContextMenu) que aparece cuando el usuario realiza un largo click sobre un elemento. Se indican las acciones que afectan el contenido seleccionado o en un marco contextual.

En el desarrollo de Android 3.0 y superior, en su lugar debe utilizar el [modo de acción contextual](http://developer.android.com/guide/topics/ui/menus.html#CAB) para permitir que las acciones sobre el contenido seleccionado. Este modo muestra los puntos de acción que afectan el contenido seleccionado en un bar en la parte superior de la pantalla y permite al usuario seleccionar varios elementos.

Vea la sección sobre la [Creación de menús contextuales](http://developer.android.com/guide/topics/ui/menus.html#context-menu) .

**El menú emergente**

Un menú desplegable muestra una lista de elementos de una lista vertical que está anclado a la vista que ha invocado el menú. Es bueno para proporcionar un desbordamiento de las acciones que se relacionan con el contenido específico o para proporcionar opciones para una segunda parte de un comando. Las acciones en un menú emergente **no** debería afectar directamente el contenido-que corresponde es lo que las acciones contextuales son para. Por el contrario, en el menú emergente es para las acciones extendidas que se relacionan con las regiones de contenido en su actividad.

Vea la sección sobre la [Creación de un menú emergente](http://developer.android.com/guide/topics/ui/menus.html#PopupMenu) .

**Definición de un menú en XML**

Para todos los tipos de menú, Android proporciona un formato estándar XML para definir los elementos del menú. En lugar de crear un menú en el código de su actividad, debe definir un menú y todos sus elementos en un XML [recurso de menú](http://developer.android.com/guide/topics/resources/menu-resource.html) . A continuación, puede inflar el recurso de menú (cargarlo como un [Menu](http://developer.android.com/reference/android/view/Menu.html) de objetos) en su actividad o fragmento.

El uso de un recurso de menú es una buena práctica por varias razones:

* Es más fácil de visualizar la estructura del menú en XML.
* Se separa el contenido del menú de código de comportamiento de la aplicación.
* Se le permite crear configuraciones alternativas de menú para las versiones de la plataforma, tamaños de pantalla, y otras configuraciones mediante el aprovechamiento de la [aplicación de recursos de](http://developer.android.com/guide/topics/resources/index.html) marco.

Para definir el menú, crear un archivo XML dentro de su proyecto de res/menu/ directorio y crear el menú con los siguientes elementos:

<menu>

Define un [Menu](http://developer.android.com/reference/android/view/Menu.html) , que es un contenedor para los elementos del menú. Un <menu> elemento debe ser el nodo raíz para el archivo y puede contener uno o más <item> y <group> elementos.

<item>

Crea un [MenuItem](http://developer.android.com/reference/android/view/MenuItem.html) , lo que representa un solo elemento de un menú. Este elemento puede contener un anidada <menu> elemento con el fin de crear un submenú.

<group>

Un contenedor opcional, invisible para <item> elementos. Se le permite clasificar los elementos del menú por lo que comparten características tales como el estado de activo y la visibilidad. Para obtener más información, consulte la sección sobre [creación de grupos de menús](http://developer.android.com/guide/topics/ui/menus.html#groups) .

He aquí un ejemplo de menú llamado game\_menu.xml :

<?xml version="1.0" encoding="utf-8"?>

<Menú xmlns:android="http://schemas.android.com/apk/res/android">

<Tema android: id = "+ id @ / new\_game"

android: icon = "@ dibujable / ic\_new\_game"

android: title = "@ cadena / new\_game"

android: showAsAction = "ifRoom" />

<Tema android: id = "+ id @ / ayuda"

android: icon = "@ dibujable / ic\_help"

android: title = "@ string / help" />

programa.

El <item> elemento admite varios atributos que puede utilizar para definir el aspecto de un elemento y su comportamiento. Los elementos del menú de arriba incluyen los siguientes atributos:

android:id

Un identificador de recurso que es único en el tema, lo que permite que la aplicación puede reconocer el elemento cuando el usuario lo selecciona.

android:icon

Una referencia a un dibujable a utilizar como icono del elemento.

android:title

Una referencia a una cadena que se utiliza como título del artículo.

android:showAsAction

Especifica cuándo y cómo este elemento debe aparecer como un elemento de acción en la [barra de acción](http://developer.android.com/guide/topics/ui/actionbar.html) .

Estos son los atributos más importantes que usted debe usar, pero hay muchos más disponibles. Para obtener información sobre todos los atributos compatibles, consulte el [recurso de menú](http://developer.android.com/guide/topics/resources/menu-resource.html) de documentos.

Usted puede agregar un submenú a un elemento de cualquier menú (excepto un submenú) mediante la adición de un <menu> elemento como el hijo de un <item> . Los submenús son útiles cuando la aplicación tiene una gran cantidad de funciones que pueden ser organizados en temas, como los elementos de barra de menús de una aplicación de PC (Archivo, Editar, Ver, etc.) Por ejemplo:

<?xml version="1.0" encoding="utf-8"?>

<Menú xmlns:android="http://schemas.android.com/apk/res/android">

<Tema android: id = "+ id @ / archivo"

android: title = "@ cadena / archivo">

<-! "Archivo" del submenú ->

&lt;file&gt;

<Tema android: id = "+ id @ / create\_new"

android: title = "@ cadena / create\_new" />

<Tema android: id = "+ id @ / abierto"

android: title = "@ cadena / abierto" />

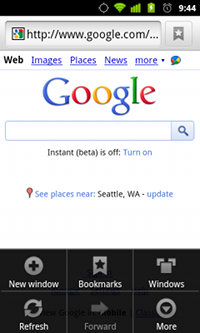
programa.

\item

programa.

Para utilizar el menú en su actividad, es necesario inflar el recurso de menú (convertir el recurso XML en un objeto programable) con [MenuInflater.inflate()](http://developer.android.com/reference/android/view/MenuInflater.html#inflate%28int,%20android.view.Menu%29) . En las secciones siguientes, veremos cómo inflar un menú para cada tipo de menú.

**Creación de un menú de opciones**



**FIGURA I.** Opciones de menú en el navegador, en Android 2.3.

El menú de opciones es donde se deben incluir acciones y otras opciones que son relevantes para el contexto de la actividad actual, como "Buscar", "Redactar correo electrónico" y "Configuración".

Cuando los elementos en el menú de opciones aparecen en la pantalla depende de la versión para el que ha desarrollado su aplicación:

* Si usted ha desarrollado su aplicación para **Android 2.3.x (API de nivel 10) o más bajo,** el contenido de su menú de opciones en la parte inferior de la pantalla cuando el usuario presiona el botón *del menú,* como se muestra en la figura 1. Cuando se abre, la parte visible primero es el menú del icono, que tiene hasta seis opciones de menú. Si su menú incluye más de seis puntos, Android coloca el sexto punto y el resto en el menú de desbordamiento, que el usuario puede abrir mediante la selección de *más.*
* Si usted ha desarrollado su aplicación para **Android 3.0 (nivel de la API 11) y más alto,** los elementos del menú de opciones están disponibles en la [barra de acción](http://developer.android.com/guide/topics/ui/actionbar.html) . Por defecto, el sistema coloca todos los elementos en el desbordamiento de la acción, que el usuario puede revelar con el icono de desbordamiento de la acción en el lado derecho de la barra de acción (o pulsando el botón de *menú del* dispositivo, si está disponible). Para habilitar el acceso rápido a acciones importantes, que pueden promover algunos elementos que aparecen en la barra de acción mediante la adición de android:showAsAction="ifRoom" a los correspondientes <item> elementos (ver figura 2).

Para obtener más información acerca de los elementos de acción y otros comportamientos de la barra de acciones, ver la [barra de acción](http://developer.android.com/guide/topics/ui/actionbar.html) de guía.

**Nota:** Incluso si usted *no* está en desarrollo para Android 3.0 o superior, usted puede crear su propio diseño barra de acción para lograr un efecto similar. Para un ejemplo de cómo puede apoyar a las versiones anteriores de Android con una barra de acción, consulte la [barra de acciones de compatibilidad](http://developer.android.com/resources/samples/ActionBarCompat/index.html) de la muestra.

http://developer.android.com/images/ui/actionbar.png

**FIGURA 2.** Barra de acciones de la [Galería de nido de abeja](http://developer.android.com/resources/samples/HoneycombGallery/index.html) aplicación, que muestra las pestañas de navegación y un elemento de acción de la cámara (más el botón de desbordamiento de acción).

Puede declarar elementos para el menú de opciones ya sea de su [Activity](http://developer.android.com/reference/android/app/Activity.html) subclase o un [Fragment](http://developer.android.com/reference/android/app/Fragment.html) subclase. Si tanto su actividad y el fragmento (s) declaran los artículos para el menú de opciones, que se combinan en la interfaz de usuario. Artículos de la actividad aparecen primero, seguido por los de cada fragmento en el orden en que se añade cada fragmento para la actividad. Si es necesario, puede volver a ordenar los elementos del menú con el android:orderInCategory atributos en cada <item> tiene que mover.

Para especificar las opciones de menú para una actividad, reemplace [onCreateOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#onCreateOptionsMenu%28android.view.Menu%29) (fragmentos de proporcionar su propio [onCreateOptionsMenu()](http://developer.android.com/reference/android/app/Fragment.html#onCreateOptionsMenu%28android.view.Menu,%20android.view.MenuInflater%29) de devolución de llamada). En este método, se puede inflar el recurso de menú ( [definido en XML](http://developer.android.com/guide/topics/ui/menus.html#xml) ) en el [Menu](http://developer.android.com/reference/android/view/Menu.html) previsto en la devolución de llamada. Por ejemplo:

@Override

onCreateOptionsMenu pública booleano (menú Menú) {

MenuInflater inflater = [getMenuInflater()](http://developer.android.com/reference/android/app/Activity.html#getMenuInflater%28%29) ;

inflater.inflate (R.menu.game\_menu, menú);

return true;

}

También puede agregar elementos del menú con [add()](http://developer.android.com/reference/android/view/Menu.html#add%28int,%20int,%20int,%20int%29) y recuperar los elementos con [findItem()](http://developer.android.com/reference/android/view/Menu.html#findItem%28int%29) a revisar sus propiedades con el [MenuItem](http://developer.android.com/reference/android/view/MenuItem.html) API.

Si usted ha desarrollado su aplicación para Android 2.3.x e inferior, el sistema llama [onCreateOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#onCreateOptionsMenu%28android.view.Menu%29) para crear el menú de opciones cuando el usuario abre el menú por primera vez. Si usted ha desarrollado para Android 3.0 y superior, el sistema llama [onCreateOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#onCreateOptionsMenu%28android.view.Menu%29) cuando se inicia la actividad, con el fin de mostrar los elementos de la barra de acción.

**Gestión de eventos de clic**

Cuando el usuario selecciona un elemento del menú de opciones (incluyendo los puntos de acción en la barra de acciones), el sistema llama a su actividad de [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) método. Este método pasa el [MenuItem](http://developer.android.com/reference/android/view/MenuItem.html) seleccionado. Puede identificar el tema llamando a [getItemId()](http://developer.android.com/reference/android/view/MenuItem.html#getItemId%28%29) , que devuelve el identificador único para el elemento de menú (definido por el android:id atributo en el recurso de menú o con un número entero dado a la [add()](http://developer.android.com/reference/android/view/Menu.html#add%28int,%20int,%20int,%20int%29) método). Puede que coincida con este identificador en contra elementos de menú conocidos para llevar a cabo la acción apropiada. Por ejemplo:

@Override

public boolean onOptionsItemSelected (elemento MenuItem) {

/ / Manipulador elemento de selección

switch (item.getItemId ()) {

caso R.id.new\_game:

newGame ();

return true;

caso R.id.help:

showHelp ();

return true;

default:

volver super.onOptionsItemSelected (punto);

}

}

Al manejar con éxito un elemento de menú, vuelva true . Si no se controla el elemento de menú, usted debe llamar a la implementación de la superclase de [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) (la implementación predeterminada devuelve false).

Si la actividad incluye fragmentos, el primer sistema de llamadas [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) para la actividad a continuación, para cada fragmento (en el orden de cada fragmento se añadió) hasta que uno vuelve true o todos los fragmentos han sido llamados.

**Sugerencia:** Android 3.0 añade la capacidad para que usted pueda definir el comportamiento de clic con el botón de un elemento de menú en formato XML, utilizando el android:onClick atributo. El valor para el atributo debe ser el nombre de un método definido por la actividad mediante el menú. El método debe ser pública y aceptar una sola [MenuItem](http://developer.android.com/reference/android/view/MenuItem.html) en parámetros cuando el sistema llama a este método, se pasa por el elemento de menú seleccionado. Para obtener más información y un ejemplo, vea el [recurso de menú](http://developer.android.com/guide/topics/resources/menu-resource.html) de documentos.

**Consejo:** Si la aplicación contiene múltiples actividades y algunos de ellos ofrecen el menú de opciones misma, considerar la creación de una actividad que implementa nada, excepto la [onCreateOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#onCreateOptionsMenu%28android.view.Menu%29) y [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) métodos. A continuación, extender esta clase para cada actividad que deben compartir el mismo menú de opciones. De esta manera, puede administrar un conjunto de códigos para el manejo de las acciones del menú y cada clase descendiente hereda los comportamientos del menú. Si desea agregar elementos de menú para una de las actividades descendientes, reemplace [onCreateOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#onCreateOptionsMenu%28android.view.Menu%29) en esa actividad. Llame super.onCreateOptionsMenu(menu) para que los elementos de menú originales se crean, a continuación, agregar nuevos elementos de menú con [menu.add()](http://developer.android.com/reference/android/view/Menu.html#add%28int,%20int,%20int,%20int%29) . También puede reemplazar el comportamiento de la super clase para los elementos de menú individuales.

**Cambio de los elementos de menú en tiempo de ejecución**

Después de que el sistema llama [onCreateOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#onCreateOptionsMenu%28android.view.Menu%29) , que mantiene una instancia de la [Menu](http://developer.android.com/reference/android/view/Menu.html) se puede rellenar y no llamará [onCreateOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#onCreateOptionsMenu%28android.view.Menu%29) de nuevo a menos que el menú está invalidado por alguna razón. Sin embargo, usted debe utilizar [onCreateOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#onCreateOptionsMenu%28android.view.Menu%29) sólo para crear el estado inicial del menú y no hacer cambios durante el ciclo de vida de la actividad.

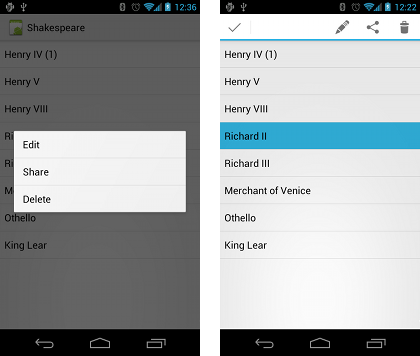
Si desea modificar el menú de opciones basadas en los eventos que ocurren durante el ciclo de vida de la actividad, puede hacerlo en el [onPrepareOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#onPrepareOptionsMenu%28android.view.Menu%29) método. Este método le pasa el [Menu](http://developer.android.com/reference/android/view/Menu.html) objetos, ya que actualmente existe lo que se puede modificar, por ejemplo, agregar, quitar o deshabilitar elementos. (Fragmentos también proporcionan una [onPrepareOptionsMenu()](http://developer.android.com/reference/android/app/Fragment.html#onPrepareOptionsMenu%28android.view.Menu%29) de devolución de llamada.)

En Android 2.3.x e inferior, el sistema llama [onPrepareOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#onPrepareOptionsMenu%28android.view.Menu%29) cada vez que el usuario abre el menú de opciones (pulsa el botón de *menú).*

En Android 3.0 y superior, el menú de opciones se considera siempre abierta cuando los elementos de menú se presentan en la barra de acción. Cuando ocurre un evento y desea realizar una actualización del menú, usted debe llamar a [invalidateOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#invalidateOptionsMenu%28%29) para solicitar que la llamada al sistema [onPrepareOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#onPrepareOptionsMenu%28android.view.Menu%29) .

**Nota:** Nunca se debe cambiar los elementos en el menú de opciones basado en la [View](http://developer.android.com/reference/android/view/View.html) en la actualidad en el foco. En el modo de contacto (cuando el usuario no está utilizando un trackball o un d-pad), puntos de vista no puede tomar el enfoque, lo que nunca debe utilizar el enfoque de base para la modificación de elementos en el menú de opciones. Si desea proporcionar elementos de menú que son sensibles al contexto a una [View](http://developer.android.com/reference/android/view/View.html) , utilice un [menú contextual](http://developer.android.com/guide/topics/ui/menus.html#context-menu) .

**Crear menús contextuales**



**Figura 3.** Capturas de pantalla de un menú contextual flotante (izquierda) y la barra de acción contextual (derecha).

Un menú contextual ofrece acciones que afectan a un elemento específico o en un marco contextual en la interfaz de usuario. Puede proporcionar un menú contextual de cualquier punto de vista, pero son los más utilizados para los elementos de un [ListView](http://developer.android.com/reference/android/widget/ListView.html) , [GridView](http://developer.android.com/reference/android/widget/GridView.html) , o de otras colecciones vistas en las que el usuario puede realizar acciones directas sobre cada elemento.

Hay dos formas de proporcionar acciones contextuales:

* En un [contexto menú flotante](http://developer.android.com/guide/topics/ui/menus.html#FloatingContextMenu) . Aparecerá un menú como una lista de elementos de menú flotante (similar a un cuadro de diálogo) cuando el usuario realiza una larga y haga clic (presione y mantenga) en una vista que declara el apoyo de un menú contextual. Los usuarios pueden realizar una acción contextual sobre un elemento a la vez.
* En el [modo de acción contextual](http://developer.android.com/guide/topics/ui/menus.html#CAB) . Este modo es una implementación del sistema de [ActionMode](http://developer.android.com/reference/android/view/ActionMode.html) que muestra una *barra de acción contextual* en la parte superior de la pantalla con los elementos de acción que afectan al elemento seleccionado (s). Cuando este modo está activo, los usuarios pueden realizar una acción en varios elementos a la vez (si su aplicación lo permite).

**Nota:** El modo de acción contextual está disponible en Android 3.0 (nivel de la API 11) y superior y es la técnica preferida para la visualización de las acciones contextuales que esté disponible. Si su aplicación compatible con las versiones inferiores a 3.0, entonces debería caer de nuevo a un menú contextual flotante en esos dispositivos.

**Creación de un menú contextual flotante**

Para proporcionar un menú contextual flotante:

1. Registro de la [View](http://developer.android.com/reference/android/view/View.html) a la que el menú de contexto debe estar asociado llamando [registerForContextMenu()](http://developer.android.com/reference/android/app/Activity.html#registerForContextMenu%28android.view.View%29) y pasarle el [View](http://developer.android.com/reference/android/view/View.html) .

Si su actividad utiliza un [ListView](http://developer.android.com/reference/android/widget/ListView.html) o [GridView](http://developer.android.com/reference/android/widget/GridView.html) y desea que cada elemento para proporcionar el mismo menú contextual, registrar todos los elementos de un menú contextual al pasar el [ListView](http://developer.android.com/reference/android/widget/ListView.html) o [GridView](http://developer.android.com/reference/android/widget/GridView.html) a [registerForContextMenu()](http://developer.android.com/reference/android/app/Activity.html#registerForContextMenu%28android.view.View%29) .

1. Implementar el [onCreateContextMenu()](http://developer.android.com/reference/android/view/View.OnCreateContextMenuListener.html#onCreateContextMenu%28android.view.ContextMenu,%20android.view.View,%20android.view.ContextMenu.ContextMenuInfo%29) método en su [Activity](http://developer.android.com/reference/android/app/Activity.html) o [Fragment](http://developer.android.com/reference/android/app/Fragment.html) .

Cuando el punto de vista social recibe un evento de larga clic con el botón, el sistema llama a su [onCreateContextMenu()](http://developer.android.com/reference/android/view/View.OnCreateContextMenuListener.html#onCreateContextMenu%28android.view.ContextMenu,%20android.view.View,%20android.view.ContextMenu.ContextMenuInfo%29) método. Aquí es donde se definen los elementos de menú, por lo general mediante el inflado de un recurso de menú. Por ejemplo:

@Override

onCreateContextMenu public void (menú ContextMenu, Vista v,

ContextMenuInfo menuInfo) {

super.onCreateContextMenu (menú, v, menuInfo);

MenuInflater inflater getMenuInflater = ();

inflater.inflate (R.menu.context\_menu, menú);

}

[MenuInflater](http://developer.android.com/reference/android/view/MenuInflater.html) le permite inflar el menú contextual de un [recurso de menú](http://developer.android.com/guide/topics/resources/menu-resource.html) . Los parámetros del método de devolución de llamada incluyen la [View](http://developer.android.com/reference/android/view/View.html) de que el usuario ha seleccionado y un [ContextMenu.ContextMenuInfo](http://developer.android.com/reference/android/view/ContextMenu.ContextMenuInfo.html) objeto que proporciona información adicional sobre el elemento seleccionado. Si su actividad tiene varios puntos de vista que cada uno ofrecen un menú contextual diferente, es posible utilizar estos parámetros para determinar qué menú contextual para inflar.

1. Implementar [onContextItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onContextItemSelected%28android.view.MenuItem%29) .

Cuando el usuario selecciona un elemento de menú, el sistema llama a este método para que pueda realizar la acción apropiada. Por ejemplo:

@Override

public boolean onContextItemSelected (elemento MenuItem) {

AdapterContextMenuInfo info = (AdapterContextMenuInfo) item.getMenuInfo ();

switch (item.getItemId ()) {

caso R.id.edit:

editNote (info.id);

return true;

caso R.id.delete:

deleteNote (info.id);

return true;

default:

volver super.onContextItemSelected (punto);

}

}

El [getItemId()](http://developer.android.com/reference/android/view/MenuItem.html#getItemId%28%29) método de consulta el ID para el elemento de menú seleccionado, que debe asignar a cada elemento de menú en XML utilizando el android:id atributo, como se muestra en la sección sobre la [Definición de un menú en XML](http://developer.android.com/guide/topics/ui/menus.html#xml) .

Al manejar con éxito un elemento de menú, vuelva true . Si no se controla el elemento de menú, debe pasar el elemento de menú a la implementación de la superclase. Si la actividad incluye fragmentos, la actividad de esta devolución de llamada recibe en primer lugar. Al llamar a la superclase, cuando no controlada, el sistema pasa el evento para el método de devolución correspondiente en cada fragmento, uno a la vez (en el orden de cada fragmento se añadió) hasta true o false se devuelve. (La aplicación por defecto para [Activity](http://developer.android.com/reference/android/app/Activity.html) y android.app.Fragment de retorno false , por lo que siempre debe llamar a la superclase, cuando no controlada.)

**Utilizando el modo de acción contextual**

El modo de acción contextual es una implementación del sistema de [ActionMode](http://developer.android.com/reference/android/view/ActionMode.html) que se centra la interacción del usuario hacia la realización de acciones contextuales. Cuando un usuario permite que este modo de seleccionar un elemento, una *barra* de *acción contextual* aparece en la parte superior de la pantalla para presentar las acciones que el usuario puede realizar en el elemento seleccionado (s). Mientras que este modo está habilitado, el usuario puede seleccionar varios elementos (si usted lo permite), artículos de anular, y continuará su camino dentro de la actividad (tanto como usted está dispuesto a permitir). El modo de acción se desactiva y la barra de acciones contextuales desaparece cuando el usuario cancela la selección de todos los artículos, presiona el botón de retroceso, o selecciona la *acción* realizada en el lado izquierdo de la barra.

**Nota:** La barra de acción contextual no está necesariamente asociada con la [barra de acción](http://developer.android.com/guide/topics/ui/actionbar.html) . Ellos operan de manera independiente, a pesar de que la barra de acción contextual visual supera a la posición de la barra de acción.

Si va a desarrollar para Android 3.0 (nivel de la API 11) o superior, por lo general deben usar el modo de acción contextual de presentar las acciones contextuales, en lugar del [menú contextual flotante](http://developer.android.com/guide/topics/ui/menus.html#FloatingContextMenu) .

Para las vistas que ofrecen las acciones contextuales, por lo general debe invocar el modo de acción contextual sobre uno de los dos eventos (o ambos):

* El usuario realiza un clic largo en la vista.
* El usuario selecciona una casilla de verificación o un componente de interfaz de usuario similar dentro de la vista.

¿Cómo la aplicación invoca el modo de acción contextual y define el comportamiento de cada acción depende de su diseño. Básicamente, existen dos diseños:

* Para las acciones contextuales sobre las opiniones individuales y arbitrarias.
* Para las acciones por lotes de contexto sobre los grupos de elementos en un [ListView](http://developer.android.com/reference/android/widget/ListView.html) o [GridView](http://developer.android.com/reference/android/widget/GridView.html) (que permite al usuario seleccionar varios elementos y realizar una acción en todos ellos).

Las secciones siguientes describen la configuración necesaria para cada escenario.

**Habilitación del modo de acción contextual de las opiniones individuales**

Si desea invocar el modo de acción contextual sólo cuando el usuario selecciona puntos de vista específicos, usted debe:

1. Implementar el [ActionMode.Callback](http://developer.android.com/reference/android/view/ActionMode.Callback.html) de interfaz. En sus métodos de devolución de llamada, puede especificar las acciones de la barra de acción contextual, haga clic en responder a los eventos en los puntos de acción, y manejar otros eventos del ciclo de vida para el modo de acción.
2. Llame [startActionMode()](http://developer.android.com/reference/android/app/Activity.html#startActionMode%28android.view.ActionMode.Callback%29) cuando se desea mostrar la barra (por ejemplo, cuando el usuario hace clic con mucho el punto de vista).

Por ejemplo:

1. Implementar el [ActionMode.Callback](http://developer.android.com/reference/android/view/ActionMode.Callback.html) de interfaz:
2. ActionMode privado.Devolución de llamada mActionModeCallback = ActionMode nuevo.Devolución de llamada () {
3. / / Se llama cuando el modo de acción se crea; startActionMode () se llama
4. @Override
5. onCreateActionMode pública booleano (ActionMode modo de menú Menú) {
6. / / Inflar un recurso de menú el suministro de elementos de menú de contexto
7. MenuInflater inflater mode.getMenuInflater = ();
8. inflater.inflate (R.menu.context\_menu, menú);
9. return true;
10. }
11. / / Se llama cada vez que se muestra el modo de acción. Siempre llamó después de onCreateActionMode, pero
12. / / Puede ser llamado varias veces si el modo se invalida.
13. @Override
14. onPrepareActionMode pública booleano (ActionMode modo de menú Menú) {
15. return false; / / Regresa falso si no se hace nada
16. }
17. / / Se llama cuando el usuario selecciona un elemento del menú contextual
18. @Override
19. public boolean onActionItemClicked (modo ActionMode, MenuItem item) {
20. switch (item.getItemId ()) {
21. caso R.id.menu\_share:
22. shareCurrentItem ();
23. mode.finish (); / / Acción elegido, tan cerca del CAB
24. return true;
25. default:
26. return false;
27. }
28. }
29. / / Se llama cuando el usuario sale del modo de acción
30. @Override
31. public void onDestroyActionMode (ActionMode modo) {
32. mActionMode = null;
33. }
34. };

Tenga en cuenta que estas devoluciones de llamada de eventos son casi exactamente las mismas que las devoluciones de llamada para el [menú de opciones](http://developer.android.com/guide/topics/ui/menus.html#options-menu) , con excepción de cada uno de ellos también pasan a la [ActionMode](http://developer.android.com/reference/android/view/ActionMode.html) objeto asociado con el evento. Usted puede utilizar [ActionMode](http://developer.android.com/reference/android/view/ActionMode.html) API para hacer varios cambios a la cabina, tales como revisar el título y el subtítulo con [setTitle()](http://developer.android.com/reference/android/view/ActionMode.html#setTitle%28int%29) y [setSubtitle()](http://developer.android.com/reference/android/view/ActionMode.html#setSubtitle%28int%29) (útil para indicar cuántos elementos están seleccionados).

Observe también que el ejemplo anterior establece el mActionMode nula variable cuando el modo de acción se destruye. En el siguiente paso, verás cómo se inicia y cómo salvar a la variable de miembro en su actividad o fragmento puede ser útil.

1. Llame [startActionMode()](http://developer.android.com/reference/android/app/Activity.html#startActionMode%28android.view.ActionMode.Callback%29) para activar el modo de acción contextual en su caso, por ejemplo, en respuesta a una larga clic en una [View](http://developer.android.com/reference/android/view/View.html) :
2. someView.setOnLongClickListener (Ver nueva.OnLongClickListener () {
3. / / Se llama cuando el usuario hace clic en largas someView
4. public boolean onLongClick (Ver opinión) {
5. if (mActionMode!nulo
6. return false;
7. }
8. / / Inicia el CAB con el ActionMode.LLAMAR definido anteriormente
9. . mActionMode = getActivity () startActionMode (mActionModeCallback);
10. view.setSelected (true);
11. return true;
12. }
13. });

Cuando se llama a [startActionMode()](http://developer.android.com/reference/android/app/Activity.html#startActionMode%28android.view.ActionMode.Callback%29) , el sistema devuelve el [ActionMode](http://developer.android.com/reference/android/view/ActionMode.html) creado. Al guardar esto en una variable miembro, usted puede realizar cambios en la barra de acción contextual en respuesta a otros eventos. En el ejemplo anterior, el [ActionMode](http://developer.android.com/reference/android/view/ActionMode.html) se utiliza para asegurar que el [ActionMode](http://developer.android.com/reference/android/view/ActionMode.html) ejemplo, no se vuelve a crear, si ya está activa, comprobando si el miembro es nulo antes de iniciar el modo de acción.

**Habilitación de lotes de contexto las acciones de un control ListView o GridView**

Si usted tiene una colección de elementos en un [ListView](http://developer.android.com/reference/android/widget/ListView.html) o [GridView](http://developer.android.com/reference/android/widget/GridView.html) (o cualquier otra extensión de [AbsListView](http://developer.android.com/reference/android/widget/AbsListView.html) ) y desea permitir a los usuarios realizar acciones por lotes, usted debe:

* Implementar el [AbsListView.MultiChoiceModeListener](http://developer.android.com/reference/android/widget/AbsListView.MultiChoiceModeListener.html) interfaz y la puso en el grupo de vista con [setMultiChoiceModeListener()](http://developer.android.com/reference/android/widget/AbsListView.html#setMultiChoiceModeListener%28android.widget.AbsListView.MultiChoiceModeListener%29) . En los métodos de devolución de llamada del oyente, puede especificar las acciones de la barra de acción contextual, haga clic en responder a los eventos de los elementos de acción, y manejar las devoluciones de llamada de otros heredados de la [ActionMode.Callback](http://developer.android.com/reference/android/view/ActionMode.Callback.html) de interfaz.
* Llame [setChoiceMode()](http://developer.android.com/reference/android/widget/AbsListView.html#setChoiceMode%28int%29) con la [CHOICE\_MODE\_MULTIPLE\_MODAL](http://developer.android.com/reference/android/widget/AbsListView.html#CHOICE_MODE_MULTIPLE_MODAL) argumento.

Por ejemplo:

ListView ListView = getListView ();

listView.setChoiceMode (ListView.CHOICE\_MODE\_MULTIPLE\_MODAL);

listView.setMultiChoiceModeListener (MultiChoiceModeListener nuevo () {

@Override

public void onItemCheckedStateChanged (modo ActionMode, int posición,

Identificación de largo, boolean seleccionada) {

/ / Aquí se puede hacer algo cuando se seleccionan los elementos / desactivada,

/ / Como actualizar el título en el CAB

}

@Override

public boolean onActionItemClicked (modo ActionMode, MenuItem item) {

/ / Responder a los clics de las acciones en el CAB

switch (item.getItemId ()) {

caso R.id.menu\_delete:

deleteSelectedItems ();

mode.finish (); / / Acción elegido, tan cerca del CAB

return true;

default:

return false;

}

}

@Override

onCreateActionMode pública booleano (ActionMode modo de menú Menú) {

/ / Inflar el menú para el CAB

MenuInflater inflater mode.getMenuInflater = ();

inflater.inflate (R.menu.context, menú);

return true;

}

@Override

public void onDestroyActionMode (ActionMode modo) {

/ / Aquí puedes hacer las actualizaciones necesarias a la actividad cuando

/ / El CAB se ha eliminado. De forma predeterminada, los elementos seleccionados son deseleccionado / sin marcar.

}

@Override

onPrepareActionMode pública booleano (ActionMode modo de menú Menú) {

/ / Aquí se pueden realizar cambios a la JCA, debido a

/ / Un [invalidate()](http://developer.android.com/reference/android/view/ActionMode.html#invalidate%28%29) solicitud de

return false;

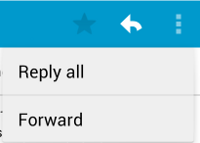
}

});

Eso es todo. Ahora cuando el usuario selecciona un elemento con una pulsación larga, el sistema llama al [onCreateActionMode()](http://developer.android.com/reference/android/view/ActionMode.Callback.html#onCreateActionMode%28android.view.ActionMode,%20android.view.Menu%29) método y muestra la barra de acción contextual con las acciones indicadas. Mientras que la barra de acción contextual es visible, los usuarios pueden seleccionar elementos adicionales.

En algunos casos en los que las acciones contextuales proporcionan los elementos comunes de acción, es posible que desee agregar una casilla de verificación o un elemento de la interfaz de usuario similar que permite a los usuarios seleccionar los elementos, porque no podría descubrir el comportamiento a largo clic. Cuando un usuario selecciona la casilla de verificación, puede invocar el modo de acción contextual estableciendo el elemento de la lista respectiva para el estado de activación con [setItemChecked()](http://developer.android.com/reference/android/widget/AbsListView.html#setItemChecked%28int,%20boolean%29) .

**Creación de un menú emergente**



**Figura 4.** Un menú desplegable en la aplicación de Gmail, anclado en el botón de desbordamiento en la parte superior derecha.

Un [PopupMenu](http://developer.android.com/reference/android/widget/PopupMenu.html) es un menú de modos de transporte anclada a una [View](http://developer.android.com/reference/android/view/View.html) . Al parecer, debajo del ancla ver si hay espacio, o por encima del punto de vista de otra manera. Es útil para:

* Ofrecer un menú de desbordamiento de estilo para las acciones *que* se *relacionan con* contenidos específicos (como los encabezados de correo electrónico de Gmail, que se muestran en la figura 4).

**Nota:** Esto no es lo mismo que un menú de contexto, que es por lo general para las acciones que *afectan* el contenido seleccionado. Para las acciones que afectan el contenido seleccionado, utilice el [modo de acción contextual](http://developer.android.com/guide/topics/ui/menus.html#CAB) o el [menú contextual flotante](http://developer.android.com/guide/topics/ui/menus.html#FloatingContextMenu) .

* Proporcionar una segunda parte de una frase de comandos (como un botón "Añadir" que produce un menú con diferentes opciones de "Agregar").
* Proporcionar un desplegable similar a [Spinner](http://developer.android.com/reference/android/widget/Spinner.html) que no retenga una selección persistente.

**Nota:** [PopupMenu](http://developer.android.com/reference/android/widget/PopupMenu.html) está disponible con el nivel de API de 11 y superior.

Si [define su menú en XML](http://developer.android.com/guide/topics/ui/menus.html#xml) , así es como se puede mostrar el menú emergente:

1. Instantate un [PopupMenu](http://developer.android.com/reference/android/widget/PopupMenu.html) con su constructor, que tiene la aplicación actual [Context](http://developer.android.com/reference/android/content/Context.html) y la [View](http://developer.android.com/reference/android/view/View.html) a la que debe ser el menú anclado.
2. Utilice [MenuInflater](http://developer.android.com/reference/android/view/MenuInflater.html) para inflar su recurso de menú en el [Menu](http://developer.android.com/reference/android/view/Menu.html) objeto devuelto por [PopupMenu.getMenu()](http://developer.android.com/reference/android/widget/PopupMenu.html#getMenu%28%29) . El nivel de la API y por encima de 14, puede utilizar [PopupMenu.inflate()](http://developer.android.com/reference/android/widget/PopupMenu.html#inflate%28int%29) en su lugar.
3. Llame [PopupMenu.show()](http://developer.android.com/reference/android/widget/PopupMenu.html#show%28%29) .

Por ejemplo, aquí hay un botón con el [android:onClick](http://developer.android.com/reference/android/R.attr.html#onClick) atributo que muestra un menú emergente:

<ImageButton

android: layout\_width = "wrap\_content"

android: layout\_height = "wrap\_content"

android: src = "@ dibujable / ic\_overflow\_holo\_dark"

android: contentDescription = "@ cadena / descr\_overflow\_button"

android: onClick = "showPopup" />

La actividad puede mostrar el menú emergente de esta manera:

public void showPopup (Ver v) {

PopupMenu popup = new PopupMenu (esto, v);

MenuInflater inflater popup.getMenuInflater = ();

inflater.inflate (R.menu.actions, los popup.getMenu ());

popup.show ();

}

En el nivel 14 y superior de la API, puede combinar las dos líneas que se inflan el menú con [PopupMenu.inflate()](http://developer.android.com/reference/android/widget/PopupMenu.html#inflate%28int%29) .

El menú es despedido cuando el usuario selecciona un elemento o toque fuera del área de menú. Usted puede escuchar el despido de eventos mediante [PopupMenu.OnDismissListener](http://developer.android.com/reference/android/widget/PopupMenu.OnDismissListener.html) .

**Gestión de eventos de clic**

Para realizar una acción cuando el usuario selecciona un elemento de menú, debe implementar la [PopupMenu.OnMenuItemClickListener](http://developer.android.com/reference/android/widget/PopupMenu.OnMenuItemClickListener.html) interfaz y registrarlo con su [PopupMenu](http://developer.android.com/reference/android/widget/PopupMenu.html) llamando [setOnMenuItemclickListener()](http://developer.android.com/reference/android/widget/PopupMenu.html#setOnMenuItemClickListener%28android.widget.PopupMenu.OnMenuItemClickListener%29) . Cuando el usuario selecciona un elemento, el sistema llama al [onMenuItemClick()](http://developer.android.com/reference/android/widget/PopupMenu.OnMenuItemClickListener.html#onMenuItemClick%28android.view.MenuItem%29) de devolución de llamada en la interfaz.

Por ejemplo:

showMenu public void (Ver v) {

PopupMenu popup = new PopupMenu (esto, v);

/ / Esta actividad implementa OnMenuItemClickListener

popup.setOnMenuItemClickListener (este);

popup.inflate (R.menu.actions);

popup.show ();

}

@Override

public boolean onMenuItemClick (elemento MenuItem) {

switch (item.getItemId ()) {

caso R.id.archive:

Archivo (punto);

return true;

caso R.id.delete:

Eliminar elemento

return true;

default:

return false;

}

}

**Creación de grupos de menú**

Un grupo de menús es una colección de elementos de menú que comparten ciertos rasgos. Con un grupo, usted puede:

* Mostrar u ocultar todos los elementos con [setGroupVisible()](http://developer.android.com/reference/android/view/Menu.html#setGroupVisible%28int,%20boolean%29)
* Activar o desactivar todos los elementos con [setGroupEnabled()](http://developer.android.com/reference/android/view/Menu.html#setGroupEnabled%28int,%20boolean%29)
* Especifique si todos los elementos son verificables con [setGroupCheckable()](http://developer.android.com/reference/android/view/Menu.html#setGroupCheckable%28int,%20boolean,%20boolean%29)

Usted puede crear un grupo de anidación <item> elementos dentro de un <group> elemento en su recurso de menú o mediante la especificación de un ID de grupo con la del [add()](http://developer.android.com/reference/android/view/Menu.html#add%28int,%20int,%20int,%20int%29) método.

He aquí un ejemplo de recurso de menú que incluye un grupo:

<?xml version="1.0" encoding="utf-8"?>

<Menú xmlns:android="http://schemas.android.com/apk/res/android">

<Tema android: id = "+ id @ / menu\_save"

android: icon = "@ dibujable / menu\_save"

android: title = "@ cadena / menu\_save" />

<-! Grupo de menú ->

grupo> android:id="@+id/group\_delete">

<Tema android: id = "+ id @ / menu\_archive"

android: title = "@ cadena / menu\_archive" />

<Tema android: id = "+ id @ / menu\_delete"

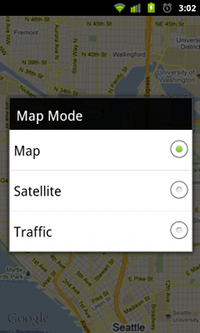
android: title = "@ cadena / menu\_delete" />

Grupos

programa.

Los elementos que están en el grupo aparecen en el mismo nivel que el primer tema-los tres elementos en el menú son hermanos. Sin embargo, puede modificar las características de los dos puntos en el grupo haciendo referencia a la ID de grupo y el uso de los métodos mencionados anteriormente. El sistema nunca se agrupan también por separado los elementos. Por ejemplo, si se declara android:showAsAction="ifRoom" para cada artículo, o bien se parecen tanto en la barra de acciones o de ambos aparecen en el desbordamiento de la acción.

**Utilización de los elementos del menú checkable**



**Figura 5.** Captura de pantalla de un submenú con los elementos comprobables.

Un menú puede ser útil como una interfaz para activar las opciones de encendido y apagado, con una casilla de verificación de opciones por separado o botones de radio para grupos de opciones mutuamente excluyentes. La Figura 5 muestra un submenú con los artículos que son verificables con los botones de radio.

**Nota:** Los elementos del menú en el menú de iconos (en el menú de opciones) no puede mostrar una casilla de verificación o botón de radio. Si opta por hacer que los elementos en el menú de iconos comprobable, de forma manual debe indicar el estado de activación mediante el canje en el icono y / o de texto cada vez que los cambios de estado.

Se puede definir el comportamiento comprobable de diferentes puntos del menú utilizando el android:checkable atributo en el <item> elemento, o para todo un grupo con el android:checkableBehavior atributo en el <group> elemento. Por ejemplo, todos los elementos de este grupo de menú son verificables, con un botón de radio:

<?xml version="1.0" encoding="utf-8"?>

<Menú xmlns:android="http://schemas.android.com/apk/res/android">

grupo> android:checkableBehavior="single">

<Tema android: id = "+ id @ / rojo"

android: title = "@ cadena / rojo" />

<Tema android: id = "+ id @ / azul"

android: title = "@ cadena / azul" />

Grupos

programa.

El android:checkableBehavior atributo acepta ya sea:

single

Sólo un elemento del grupo se puede comprobar (botones de radio)

all

Todos los artículos se puede comprobar (casillas)

none

No hay elementos son verificables

Usted puede solicitar un incumplimiento comprobado el estado de un elemento con el android:checked atributo en el <item> elemento y el cambio en el código con la [setChecked()](http://developer.android.com/reference/android/view/MenuItem.html#setChecked%28boolean%29) método.

Cuando un elemento comprobable es seleccionado, el sistema llama a su respectivo tema seleccionado por el método de devolución de llamada (por ejemplo, [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) ). Es ahí donde se debe establecer el estado de la casilla de verificación, ya que un botón de casilla de verificación o de radio no cambia su estado de forma automática. Puede consultar el estado actual del tema (como lo era antes de que el usuario lo selecciona) con [isChecked()](http://developer.android.com/reference/android/view/MenuItem.html#isChecked%28%29) y luego establecer el estado de activación con [setChecked()](http://developer.android.com/reference/android/view/MenuItem.html#setChecked%28boolean%29) . Por ejemplo:

@Override

public boolean onOptionsItemSelected (elemento MenuItem) {

switch (item.getItemId ()) {

caso R.id.vibrate:

caso R.id.dont\_vibrate:

if (item.isChecked ()) item.setChecked (false);

otra item.setChecked (true);

return true;

default:

volver super.onOptionsItemSelected (punto);

}

}

Si no se establece el estado de activación de esta manera, entonces el estado de visibilidad del tema (la casilla de verificación o botón de radio) no va a cambiar cuando el usuario lo selecciona. Cuando se establezca el estado, la actividad mantiene el estado de activación del elemento de modo que cuando el usuario abre el menú más tarde, el estado de activación que se establece es visible.

**Nota: Los** elementos del menú checkable están destinados a ser utilizados sólo en función de cada sesión y no se guardan después de la aplicación se destruye. Si usted tiene configuración de la aplicación que le gustaría ahorrar para el usuario, debe almacenar los datos con [las preferencias compartidas](http://developer.android.com/guide/topics/data/data-storage.html#pref) .

**Agregar elementos de menú, según una intención**

A veces se quiere un elemento de menú para poner en marcha una actividad con una [Intent](http://developer.android.com/reference/android/content/Intent.html) (si se trata de una actividad en su aplicación o en otra aplicación). Cuando se conoce la intención que desea utilizar y tener un elemento de menú específico que debe iniciar la intención, puede ejecutar la intención con la [startActivity()](http://developer.android.com/reference/android/app/Activity.html#startActivity%28android.content.Intent%29) en el método de devolución apropiada en el punto seleccionado (como el [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) de devolución de llamada).

Sin embargo, si usted no está seguro de que el dispositivo del usuario contiene una aplicación que se encarga de la intención, a continuación, añadir un elemento de menú que invoca puede dar lugar a un elemento de menú no funciona, porque la intención no puede resolver en la actividad. Para solucionar esto, Android le permite agregar dinámicamente elementos de menú para el menú cuando se encuentra Android actividades en el dispositivo que se encargan de su intención.

Para agregar elementos de menú sobre la base de las actividades disponibles que aceptan la intención:

1. Definir la intención con la categoría [CATEGORY\_ALTERNATIVE](http://developer.android.com/reference/android/content/Intent.html#CATEGORY_ALTERNATIVE) y / o [CATEGORY\_SELECTED\_ALTERNATIVE](http://developer.android.com/reference/android/content/Intent.html#CATEGORY_SELECTED_ALTERNATIVE) , además de otros requisitos.
2. Llame [Menu.addIntentOptions()](http://developer.android.com/reference/android/view/Menu.html#addIntentOptions%28int,%20int,%20int,%20android.content.ComponentName,%20android.content.Intent%5B%5D,%20android.content.Intent,%20int,%20android.view.MenuItem%5B%5D%29) . Android busca entonces todas las aplicaciones que pueden llevar a cabo la intención y los agrega a su menú.

Si no hay aplicaciones instaladas que tener la misma finalidad, entonces no hay elementos de menú se agrega.

**Nota:** [CATEGORY\_SELECTED\_ALTERNATIVE](http://developer.android.com/reference/android/content/Intent.html#CATEGORY_SELECTED_ALTERNATIVE) se utiliza para controlar el elemento seleccionado en ese momento en la pantalla. Por lo tanto, sólo debe utilizarse al crear un menú en [onCreateContextMenu()](http://developer.android.com/reference/android/app/Activity.html#onCreateContextMenu%28android.view.ContextMenu,%20android.view.View,%20android.view.ContextMenu.ContextMenuInfo%29) .

Por ejemplo:

@Override

onCreateOptionsMenu pública booleano (menú Menú) {

super.onCreateOptionsMenu (menú);

/ / Crear una intención que describe los requisitos que cumplir, para ser incluido

/ / En nuestro menú. La aplicación debe incluir una oferta de valor de la categoría de intención.CATEGORY\_ALTERNATIVE.

Intención de Intención = nuevo intento (null, dataUri);

intent.addCategory (Intent.CATEGORY\_ALTERNATIVE);

/ / Busca y completar el menú con las aplicaciones que ofrece aceptables.

(menu.addIntentOptions

R.id.intent\_group, / / ​​grupo de artículos de menú al que se añadirán nuevos

0, / / ​​ID único elemento (ninguno)

0, / / ​​Orden de los elementos (ninguno)

this.getComponentName (), / / ​​El nombre de la actividad actual

NULL, / / ​​los elementos específicos para el primer lugar (ninguno)

la intención, / / ​​Intención creado anteriormente que describe los requisitos

0, banderas / / adicionales en los elementos de control (no hay)

null); / / Array de objetos MenuItem que se correlacionan con temas específicos (ninguno)

return true;

}

Para cada actividad determinó que ofrece un filtro de la intención que coincida con el propósito definido, un elemento de menú se agrega, utilizando el valor en el intento de filtro de android:label , como el título del artículo del menú y el icono de la aplicación como el icono del menú. El [addIntentOptions()](http://developer.android.com/reference/android/view/Menu.html#addIntentOptions%28int,%20int,%20int,%20android.content.ComponentName,%20android.content.Intent%5B%5D,%20android.content.Intent,%20int,%20android.view.MenuItem%5B%5D%29) método devuelve el número de elementos de menú añadido.

**Nota:** Cuando se llama a [addIntentOptions()](http://developer.android.com/reference/android/view/Menu.html#addIntentOptions%28int,%20int,%20int,%20android.content.ComponentName,%20android.content.Intent%5B%5D,%20android.content.Intent,%20int,%20android.view.MenuItem%5B%5D%29) , anula cualquier y todos los elementos del menú por el grupo de menú especificado en el primer argumento.

**Permitir que la actividad que se añade a otros menús**

También puede ofrecer los servicios de su actividad a otras aplicaciones, por lo que su aplicación puede ser incluido en el menú de los demás (invertir los papeles descritos más arriba).

Para ser incluido en los menús de la aplicación de otro modo, es necesario definir un filtro de la intención, como de costumbre, pero asegúrese de incluir los [CATEGORY\_ALTERNATIVE](http://developer.android.com/reference/android/content/Intent.html#CATEGORY_ALTERNATIVE) y / o [CATEGORY\_SELECTED\_ALTERNATIVE](http://developer.android.com/reference/android/content/Intent.html#CATEGORY_SELECTED_ALTERNATIVE) valores de la categoría intención de filtro. Por ejemplo:

<intent-filter label="@string/resize\_image">

...

<Categoría de android: name = "android.intent.category.ALTERNATIVA "/>

<Categoría de android: name = "android.intent.category.SELECTED\_ALTERNATIVE "/>

...

</ Intención de filtro>

Lea más acerca de la escritura de filtros en la intención de las [intenciones y la intención de filtros](http://developer.android.com/guide/topics/intents/intents-filters.html) de documentos.

Para una aplicación de ejemplo que utiliza esta técnica, consulte el [Bloc de Notas](http://developer.android.com/resources/samples/NotePad/src/com/example/android/notepad/NoteEditor.html) código de ejemplo.

**Barra de acción**

**Vista rápida**

* Una barra de título que incluye el icono de la aplicación y el título de la actividad
* Proporciona acceso a los elementos de menú y los modos de navegación, tales como fichas
* Requiere nivel 11 o mayor de la API

**En este documento**

1. [Adición de la Barra de acciones](http://developer.android.com/guide/topics/ui/actionbar.html#Adding)
   1. [Extracción de la barra de acción](http://developer.android.com/guide/topics/ui/actionbar.html#Removing)
2. [Agregar elementos de acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionItems)
   1. [La elección de sus puntos de acción](http://developer.android.com/guide/topics/ui/actionbar.html#ChoosingActionItems)
   2. [Usar barra de división de acción](http://developer.android.com/guide/topics/ui/actionbar.html#SplitBar)
3. [Utilizando el icono de la aplicación para la navegación](http://developer.android.com/guide/topics/ui/actionbar.html#Home)
   1. [Navegando por](http://developer.android.com/guide/topics/ui/actionbar.html#Up)
4. [Adición de una vista de acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionView)
   1. [Manejo de puntos de vista plegables de acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionViewCollapsing)
5. [Adición de un proveedor de Acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionProvider)
   1. [Usando el ShareActionProvider](http://developer.android.com/guide/topics/ui/actionbar.html#ShareActionProvider)
   2. [Creación de un proveedor de acción personalizada](http://developer.android.com/guide/topics/ui/actionbar.html#CreatingActionProvider)
6. [Agregar fichas de navegación](http://developer.android.com/guide/topics/ui/actionbar.html#Tabs)
7. [Añadir desplegable de navegación](http://developer.android.com/guide/topics/ui/actionbar.html#Dropdown)
8. [El estilo de la barra de acciones](http://developer.android.com/guide/topics/ui/actionbar.html#Style)
   1. [Aspecto general](http://developer.android.com/guide/topics/ui/actionbar.html#GeneralStyles)
   2. [Los temas de acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionItemStyles)
   3. [Fichas de navegación](http://developer.android.com/guide/topics/ui/actionbar.html#NavigationStyles)
   4. [Lista desplegable](http://developer.android.com/guide/topics/ui/actionbar.html#DropDownStyles)
   5. [Diseño avanzado](http://developer.android.com/guide/topics/ui/actionbar.html#AdvancedStyles)

**Clases de clave**

1. [ActionBar](http://developer.android.com/reference/android/app/ActionBar.html)
2. [Menu](http://developer.android.com/reference/android/view/Menu.html)
3. [ActionProvider](http://developer.android.com/reference/android/view/ActionProvider.html)

**Muestras relacionadas**

1. [Galería de nido de abeja](http://developer.android.com/resources/samples/HoneycombGallery/index.html)
2. [Acción de compatibilidad de Abogados](http://developer.android.com/resources/samples/ActionBarCompat/index.html)
3. [Demos de la API](http://developer.android.com/resources/samples/ApiDemos/src/com/example/android/apis/app/index.html#ActionBar)

**Véase también**

1. [Diseño de Android: Barra de acción](http://developer.android.com/design/patterns/actionbar.html)
2. [Menús](http://developer.android.com/guide/topics/ui/menus.html)
3. [Tabletas de apoyo y Móviles](http://developer.android.com/guide/practices/tablets-and-handsets.html)

La barra de acción es una característica de la ventana que identifica la aplicación y la ubicación del usuario, y proporciona las acciones del usuario y los modos de navegación. Se debe utilizar la barra de acción en la mayoría de las actividades que deben presentar de forma destacada las acciones del usuario o la navegación mundial, debido a que la barra de acción ofrece a los usuarios una interfaz coherente en todas las aplicaciones y el sistema se adapta con elegancia la apariencia de la barra de acción para configuraciones de pantalla diferentes. Usted puede controlar los comportamientos y la visibilidad de la barra de acción con el [ActionBar](http://developer.android.com/reference/android/app/ActionBar.html) API, que se agregaron en Android 3.0 (nivel de la API 11).

Los objetivos principales de la barra de acción son los siguientes:

* Proporcionar un espacio dedicado a la identificación de la marca de la aplicación y la ubicación del usuario.

Esto se logra con el icono de la aplicación o logotipo en el lado izquierdo y el título de la actividad. Usted puede optar por retirar el título de la actividad, sin embargo, si la vista actual se identifica por una etiqueta de navegación, tales como la pestaña seleccionada.

* Proporcionar una navegación consistente y ver el refinamiento través de diferentes aplicaciones.

La barra de acción proporciona un sistema integrado de navegación ficha para cambiar entre los [fragmentos](http://developer.android.com/guide/topics/fundamentals/fragments.html) . También ofrece una lista desplegable se puede utilizar como una alternativa modo de navegación o para afinar la vista actual (por ejemplo, para ordenar una lista por diferentes criterios).

* Realiza acciones clave para la actividad (por ejemplo, "búsqueda", "cree", "compartir", etc) destacados y accesibles para el usuario de una manera predecible.

Puede proporcionar acceso instantáneo a las acciones del usuario mediante la colocación de los elementos clave de la [opciones de menú](http://developer.android.com/guide/topics/ui/menus.html#OptionsMenu) directamente en la barra de acción, como "puntos de acción." Los temas de acción también puede proporcionar una "visión de acción", que proporciona un widget integrado por conductas de acción más inmediatos. Los elementos de menú que no sean promovidos a un elemento de acción están disponibles en el menú de desbordamiento, reveló, ya sea el botón de *menú del* dispositivo (si está disponible) o por un "menú de desbordamiento" en la barra de acción (cuando el dispositivo no incluye un botón de *Menú* ).

http://developer.android.com/images/ui/actionbar.png

**Figura 1.** Barra de acciones de la [Galería de nido de abeja](http://developer.android.com/resources/samples/HoneycombGallery/index.html) aplicación (en el auricular del paisaje), que muestra el logo a la izquierda, pestañas de navegación, y un elemento de acción de la derecha (más el botón de menú de desbordamiento).

**Nota:** Si usted está buscando para obtener información sobre la barra de acciones contextuales para la visualización de los elementos contextuales de acción, ver el [menú](http://developer.android.com/guide/topics/ui/menus.html#context-menu) guía.

**Acción de Diseño de Abogados**

Para conocer las directrices de diseño, leer Diseño de Android [Barra de acción del](http://developer.android.com/design/patterns/actionbar.html) guía.

**Restante compatible con versiones anteriores**

Si desea proporcionar una barra de acción en su solicitud *y* siguen siendo compatibles con las versiones de Android mayor que 3,0, lo que necesita para crear la barra de acción en el diseño de su actividad (debido a que el [ActionBar](http://developer.android.com/reference/android/app/ActionBar.html) la clase no está disponible en las versiones anteriores).

Para ayudarle, la [barra de acciones de compatibilidad](http://developer.android.com/resources/samples/ActionBarCompat/index.html) aplicación de ejemplo proporciona un nivel de API y el diseño de la barra de acción que permite a su aplicación para utilizar algunos de los [ActionBar](http://developer.android.com/reference/android/app/ActionBar.html) API y también apoyar a las versiones anteriores de Android mediante la sustitución de la barra de título tradicional con un diseño de barra de acción personalizado.

**Adición de la Barra de acciones**

Empezando con Android 3.0 (nivel de la API 11), la barra de acción está incluido en todas las actividades que utilizan el [Theme.Holo](http://developer.android.com/reference/android/R.style.html#Theme_Holo) tema (o uno de sus descendientes), que es el tema por defecto, cuando ya sea el [targetSdkVersion](http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#target) o [minSdkVersion](http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#min) atributo se establece en "11" o mayor. Por ejemplo:

<Manifiesta ... >

<Usos-SDK de Android: minSdkVersion = "4"

**android: targetSdkVersion = "11"** />

...

</ Manifest>

En este ejemplo, la aplicación requiere una versión mínima del nivel de la API 4 (Android 1.6), pero también apunta a nivel de la API 11 (Android 3.0). De esta manera, cuando la aplicación se ejecuta en Android 3.0 o superior, el sistema aplica el tema holográfico para cada actividad, y por lo tanto, cada actividad incluye la barra de acción.

Si desea utilizar [ActionBar](http://developer.android.com/reference/android/app/ActionBar.html) API, como para añadir los modos de navegación y modificar estilos de barra de acción, debe establecer la [minSdkVersion](http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#min) al "11" o mayor. Si desea que su aplicación para apoyar a las versiones anteriores de Android, hay maneras de utilizar un conjunto limitado de [ActionBar](http://developer.android.com/reference/android/app/ActionBar.html) API en dispositivos compatibles con el nivel API de 11 o superior, mientras que aún se ejecuta en las versiones anteriores. Consulte la información acerca de sidebox restante compatible con versiones anteriores.

**Extracción de la barra de acción**

Si usted no desea que la barra de acción para una actividad particular, definir el tema de la actividad de [Theme.Holo.NoActionBar](http://developer.android.com/reference/android/R.style.html#Theme_Holo_NoActionBar) . Por ejemplo:

<Actividad de Android: tema = "@ android: estilo / tema.Holo.NoActionBar ">

También puede ocultar la barra de acciones en tiempo de ejecución llamando a [hide()](http://developer.android.com/reference/android/app/ActionBar.html#hide%28%29) . Por ejemplo:

Actionbar actionbar = [getActionBar()](http://developer.android.com/reference/android/app/Activity.html#getActionBar%28%29) ;

actionBar.hide ();

Cuando se oculta la barra de acciones, el sistema se ajusta a su diseño de actividad para llenar todo el espacio de pantalla disponible. Usted puede traer a la barra de acción vuelve con [show()](http://developer.android.com/reference/android/app/ActionBar.html#show%28%29) .

Tenga en cuenta que ocultar y quitar la barra de acción hace que su actividad de re-diseño con el fin de explicar el espacio consumido por la barra de acción. Si su actividad regularmente se esconde y muestra la barra de acción (como en la aplicación Android Gallery), es posible que desee utilizar el modo de superposición. Superposición de modo señala a la barra de acción en la parte superior de su diseño de la actividad en lugar de en su propia área de la pantalla. De esta manera, el diseño permanece fija cuando se oculta la barra de acciones y re-aparece. Para activar el modo de superposición, crear un tema para su actividad y establecer [android:windowActionBarOverlay](http://developer.android.com/reference/android/R.attr.html#windowActionBarOverlay) de true . Para obtener más información, consulte la sección acerca de [estilos a la barra de acción](http://developer.android.com/guide/topics/ui/actionbar.html#Style) .

**Consejo:** Si tienes un tema personalizado en el que la actividad que le gustaría quitar la barra de acción, establecer el [android:windowActionBar](http://developer.android.com/reference/android/R.styleable.html#Theme_windowActionBar) propiedad de estilo de false . Sin embargo, si se quita la barra de acción con un tema, a continuación, la ventana no permitirá que la barra de acción en absoluto, por lo que no puede añadirlo más tarde-llamado [getActionBar()](http://developer.android.com/reference/android/app/Activity.html#getActionBar%28%29) devolverá un valor nulo.

**Agregar elementos de acción**

A veces es posible que desee dar a los usuarios acceso inmediato a un elemento de la [menú de opciones](http://developer.android.com/guide/topics/ui/menus.html#OptionsMenu) . Para ello, se puede declarar que el elemento de menú aparecerá en la barra de acción como un "elemento de acción." Un elemento de acción puede incluir un icono y / o un título de texto. Si un elemento de menú no aparece como un elemento de acción, entonces el sistema lo coloca en el menú de desbordamiento. El menú de desbordamiento se revela, ya sea por el botón de *menú del* dispositivo (si es proporcionado por el dispositivo) o un botón adicional en la barra de acción (si el dispositivo no ofrece el botón de *menú).*

http://developer.android.com/images/ui/actionbar-item-withtext.png

**Figura 2.** Dos elementos de acción con títulos de los iconos y el texto y el botón de menú de desbordamiento.

Cuando la primera actividad se inicia, el sistema rellena la barra de acción y el menú de desbordamiento llamando [onCreateOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#onCreateOptionsMenu%28android.view.Menu%29) para su actividad. Como se discutió en el [Menús](http://developer.android.com/guide/topics/ui/menus.html) guía para el desarrollador, es en este método de devolución de llamada que se debe inflar un XML [recurso de menú](http://developer.android.com/guide/topics/resources/menu-resource.html) que define los elementos del menú. Por ejemplo:

@ Override

onCreateOptionsMenu pública booleano (menú Menú) {

MenuInflater inflater getMenuInflater = ();

inflater.inflate (R.menu.main\_activity, menú);

return true;

}

En el archivo XML, se puede solicitar un elemento de menú para que aparezca como un elemento de acción, declarando android:showAsAction="ifRoom" para el <item> elemento. De esta manera, el elemento de menú aparece en la barra de acción para un acceso rápido sólo *si hay espacio* disponible. Si no hay suficiente espacio, el elemento aparece en el menú de desbordamiento.

Si el elemento de menú proporciona un título y un icono-con el android:title y android:icon los atributos, entonces el elemento de acción muestra sólo el icono por defecto. Si desea mostrar el título del texto, añadir "withText" a la android:showAsAction atributo. Por ejemplo:

<?xml version = "1.0" encoding = "UTF-8"?>

<Menú xmlns:android="http://schemas.android.com/apk/res/android">

<Tema android: id = "+ id @ / menu\_save"

android: icon = "@ dibujable / ic\_menu\_save"

android: title = "@ cadena / menu\_save"

**android: showAsAction = "ifRoom | withText"** />

</ Menu>

**Nota:** El "withText" valor es una *pista* a la barra de acción que el título del texto debe aparecer. La barra de acción mostrará el título cuando sea posible, pero tal vez no, si está disponible un icono y la barra de acción está limitada por el espacio.

Cuando el usuario selecciona un elemento de acción, su actividad recibe una llamada a [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) , pasando por la identificación proporcionada por el android:id de atributos de la misma devolución de llamada recibida para todos los elementos en el menú de opciones.

Es importante que usted defina siempre android:title de cada elemento de menú, incluso si usted no declara que el título aparecerá con la acción del elemento-por tres razones:

* Si no hay suficiente espacio en la barra de acción para el elemento de acción, el elemento de menú aparece en el menú de desbordamiento y sólo aparece el título.
* Los lectores de pantalla para usuarios con deficiencias visuales leer el título del elemento de menú.
* Si el elemento de acción aparece con sólo el icono, el usuario puede pulsar el largo tema para revelar una información sobre herramientas que muestra el título del artículo de acción.

El android:icon es siempre opcional, pero recomendable. Por recomendaciones de diseño de iconos, ver las [de acción icono de la barra](http://developer.android.com/guide/practices/ui_guidelines/icon_design_action_bar.html) pautas de diseño.

**Nota:** Si ha agregado el elemento de menú de un fragmento, a través del [Fragment](http://developer.android.com/reference/android/app/Fragment.html) la clase de [onCreateOptionsMenu](http://developer.android.com/reference/android/app/Fragment.html#onCreateOptionsMenu%28android.view.Menu,%20android.view.MenuInflater%29) de devolución de llamada, el sistema llama a la respectiva [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Fragment.html#onOptionsItemSelected%28android.view.MenuItem%29) método para que el fragmento cuando el usuario selecciona uno de los elementos del fragmento. Sin embargo, la actividad tiene la oportunidad de controlar el evento en primer lugar, por lo que el sistema llama [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) sobre la actividad antes de llamar a la devolución de llamada para el mismo fragmento.

También puede declarar un elemento para *"siempre"* aparecen como un elemento de acción, en lugar de colocarse en el menú de desbordamiento cuando el espacio es limitado. En la mayoría de los casos, **no** se **debe** forzar a un elemento que aparezca en la barra de acción mediante el "always" valor. Sin embargo, es posible que tenga un elemento para que aparezca siempre cuando se ofrece una [visión de acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionView) que no ofrece una acción predeterminada para el menú de desbordamiento. Tenga en cuenta que los elementos de acción demasiados puede crear una interfaz de usuario desordenado y causar problemas de diseño en los dispositivos con una pantalla estrecha. Lo mejor es usar en su lugar "ifRoom" para solicitar que un artículo aparezca en la barra de acción, pero permitir que el sistema se mueve en el menú de desbordamiento cuando no hay suficiente espacio.

Para obtener más información sobre cómo crear el menú de opciones que define sus elementos de acción, consulte la [Menús](http://developer.android.com/guide/topics/ui/menus.html#options-menu) guía para el desarrollador.

**La elección de sus puntos de acción**

**Los elementos del menú de aplicaciones frente a los controles de otros**

Como regla general, todos los elementos del [menú de opciones](http://developer.android.com/guide/topics/ui/menus.html#OptionsMenu) (por no hablar de los puntos de acción) deben tener un impacto global en la aplicación, en lugar de afectar a sólo una pequeña parte de la interfaz. Por ejemplo, si usted tiene un diseño de panel multi-y un panel se muestra un vídeo, mientras que otra muestra todos los vídeos, los controles del reproductor de vídeo debe aparecer en el panel que contiene el video (no en la barra de acciones), mientras que la barra de acción podría proporcionar una acción artículos para compartir el video o guardar el vídeo a una lista de favoritos.

Así, incluso antes de decidir si un elemento de menú debe aparecer como un elemento de acción, asegúrese de que el tema tiene un alcance global para la actividad actual. Si no es así, entonces usted debe colocarlo como un botón en el contexto apropiado de la distribución de la actividad.

Usted debe elegir con cuidado qué elementos de su menú de opciones debe aparecer como puntos de acción mediante la evaluación de algunos rasgos fundamentales. En general, cada elemento de acción debe ser *por lo menos uno* de los siguientes:

1. **Utilizado con frecuencia:** Es una acción que los usuarios necesitan siete de cada diez visitas o que lo utilizan varias veces en una fila.

Ejemplos frecuentes acciones: "Nuevo mensaje" en la aplicación de mensajería y "Buscar" en la reproducción de Google.

1. **Importante:** Es una acción que es necesario que los usuarios descubran fácilmente o, si no se utiliza con frecuencia, es importante que sea fácil de realizar en los pocos casos en que los usuarios lo necesitan.

Ejemplo acciones importantes: "Agregar red" en la configuración Wi-Fi y "Cambiar a cámara" en la aplicación Galería.

1. **Típico:** Es una acción que se proporciona normalmente en la barra de acción en aplicaciones similares, por lo que sus usuarios esperan encontrar en el tuyo.

Ejemplos típicos de las acciones: "Actualizar" en una aplicación de correo electrónico o social, y "Nuevo contacto" en la aplicación de la gente.

Si usted cree que más de cuatro de sus elementos de menú se puede justificar como elementos de acción, entonces usted debe considerar cuidadosamente su nivel de importancia relativa y tratar de fijar no más de cuatro como elementos de acción (y lo hacen con el "ifRoom" valor a permitir que el sistema de poner un poco de nuevo en el menú de desbordamiento cuando el espacio es limitado en pantallas más pequeñas). Incluso si hay espacio disponible en una amplia pantalla, no se debe crear una larga secuencia de puntos de acción que el desorden de la interfaz de usuario y aparecen como una barra de herramientas de escritorio, a fin de mantener el número de elementos de acción a un mínimo.

Además, las siguientes acciones nunca deben aparecer como puntos de acción: Ajustes, Ayuda, sugerencias o similares. Siempre que permanezcan en el menú de desbordamiento.

**Nota:** Recuerde que no todos los dispositivos cuentan con un botón de hardware dedicado para la búsqueda, por lo que si es una característica importante en su aplicación, siempre debe aparecer como un elemento de acción (y por lo general como primer tema, sobre todo si se ofrecen con una [vista de la acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionView) ).

**Usar barra de división de acción**

Cuando se ejecuta la aplicación en Android 4.0 (nivel de la API 14) y superior, hay un modo extra disponibles para la barra de acción llamada "barra de acción de división". Cuando se habilita barra de división acción, un bar separado en la parte inferior de la pantalla para mostrar todos los elementos de acción cuando la actividad se está ejecutando en una pantalla estrecha (por ejemplo, un teléfono con orientación vertical). La división de la barra de acción para separar los elementos de acción asegura que una cantidad razonable de espacio disponible para mostrar todos los elementos de acción en una pantalla estrecha, dejando espacio para la navegación y elementos de título en la parte superior.

Para activar la barra de acción de división, sólo tiene que añadir uiOptions="splitActionBarWhenNarrow" a sus [<activity>](http://developer.android.com/guide/topics/manifest/activity-element.html) o [<application>](http://developer.android.com/guide/topics/manifest/application-element.html) elemento manifiesto.

Tenga en cuenta que Android ajusta la apariencia de la barra de acción en una variedad de formas, basado en el tamaño de la pantalla actual. Usar barra de división acción es sólo una opción que se puede habilitar para permitir que la barra de acción para optimizar aún más la experiencia de usuario para diferentes tamaños de pantalla. Al hacerlo, usted también puede permitir que la barra de acción para cerrar las pestañas de navegación en la barra de la acción principal. Es decir, si utiliza [pestañas de navegación](http://developer.android.com/guide/topics/ui/actionbar.html#Tabs) en tu barra de acción, una vez que los puntos de acción se separan en una pantalla estrecha, las fichas de navegación puede ser capaz de encajar en la barra de acción principal en lugar de separarse en la "barra de acción apilados". En concreto, si ha desactivado el icono de la barra de acciones y el título (con [setDisplayShowHomeEnabled(false)](http://developer.android.com/reference/android/app/ActionBar.html#setDisplayShowHomeEnabled%28boolean%29) y [setDisplayShowTitleEnabled(false)](http://developer.android.com/reference/android/app/ActionBar.html#setDisplayShowTitleEnabled%28boolean%29) ), luego del colapso de navegación pestañas en la barra de acción principal, como se muestra en el segundo dispositivo en la figura 3.



**Figura 3.** Las maquetas de barra de acción dividida con pestañas de navegación a la izquierda; con el icono de la aplicación y el título de discapacitados a la derecha.

**Nota:** Aunque el [android:uiOptions](http://developer.android.com/reference/android/R.attr.html#uiOptions) atributo se añadió en Android 4.0 (nivel de la API 14), que con seguridad puede incluir en su solicitud, incluso si su [minSdkVersion](http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#min) se establece en un valor inferior a "14" siga siendo compatible con versiones anteriores de Android. Cuando se ejecuta en las versiones anteriores, el sistema simplemente ignora el atributo XML, ya que no lo entiendo. La única condición para su inclusión en el manifiesto es que se debe compilar la aplicación en contra de una versión de la plataforma que apoya a nivel de API de 14 o superior. Sólo asegúrese de que usted no usa abiertamente otras API en su código de aplicación que no son compatibles con la versión declarada por el [minSdkVersion](http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#min) atributo de sólo los atributos XML de forma segura ignorado por plataformas más antiguas.

**Utilizando el icono de la aplicación para la navegación**

**El uso de un logotipo en lugar del icono**

Por defecto, el sistema utiliza el icono de la aplicación en la barra de acción, según lo especificado por el [android:icon](http://developer.android.com/guide/topics/manifest/application-element.html#icon) atributo en el [<application>](http://developer.android.com/guide/topics/manifest/application-element.html) o [<activity>](http://developer.android.com/guide/topics/manifest/activity-element.html) elemento. Sin embargo, si también se especifica el [android:logo](http://developer.android.com/guide/topics/manifest/application-element.html#logo) atributo, entonces la barra de acción utiliza la imagen del logotipo en lugar del icono.

Un logotipo debe ser generalmente más ancho que el icono, pero no debe incluir el texto innecesario. En general, usted debe utilizar un logo sólo cuando representa su marca en un formato tradicional de que los usuarios a reconocer. Un buen ejemplo es el logotipo de la aplicación de YouTube logo representa la marca de usuario esperado, mientras que el icono de la aplicación es una versión modificada que cumpla con el requisito de plaza.

Por defecto, el icono de la aplicación aparece en la barra de acción en el lado izquierdo. Si lo desea, puede activar el icono de comportarse como un elemento de acción. En respuesta a una acción del usuario sobre el icono, la aplicación debe hacer una de dos cosas:

* Vaya a la aplicación "casa" la actividad, o
* Navegar hacia "arriba" la jerarquía estructural de la aplicación

Cuando el usuario toca el icono, el sistema llama a su actividad de [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) método con el android.R.id.home ID. En respuesta, usted debe comenzar la actividad en casa o llevar al usuario a un paso en la jerarquía estructural de la aplicación.

Si usted responde a la icono de la aplicación por volver a la actividad en el hogar, usted debe incluir el [FLAG\_ACTIVITY\_CLEAR\_TOP](http://developer.android.com/reference/android/content/Intent.html#FLAG_ACTIVITY_CLEAR_TOP) bandera en la [Intent](http://developer.android.com/reference/android/content/Intent.html) . Con este indicador, si la actividad que está empezando ya existe en la tarea actual, a continuación, todas las actividades en la parte superior de la misma se destruyen y se trajo a primer plano. La adición de este indicador es a menudo importante, porque va a "casa" es una acción que es equivalente a "volver" y no debe por lo general crear una nueva instancia de la actividad en el hogar. De lo contrario, podría terminar con una pila larga de actividades en la tarea actual, con varias instancias de la actividad en el hogar.

Por ejemplo, aquí hay una implementación de [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) que devuelve a la aplicación de la "casa" la actividad:

@ Override

public boolean onOptionsItemSelected (elemento MenuItem) {

switch (item.getItemId ()) {

caso de Android.R.id.home:

/ / Icono de la aplicación en la barra de acción de hacer clic, ir a casa

Intención de Intención = nuevo intento (esto, HomeActivity.class);

intent.addFlags (Intent.FLAG\_ACTIVITY\_CLEAR\_TOP);

startActivity (intención);

return true;

default:

volver super.onOptionsItemSelected (punto);

}

}

En caso de que el usuario puede introducir la actividad actual desde otra aplicación, es posible que también desee agregar el [FLAG\_ACTIVITY\_NEW\_TASK](http://developer.android.com/reference/android/content/Intent.html#FLAG_ACTIVITY_NEW_TASK) bandera. Esta bandera se asegura de que, cuando el usuario navega bien "casa" o "arriba", la nueva actividad **no** se agrega a la tarea actual, sino que se inició en una tarea que pertenece a su aplicación. Por ejemplo, si el usuario inicia una actividad en su aplicación a través de la intención invocada por otra aplicación, a continuación, selecciona el icono de la barra de acción para navegar en casa o arriba, el [FLAG\_ACTIVITY\_CLEAR\_TOP](http://developer.android.com/reference/android/content/Intent.html#FLAG_ACTIVITY_CLEAR_TOP) bandera comienza la actividad en una tarea que pertenece a su aplicación (no el actual tarea). El sistema o empieza una nueva tarea con su nueva actividad como la actividad de las raíces o, si existe una tarea existente en el fondo con una instancia de esa actividad, entonces esa tarea se lleva adelante y la actividad de destino recibe [onNewIntent()](http://developer.android.com/reference/android/app/Activity.html#onNewIntent%28android.content.Intent%29) . Así que si su actividad acepta los intentos de otras aplicaciones (que declara la intención de los filtros genéricos), por lo general, debe agregar el [FLAG\_ACTIVITY\_NEW\_TASK](http://developer.android.com/reference/android/content/Intent.html#FLAG_ACTIVITY_NEW_TASK) bandera a la intención de:

intent.addFlags (Intent.FLAG\_ACTIVITY\_CLEAR\_TOP | Intención.FLAG\_ACTIVITY\_NEW\_TASK);

Para obtener más información acerca de estos indicadores y otros comportamientos de la pila de espalda, lea las [tareas y Volver Stack](http://developer.android.com/guide/topics/fundamentals/tasks-and-back-stack.html) guía para el desarrollador.

**Nota:** Si usted está usando el icono para ir a la actividad en el hogar, ten en cuenta que inicio con Android 4.0 (nivel de la API 14), debe habilitar explícitamente el icono como un elemento de acción llamando a [setHomeButtonEnabled(true)](http://developer.android.com/reference/android/app/ActionBar.html#setHomeButtonEnabled%28boolean%29) (en versiones anteriores, la icono ha sido habilitado como un elemento de acción por defecto).

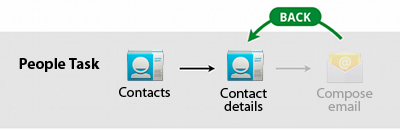
**Navegando por**



**Figura 4.** Icono estándar de la aplicación E-mail (a la izquierda) y el "navegar hacia arriba" icono (derecha). El sistema añade automáticamente el "arriba" del indicador.

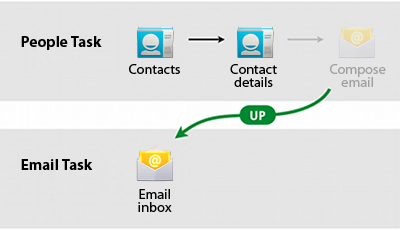
Como complemento a la tradicional "vuelta" de navegación, lo cual lleva al usuario a la pantalla anterior en la tarea de la historia, se puede activar el icono de la barra de acción para ofrecer "arriba" de navegación, que debe llevar al usuario un paso adelante en la jerarquía estructural de la aplicación . Por ejemplo, si la pantalla actual es en algún lugar profundo en la jerarquía de la aplicación, tocando el icono de la aplicación debe navegar hacia arriba un nivel, a los padres de la pantalla actual.

Por ejemplo, la figura 5 se muestra cómo se comporta el botón BACK cuando el usuario navega de una aplicación a una actividad que pertenece a una aplicación diferente (en concreto, al redactar un correo electrónico a una persona seleccionada de la aplicación de Personas).



**Figura 5.** El comportamiento del botón VOLVER después de entrar en la aplicación de correo electrónico de las personas (o contactos) de aplicaciones.

Sin embargo, si el usuario quiere permanecer dentro de la aplicación de correo electrónico después de componer la dirección de correo electrónico, hasta la navegación permite al usuario navegar hacia arriba en la aplicación de correo electrónico, en lugar de volver a la actividad anterior. La figura 6 muestra este escenario, en el que el usuario nuevo entra en la aplicación de correo electrónico, pero presiona el icono de la barra de acción para desplazarse hacia arriba, no hacia atrás.



**Figura 6.** Ejemplo de comportamiento de navegación hacia arriba después de entrar en la aplicación de correo electrónico de la aplicación de la gente.

**Diseño de navegación**

Para más información sobre cómo *arriba* y hacia *atrás* de navegación diferentes, leer Android Diseño de [navegación](http://developer.android.com/design/patterns/navigation.html) guía.

Para activar el icono para un máximo de navegación (que muestra el "arriba" indicador situado junto al icono), llame a [setDisplayHomeAsUpEnabled(true)](http://developer.android.com/reference/android/app/ActionBar.html#setDisplayHomeAsUpEnabled%28boolean%29) en su [ActionBar](http://developer.android.com/reference/android/app/ActionBar.html) :

protected void onCreate (paquete savedInstanceState) {

super.onCreate (savedInstanceState);

setContentView (R.layout.main);

Actionbar actionbar getActionBar = ();

actionBar.setDisplayHomeAsUpEnabled (true);

...

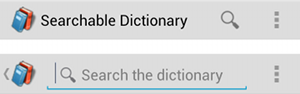
}

Cuando el usuario toca el icono, el sistema llama a su actividad de [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) método con el android.R.id.home ID, como se muestra en la sección anterior acerca de [utilizar el icono de aplicación para la navegación](http://developer.android.com/guide/topics/ui/actionbar.html#Home) .

Recuerde que debe utilizar el [FLAG\_ACTIVITY\_CLEAR\_TOP](http://developer.android.com/reference/android/content/Intent.html#FLAG_ACTIVITY_CLEAR_TOP) bandera en la [Intent](http://developer.android.com/reference/android/content/Intent.html) , de modo que usted no crea una nueva instancia de la actividad de los padres si ya existe uno. Por ejemplo, si usted no utiliza el [FLAG\_ACTIVITY\_CLEAR\_TOP](http://developer.android.com/reference/android/content/Intent.html#FLAG_ACTIVITY_CLEAR_TOP) bandera, a continuación, después de navegar hacia arriba, el botón de nuevo en realidad llevará al usuario "adelante", con respecto a la estructura de la aplicación, lo que sería extraño.

**Nota:** Si hay muchos caminos para que el usuario podría haber tenido para llegar a la actividad actual dentro de la aplicación, el icono de arriba debe desplazarse hacia atrás a lo largo de la ruta de acceso que el usuario realmente seguido para llegar a la actividad actual.

**Adición de una vista de acción**



**Figura 7.** Una barra de acción con el fin de la acción se derrumbó para la búsqueda (arriba), vista la acción luego se amplió con la [SearchView](http://developer.android.com/reference/android/widget/SearchView.html) widget (parte inferior).

Una vista de la acción es un widget que aparece en la barra de acciones como un sustituto para el botón de un elemento de acción. Por ejemplo, si usted tiene un elemento en el menú de opciones de "Buscar", se puede añadir un punto de vista la acción que reemplaza el botón con un [SearchView](http://developer.android.com/reference/android/widget/SearchView.html) widget, como se muestra en la figura 7.

Para declarar un punto de vista de acción para un artículo en su [recurso de menú](http://developer.android.com/guide/topics/resources/menu-resource.html) , utilice el android:actionLayout o android:actionViewClass atributo para especificar un recurso de diseño o de clase widget a utilizar, respectivamente. Por ejemplo:

<?xml version = "1.0" encoding = "UTF-8"?>

<Menú xmlns:android="http://schemas.android.com/apk/res/android">

<Tema android: id = "+ id @ / menu\_search"

android: title = "@ cadena / menu\_search"

android: icon = "@ dibujable / ic\_menu\_search"

android: showAsAction = "ifRoom | collapseActionView"

**android: actionViewClass = "android.widget.SearchView "/>**

</ Menu>

Observe que el android:showAsAction atributo también incluye "collapseActionView" . Esto es opcional y declara que el punto de vista la acción debe ser derrumbado en un botón. Cuando el usuario selecciona el botón, la vista de la acción se expande. De lo contrario, la opinión de la acción es visible por defecto y puede consumir un valioso espacio barra de acción, incluso cuando el usuario no lo está utilizando. Para obtener más información, consulte la siguiente sección sobre [Manejo de puntos de vista plegables de acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionViewCollapsing) .

Si es necesario agregar algunos ganchos de eventos a la vista la acción, puede hacerlo durante el [onCreateOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#onCreateOptionsMenu%28android.view.Menu%29) de devolución de llamada. Usted puede adquirir los elementos en una vista de la acción llamando a [findItem()](http://developer.android.com/reference/android/view/Menu.html#findItem%28int%29) con el ID del elemento de menú, a continuación, llamar [getActionView()](http://developer.android.com/reference/android/view/MenuItem.html#getActionView%28%29) . Por ejemplo, el widget de búsqueda en el ejemplo anterior se adquiere de esta manera:

@Override

onCreateOptionsMenu pública booleano (menú Menú) {

. getMenuInflater () se inflan (R.menu.options, menú);

SearchView SearchView = (SearchView) menu.findItem (R.id.menu\_search) getActionView ().;

/ / Configurar la información de búsqueda y añadir cualquier detector de eventos

...

retorno super.onCreateOptionsMenu (menú);

}

Para obtener más información sobre cómo utilizar el widget de búsqueda, consulte [Creación de una interfaz de búsqueda](http://developer.android.com/guide/topics/search/search-dialog.html) .

**Manejo de puntos de vista plegables de acción**

**Apoyo a Android 3.0 con una visión de acción**

El "collapseActionView" opción se añadió con Android 4.0 (nivel de la API 14). Sin embargo, si su aplicación es compatible con versiones anteriores, aún debe declarar "collapseActionView" con el fin de apoyar mejor a pantallas más pequeñas. Dispositivos con Android 4.0 y superior se mostrará el punto de vista la acción se derrumbó, mientras que las versiones anteriores funciona como fue diseñado de otra manera.

La adición de este valor, es necesario configurar el destino de generación de Android 4.0 o superior para poder compilar. Las versiones anteriores de Android caso omiso de la "collapseActionView" valor porque no lo entiendo. Sólo asegúrese de no utilizar otras API en su código fuente que no se admiten en la versión declarada por el [minSdkVersion](http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#min) , a menos que usted agregue la comprobación de la versión adecuada en tiempo de ejecución.

Puntos de vista de acción le permiten proporcionar un acceso rápido a las acciones de ricos sin necesidad de cambiar actividades o fragmentos, o sustitución de la barra de acción. Sin embargo, podría no ser apropiada para hacer un punto de vista la acción visible de forma predeterminada. A fin de preservar el espacio de la barra de acción (especialmente cuando se ejecuta en pantallas más pequeñas), puede contraer su punto de vista la acción como un control de elemento de acción. Cuando el usuario selecciona el botón, la vista de la acción aparece en la barra de acción. Cuando se derrumbó, el ​​sistema puede colocar el elemento en el menú de desbordamiento si se ha definido android:showAsAction con "ifRoom" , pero la vista la acción sigue apareciendo en la barra de acción cuando el usuario selecciona el elemento. Usted puede hacer su acción de ver plegable mediante la adición de "collapseActionView" a la android:showAsAction atributo, como se muestra en el XML de arriba.

Debido a que el sistema se amplíe la vista de la acción cuando el usuario selecciona el elemento, por lo que *no es* necesario para responder a el tema en el [onOptionsItemSelected](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) de devolución de llamada. El sistema todavía llama [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) cuando el usuario lo selecciona, pero el sistema siempre se amplíe la vista de la acción a menos que devuelva true (lo que indica que usted ha manejado el caso en su lugar).

El sistema también se derrumba la acción de la vista cuando el usuario selecciona el "arriba" icono en la barra de acción o pulsa el botón BACK.

Si es necesario, puede expandir o contraer la vista de la acción en su propio código llamando [expandActionView()](http://developer.android.com/reference/android/view/MenuItem.html#expandActionView%28%29) y [collapseActionView()](http://developer.android.com/reference/android/view/MenuItem.html#collapseActionView%28%29) en el [MenuItem](http://developer.android.com/reference/android/view/MenuItem.html) .

**Nota:** Aunque el colapso de su punto de vista la acción es opcional, le recomendamos que siempre se derrumban a su punto de vista la acción si se incluye [SearchView](http://developer.android.com/reference/android/widget/SearchView.html) . También tenga en cuenta que algunos dispositivos cuentan con un botón de búsqueda dedicada y debe ampliar su visión de acción de búsqueda si el usuario presiona el botón BUSCAR. Basta con sustituir la actividad de [onKeyUp()](http://developer.android.com/reference/android/app/Activity.html#onKeyUp%28int,%20android.view.KeyEvent%29) el método de devolución de llamada, escuche el [KEYCODE\_SEARCH](http://developer.android.com/reference/android/view/KeyEvent.html#KEYCODE_SEARCH) evento, a continuación, llamar [expandActionView()](http://developer.android.com/reference/android/view/MenuItem.html#expandActionView%28%29) .

Si usted necesita poner al día su actividad en base a la visibilidad de su punto de vista la acción, puede recibir devoluciones de llamada cuando se amplió y se derrumbó por la definición de un [OnActionExpandListener](http://developer.android.com/reference/android/view/MenuItem.OnActionExpandListener.html) y registrarlo con [setOnActionExpandListener()](http://developer.android.com/reference/android/view/MenuItem.html#setOnActionExpandListener%28android.view.MenuItem.OnActionExpandListener%29) . Por ejemplo:

@ Override

onCreateOptionsMenu pública booleano (menú Menú) {

. getMenuInflater () se inflan (R.menu.options, menú);

MenuItem menuitem = menu.findItem (R.id.actionItem);

...

menuItem.setOnActionExpandListener (OnActionExpandListener nuevo () {

@ Override

onMenuItemActionCollapse pública booleano (tema MenuItem) {

/ / Hacer algo cuando se derrumbó

return true; / / Devuelve verdadero colapso de vista de la acción

}

@ Override

public boolean onMenuItemActionExpand (elemento MenuItem) {

/ / Hacer algo cuando se expande

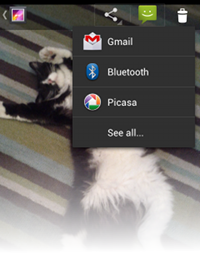
return true; / / Devuelve verdadero punto de vista de ampliar la acción

}

});

}

**Adición de un proveedor de Acción**



**Figura 8.** Captura de pantalla de la aplicación Galería, con la [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) submenú ampliado para mostrar los objetivos de las acciones.

Similar a un [punto de vista la acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionView) , un proveedor de la acción (definido por el [ActionProvider](http://developer.android.com/reference/android/view/ActionProvider.html) clase) reemplaza un elemento de acción, con un diseño personalizado, sino que también toma el control de todos los comportamientos del elemento. Cuando se declara un proveedor de acción para un elemento de menú en la barra de acción, no sólo controla la apariencia del elemento de la barra de acción, con un diseño personalizado, sino que también controla el evento predeterminado para el elemento de menú cuando aparece en el menú de desbordamiento . También puede proporcionar un submenú de cualquiera de la barra de acción o el menú de desbordamiento.

Por ejemplo, el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) es una extensión de [ActionProvider](http://developer.android.com/reference/android/view/ActionProvider.html) que facilita una "parte" de acción, mostrando una lista de objetivos de acciones disponibles en la barra de acción. En lugar de utilizar un elemento de acción tradicional que invoca la [ACTION\_SEND](http://developer.android.com/reference/android/content/Intent.html#ACTION_SEND) intención, se puede declarar una instancia de [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) para manejar un elemento de acción. Este proveedor de la acción presenta una visión de acción con una lista desplegable de aplicaciones que manejan el [ACTION\_SEND](http://developer.android.com/reference/android/content/Intent.html#ACTION_SEND) intención, incluso cuando el elemento de menú aparece en el menú de desbordamiento. Por lo tanto, cuando se utiliza un proveedor de una acción como ésta, usted no tiene que manejar los eventos de usuario en el menú.

Para declarar un proveedor de acción para un elemento de acción, definir el android:actionProviderClass de atributos para el caso, la <item> elemento en su [recurso de menú](http://developer.android.com/guide/topics/resources/menu-resource.html) , utilizando el nombre de clase completo del proveedor de la acción. Por ejemplo:

<?xml version = "1.0" encoding = "UTF-8"?>

<Menú xmlns:android="http://schemas.android.com/apk/res/android">

<Tema android: id = "+ id @ / menu\_share"

android: title = "@ string / share"

android: showAsAction = "ifRoom"

**android: actionProviderClass = "android.widget.ShareActionProvider "/>**

...

</ Menu>

En este ejemplo, el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) se utiliza como el proveedor de acción. En este punto, el proveedor de la acción oficial se hace con el elemento de menú y se ocupa de su aspecto y comportamiento en la barra de acciones y su comportamiento en el menú de desbordamiento. No obstante, debe proporcionar un título de texto para el elemento que se utiliza en el menú de desbordamiento.

Aunque el proveedor de la acción se puede realizar la acción predeterminada para el elemento de menú cuando aparece en el menú de desbordamiento, su actividad (o fragmento) puede reemplazar dicho comportamiento también se controla el evento de clic de la [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) el método de devolución de llamada. Si no controlar el evento en el que la devolución de llamada, entonces el proveedor de la acción recibe el [onPerformDefaultAction()](http://developer.android.com/reference/android/view/ActionProvider.html#onPerformDefaultAction%28%29) de devolución de llamada para controlar el evento. Sin embargo, si el proveedor de acción ofrece un submenú, entonces su actividad no recibirá el [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) de devolución de llamada, ya que el submenú se muestra en lugar de invocar el menú por defecto comportamiento de elemento cuando se selecciona.

**Usando el ShareActionProvider**

Si desea proporcionar un "compartir" la acción en tu barra de acción mediante el aprovechamiento de otras aplicaciones instaladas en el dispositivo (por ejemplo, para compartir una foto con una aplicación de mensajería o social), y luego usando [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) es una manera efectiva de hacerlo, en lugar que la adición de un elemento de acción que invoca el [ACTION\_SEND](http://developer.android.com/reference/android/content/Intent.html#ACTION_SEND) intención. Cuando se utiliza [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) de un elemento de acción, que presenta una visión de acción con una lista desplegable de aplicaciones que manejan el [ACTION\_SEND](http://developer.android.com/reference/android/content/Intent.html#ACTION_SEND) intención (como se muestra en la figura 8).

Toda la lógica de la creación del submenú, rellenarlo con los objetivos de acción, y el manejo de eventos de clic (incluso cuando el tema aparece en el menú de desbordamiento) es implementado por el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) -el único código que necesita para escribir es declarar que el proveedor de acción para la del menú y especifique la calidad de parte.

Por defecto, el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) conserva una clasificación para cada objetivo de cuota sobre la base de la frecuencia con la que el usuario selecciona cada una de ellas. La cuota de los objetivos utilizados con más frecuencia aparecen en la parte superior de la lista desplegable y el objetivo de utilizar con mayor frecuencia aparece directamente en la barra de acciones como el objetivo compartido predeterminado. De forma predeterminada, la información de clasificación se guardan en un archivo privado con un nombre especificado por [DEFAULT\_SHARE\_HISTORY\_FILE\_NAME](http://developer.android.com/reference/android/widget/ShareActionProvider.html#DEFAULT_SHARE_HISTORY_FILE_NAME) . Si utiliza el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) o una extensión de la misma para un solo tipo de acción, entonces usted debe seguir para utilizar este archivo por defecto la historia y no hay nada que usted necesita hacer. Sin embargo, si utiliza [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) o una extensión de la misma para varias acciones con significados semánticamente diferentes, entonces cada [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) debe especificar el archivo de la propia historia con el fin de mantener su propia historia. Para especificar un archivo de historia diferente para el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) , llamada [setShareHistoryFileName()](http://developer.android.com/reference/android/widget/ShareActionProvider.html#setShareHistoryFileName%28java.lang.String%29) y proporcionar un nombre de archivo XML (por ejemplo, "custom\_share\_history.xml" ).

**Nota:** Aunque el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) cuota de rangos objetivos basados ​​en la frecuencia de uso, el comportamiento es extensible y las extensiones de [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) puede llevar a cabo diferentes comportamientos y clasificación basados ​​en el archivo de la historia (si procede).

Para agregar [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) , basta con definir el android:actionProviderClass atributo con "android.widget.ShareActionProvider" , como se muestra en el ejemplo XML anterior. Lo único que queda por hacer es definir la [Intent](http://developer.android.com/reference/android/content/Intent.html) que desea utilizar para el intercambio. Para ello, debe llamar a [getActionProvider()](http://developer.android.com/reference/android/view/MenuItem.html#getActionProvider%28%29) para recuperar el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) que está asociada con un [MenuItem](http://developer.android.com/reference/android/view/MenuItem.html) , a continuación, llamar [setShareIntent()](http://developer.android.com/reference/android/widget/ShareActionProvider.html#setShareIntent%28android.content.Intent%29) .

Si el formato de la intención de acción depende de la opción seleccionada o de otras variables que cambian durante el ciclo de vida de la actividad, debe guardar el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) en un campo de miembro y actualizarlo llamando [setShareIntent()](http://developer.android.com/reference/android/widget/ShareActionProvider.html#setShareIntent%28android.content.Intent%29) según sea necesario. Por ejemplo:

privada mShareActionProvider ShareActionProvider;

...

@ Override

onCreateOptionsMenu pública booleano (menú Menú) {

mShareActionProvider = (ShareActionProvider) menu.findItem (R.id.menu\_share) getActionProvider ().;

/ / Si utiliza más de un ShareActionProvider, cada uno por una acción diferente,

/ / Utilizar la línea siguiente para especificar un archivo de la historia única para cada uno.

/ / MShareActionProvider.setShareHistoryFileName ("custom\_share\_history.xml");

/ / Establecer la intención recurso por defecto

mShareActionProvider.setShareIntent (getDefaultShareIntent ());

return true;

}

/ / Cuando es necesario actualizar la intención de participación de algún otro lugar en la aplicación, llame al

/ / MShareActionProvider. [setShareIntent()](http://developer.android.com/reference/android/widget/ShareActionProvider.html#setShareIntent%28android.content.Intent%29)

El [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) ahora se encarga de toda la interacción del usuario con el tema y *no es* necesario para manejar los eventos de clic de la [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) el método de devolución de llamada.

Para obtener una muestra mediante el proveedor de la acción social, véase [ActionBarShareActionProviderActivity](http://developer.android.com/resources/samples/ApiDemos/src/com/example/android/apis/app/ActionBarShareActionProviderActivity.html) .

**Creación de un proveedor de acción personalizada**

Cuando se desea crear un punto de vista la acción que tiene comportamiento dinámico y una acción predeterminada en el menú de desbordamiento, que se extiende [ActionProvider](http://developer.android.com/reference/android/view/ActionProvider.html) para definir las conductas es una buena solución. Creación de su proveedor de la propia acción que ofrece un componente organizado y reutilizables, en lugar de manejar las transformaciones de acción diferentes elementos y comportamientos en el fragmento o código de actividad. Como se muestra en la sección anterior, Android proporciona una implementación de [ActionProvider](http://developer.android.com/reference/android/view/ActionProvider.html) para las acciones de participación: el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) .

Para crear su cuenta, simplemente se extiende el [ActionProvider](http://developer.android.com/reference/android/view/ActionProvider.html) clase y poner en práctica sus métodos de devolución de llamada, según corresponda. Lo más importante, se deben implementar las siguientes acciones:

[ActionProvider()](http://developer.android.com/reference/android/view/ActionProvider.html#ActionProvider%28android.content.Context%29)

Este constructor se pasa a la aplicación [Context](http://developer.android.com/reference/android/content/Context.html) , que debe guardar en un campo de miembro para utilizar en los métodos de devolución de llamada de otros.

[onCreateActionView()](http://developer.android.com/reference/android/view/ActionProvider.html#onCreateActionView%28%29)

Aquí es donde se define el punto de vista de acción para el elemento. Utilice el [Context](http://developer.android.com/reference/android/content/Context.html) adquiere desde el constructor para crear instancias de un [LayoutInflater](http://developer.android.com/reference/android/view/LayoutInflater.html) e inflar el diseño de vista la acción de un recurso XML, a continuación, conectar detectores de eventos. Por ejemplo:

pública onCreateActionView Vista () {

/ / Inflar el punto de vista la acción que se muestre en la barra de acción.

LayoutInflater layoutInflater = LayoutInflater.from (mContext);

Ver vista = layoutInflater.inflate (R.layout.action\_provider, null);

ImageButton botón = (ImageButton) view.findViewById (R.id.button);

button.setOnClickListener (Ver nueva.OnClickListener () {

@ Override

public void onClick (View v) {

/ / Hacer algo ...

}

});

volver ver;

}

[onPerformDefaultAction()](http://developer.android.com/reference/android/view/ActionProvider.html#onPerformDefaultAction%28%29)

El sistema llama a esto cuando el elemento de menú se selecciona del menú de desbordamiento y el proveedor de la acción debe llevar a cabo una acción predeterminada para el elemento de menú.

Sin embargo, si su proveedor de acción proporciona un submenú, a través de la [onPrepareSubMenu()](http://developer.android.com/reference/android/view/ActionProvider.html#onPrepareSubMenu%28android.view.SubMenu%29) de devolución de llamada, luego el submenú aparece incluso cuando el elemento del menú se encuentra en el menú de desbordamiento. Por lo tanto, [onPerformDefaultAction()](http://developer.android.com/reference/android/view/ActionProvider.html#onPerformDefaultAction%28%29) no se llama cuando hay un submenú.

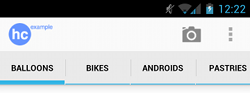
**Nota:** Una actividad o un fragmento que implementa [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) puede reemplazar el comportamiento del proveedor acción predeterminado controlando el evento de elemento seleccionado (y volviendo verdadero), en cuyo caso, el sistema no requiere [onPerformDefaultAction()](http://developer.android.com/reference/android/view/ActionProvider.html#onPerformDefaultAction%28%29) .

Para un ejemplo la extensión de [ActionProvider](http://developer.android.com/reference/android/view/ActionProvider.html) , consulte [ActionBarSettingsActionProviderActivity](http://developer.android.com/resources/samples/ApiDemos/src/com/example/android/apis/app/ActionBarSettingsActionProviderActivity.html) .

**Agregar fichas de navegación**

http://developer.android.com/images/ui/actionbar.png

**Figura 9.** Captura de pantalla de las pestañas de la barra de acciones [Galería de nido de abeja](http://developer.android.com/resources/samples/HoneycombGallery/index.html) aplicación.



**Figura 10.** Captura de pantalla de las pestañas en la barra de acción apilados en una pantalla estrecha.

Cuando se desea proporcionar fichas de navegación en una actividad, uso de las pestañas de la barra de acción es una gran opción (en lugar de utilizar [TabWidget](http://developer.android.com/reference/android/widget/TabWidget.html) ), porque el sistema adapta las fichas de la barra de acciones para la pantalla diferente a los tamaños de colocación de la barra de acción principal, cuando la pantalla es lo suficientemente amplia, o en una barra separada (conocida como la "barra de acción apilada") cuando la pantalla es demasiado estrecha, como se muestra en las figuras 9 y 10.

Para cambiar entre los fragmentos utilizando las fichas, debe realizar una transacción fragmento cada vez que se selecciona una ficha. Si no estás familiarizado con la forma de cambiar fragmentos utilizando [FragmentTransaction](http://developer.android.com/reference/android/app/FragmentTransaction.html) , primero lea la [Fragmentos](http://developer.android.com/guide/topics/fundamentals/fragments.html) guía para el desarrollador.

Para empezar, su diseño debe incluir un [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html) en el que se coloca cada [Fragment](http://developer.android.com/reference/android/app/Fragment.html) asociado con una ficha. Asegúrese de que el [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html) tiene un identificador de recursos para que pueda hacer referencia a ella desde la pestaña de intercambio de código. Por otra parte, si el contenido de la ficha se llenará el diseño de la actividad (con exclusión de la barra de acciones), entonces su actividad no necesita un diseño en absoluto (ni siquiera tiene que llamar [setContentView()](http://developer.android.com/reference/android/app/Activity.html#setContentView%28android.view.View%29) ). En su lugar, puede colocar cada fragmento de la raíz por defecto [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html) , que puede consultar con el android.R.id.content ID (se puede ver este identificador utilizado en el código de ejemplo siguiente, durante las transacciones de fragmentos).

Una vez que determine que los fragmentos aparecen en el diseño, el procedimiento básico para agregar fichas es la siguiente:

1. Implementar el [ActionBar.TabListener](http://developer.android.com/reference/android/app/ActionBar.TabListener.html) interfaz. Devoluciones de llamada de esta interfaz de responder a los eventos del usuario en las fichas para que pueda intercambiar fragmentos.
2. Por cada ficha que desea agregar, una instancia de un [ActionBar.Tab](http://developer.android.com/reference/android/app/ActionBar.Tab.html) y establecer el [ActionBar.TabListener](http://developer.android.com/reference/android/app/ActionBar.TabListener.html) llamando [setTabListener()](http://developer.android.com/reference/android/app/ActionBar.Tab.html#setTabListener%28android.app.ActionBar.TabListener%29) . También se establece el título de la ficha y / o el icono con [setText()](http://developer.android.com/reference/android/app/ActionBar.Tab.html#setText%28int%29) y / o [setIcon()](http://developer.android.com/reference/android/app/ActionBar.Tab.html#setIcon%28android.graphics.drawable.Drawable%29) .
3. Añadir cada pestaña a la barra de acción llamando a [addTab()](http://developer.android.com/reference/android/app/ActionBar.html#addTab%28android.app.ActionBar.Tab%29) .

Al mirar el [ActionBar.TabListener](http://developer.android.com/reference/android/app/ActionBar.TabListener.html) la interfaz, el aviso de que los métodos de devolución de llamada proporcionar sólo la [ActionBar.Tab](http://developer.android.com/reference/android/app/ActionBar.Tab.html) que se ha seleccionado y una [FragmentTransaction](http://developer.android.com/reference/android/app/FragmentTransaction.html) para que usted pueda realizar transacciones-que los fragmentos no dice nada acerca de lo que usted debe cambiar fragmento de dentro o fuera. Por lo tanto, debe definir su propia asociación entre cada [ActionBar.Tab](http://developer.android.com/reference/android/app/ActionBar.Tab.html) y el apropiado [Fragment](http://developer.android.com/reference/android/app/Fragment.html) que representa (a fin de realizar la transacción fragmento apropiado). Hay varias maneras de definir la asociación, en función de su diseño. En el siguiente ejemplo, el [ActionBar.TabListener](http://developer.android.com/reference/android/app/ActionBar.TabListener.html) la aplicación proporciona un constructor de manera que cada nueva pestaña utiliza su propia instancia del oyente. Cada instancia del detector define varios campos que son necesarios para llevar a cabo más adelante una transacción en el fragmento correspondiente.

Por ejemplo, aquí es cómo se puede implementar el [ActionBar.TabListener](http://developer.android.com/reference/android/app/ActionBar.TabListener.html) de tal manera que cada pestaña utiliza su propia instancia del oyente:

<T pública estática TabListener clase extiende Fragment> implementa actionbar.TabListener {

Fragmento mFragment privada;

mActivity privado final;

MTag privada final String;

final privado Clase <T> mCLASS;

/ \*\* Constructor utilizado cada vez que se crea una nueva pestaña.

\* @ Param La actividad de la Actividad de acogida, que se utiliza para crear una instancia del fragmento

\* @ Param etiqueta etiqueta de identificación para el fragmento

\* @ Param ClZ El fragmento de la clase, que se utiliza para crear una instancia del fragmento

\* /

TabListener público (la actividad de la actividad, la etiqueta de cadena, Clase <T> ClZ) {

mActivity = actividad;

MTAG = etiqueta;

mCLASS = ClZ;

}

/ \* Las siguientes son cada uno de los [ActionBar.TabListener](http://developer.android.com/reference/android/app/ActionBar.TabListener.html) devoluciones de llamada \* /

public void onTabSelected (ficha Tab, FragmentTransaction pies) {

/ / Comprobar si el fragmento ya se ha inicializado

if (mFragment == null) {

/ / Si no, crear una instancia y agregarlo a la actividad

mFragment = Fragment.instantiate (mActivity, mClass.getName ());

ft.add (android.R.id.content, mFragment, MTag);

} else {

/ / Si existe, sólo tiene que conectar con el fin de mostrar

ft.attach (mFragment);

}

}

public void onTabUnselected (ficha Tab, FragmentTransaction pies) {

if (mFragment!= Null) {

/ / Desata el fragmento, porque otro se va a adjuntar

ft.detach (mFragment);

}

}

public void onTabReselected (ficha Tab, FragmentTransaction pies) {

/ / El usuario selecciona la pestaña ya seleccionada. Por lo general, no hacer nada.

}

}

**Precaución: No debe** llamar a [commit()](http://developer.android.com/reference/android/app/FragmentTransaction.html#commit%28%29) para la transacción fragmento en cada una de estas devoluciones de llamada, el sistema que te llama y se puede producir una excepción si se llama a sí mismo. También **no se puede** agregar fragmentos de estas transacciones a la pila de la espalda.

En este ejemplo, el oyente se conecta simplemente ( [attach()](http://developer.android.com/reference/android/app/FragmentTransaction.html#attach%28android.app.Fragment%29) ) un fragmento de la actividad de diseño, o si no se crea una instancia, se crea el fragmento y agrega ( [add()](http://developer.android.com/reference/android/app/FragmentTransaction.html#add%28android.app.Fragment,%20java.lang.String%29) ) para el diseño (como un niño de la android.R.id.content grupo de opinión), cuando en la ficha correspondiente se selecciona y separa ( [detach()](http://developer.android.com/reference/android/app/FragmentTransaction.html#detach%28android.app.Fragment%29) ) cuando la ficha no está seleccionada.

El [ActionBar.TabListener](http://developer.android.com/reference/android/app/ActionBar.TabListener.html) aplicación es la mayor parte del trabajo. Todo lo que queda es la creación de cada uno de [ActionBar.Tab](http://developer.android.com/reference/android/app/ActionBar.Tab.html) y agregarlo a la [ActionBar](http://developer.android.com/reference/android/app/ActionBar.html) . Además, debe llamar a [setNavigationMode(NAVIGATION\_MODE\_TABS)](http://developer.android.com/reference/android/app/ActionBar.html#setNavigationMode%28int%29) para hacer las fichas visibles. Usted también puede desactivar el título de la actividad llamando [setDisplayShowTitleEnabled(false)](http://developer.android.com/reference/android/app/ActionBar.html#setDisplayShowTitleEnabled%28boolean%29) si los títulos de las fichas en realidad indican la vista actual.

Por ejemplo, el código siguiente agrega dos pestañas utilizando el oyente se ha definido anteriormente:

@ Override

protected void onCreate (paquete savedInstanceState) {

super.onCreate (savedInstanceState);

/ / Tenga en cuenta que setContentView () no se utiliza, debido a que usamos la raíz

/ / Androide.R.id.content como contenedor para cada fragmento

/ / Configuración de barra de acción para las pestañas

Actionbar actionbar getActionBar = ();

actionBar.setNavigationMode (ActionBar.NAVIGATION\_MODE\_TABS);

actionBar.setDisplayShowTitleEnabled (false);

Tab tab = actionBar.newTab ()

. SetText (R.string.artist)

. SetTabListener (nuevo TabListener <ArtistFragment> (

esto, "artista", ArtistFragment.class));

actionBar.addTab (ficha);

tab = actionBar.newTab ()

. SetText (R.string.album)

. SetTabListener (nuevo TabListener <AlbumFragment> (

esto, "álbum", AlbumFragment.class));

actionBar.addTab (ficha);

}

**Nota:** La implementación de arriba para [ActionBar.TabListener](http://developer.android.com/reference/android/app/ActionBar.TabListener.html) es una de varias técnicas posibles. Puedes ver más de este estilo en la [API Demos](http://developer.android.com/resources/samples/ApiDemos/src/com/example/android/apis/app/FragmentTabs.html) aplicación.

Si su actividad se detiene, debe conservar la ficha actualmente seleccionada con el [estado de la instancia salvo](http://developer.android.com/guide/topics/fundamentals/activities.html#SavingActivityState) por lo que puede abrir la ficha correspondiente cuando el usuario vuelve. Cuando es el momento de guardar el estado, se puede consultar la ficha seleccionada actualmente con [getSelectedNavigationIndex()](http://developer.android.com/reference/android/app/ActionBar.html#getSelectedNavigationIndex%28%29) . Esto devuelve la posición de índice de la pestaña seleccionada.

**Precaución:** Es importante que guarde el estado de cada fragmento según sea necesario, de modo que cuando los usuarios cambian los fragmentos con las pestañas y luego regresar a un fragmento anterior, se ve como lo hizo cuando se fueron. Para información sobre cómo guardar el estado de su fragmento, ver el [Fragmentos](http://developer.android.com/guide/topics/fundamentals/fragments.html) guía para el desarrollador.

**Nota:** En algunos casos, el sistema Android mostrará las pestañas barra de acciones como una lista desplegable con el fin de garantizar el mejor ajuste en la barra de acción.

**Añadir desplegable de navegación**

Como otro modo de navegación (o filtrado) dentro de su actividad, la barra de acción ofrece un sistema incorporado en la lista desplegable. Por ejemplo, la lista desplegable puede ofrecer diferentes modos en que se ordena el contenido de la actividad.

El procedimiento básico para permitir desplegable de navegación es:

1. Crear un [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) que proporciona la lista de elementos seleccionables para la lista desplegable y la disposición a utilizar en la elaboración de cada elemento de la lista.
2. Implementar [ActionBar.OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) para definir el comportamiento que se produce cuando el usuario selecciona un elemento de la lista.
3. Activar el modo de navegación de la barra de acción con [setNavigationMode()](http://developer.android.com/reference/android/app/ActionBar.html#setNavigationMode%28int%29) . Por ejemplo:
4. Actionbar actionbar getActionBar = ();
5. actionBar.setNavigationMode (ActionBar.NAVIGATION\_MODE\_LIST);

**Nota:** Usted debe realizar esta actividad durante el de [onCreate()](http://developer.android.com/reference/android/app/Activity.html#onCreate%28android.os.Bundle%29) método.

1. Establecer la devolución de llamada de la lista desplegable con [setListNavigationCallbacks()](http://developer.android.com/reference/android/app/ActionBar.html#setListNavigationCallbacks%28android.widget.SpinnerAdapter,%20android.app.ActionBar.OnNavigationListener%29) . Por ejemplo:
2. actionBar.setListNavigationCallbacks (mSpinnerAdapter, mNavigationCallback);

Este método tiene sus [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) y [ActionBar.OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) .

Esa es la configuración básica. Sin embargo, la aplicación de las [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) y [ActionBar.OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) es donde la mayor parte del trabajo está hecho. Hay muchas maneras que usted puede implementar estos para definir la funcionalidad de su desplegable de navegación y ejecución de los distintos tipos de [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) está fuera del alcance de este documento (que debe referirse a la [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) referencia de clase para más información). Sin embargo, a continuación es un ejemplo simple de un [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) y [ActionBar.OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) para empezar (haga clic en el título para revelar la muestra).

[**http://developer.android.com/assets/images/triangle-closed.pngSpinnerAdapter ejemplo y OnNavigationListener**](http://developer.android.com/guide/topics/ui/actionbar.html)

[SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) es un adaptador que proporciona datos para un control giratorio, como la lista desplegable en la barra de acción. [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) es una interfaz que se puede implementar, pero Android incluye algunas implementaciones útiles que se pueden ampliar, como [ArrayAdapter](http://developer.android.com/reference/android/widget/ArrayAdapter.html) y [SimpleCursorAdapter](http://developer.android.com/reference/android/widget/SimpleCursorAdapter.html) . Por ejemplo, aquí está una manera fácil de crear un [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) mediante el uso de [ArrayAdapter](http://developer.android.com/reference/android/widget/ArrayAdapter.html) aplicación, que utiliza una matriz de cadenas como la fuente de datos:

SpinnerAdapter mSpinnerAdapter = ArrayAdapter.createFromResource (esto, R.array.action\_list,

androide.R.layout.simple\_spinner\_dropdown\_item);

El [createFromResource()](http://developer.android.com/reference/android/widget/ArrayAdapter.html#createFromResource%28android.content.Context,%20int,%20int%29) método toma tres parámetros: la aplicación [Context](http://developer.android.com/reference/android/content/Context.html) , la identificación de recursos para la matriz de cadenas, y la disposición a utilizar para cada elemento de la lista.

Una [matriz de cadena](http://developer.android.com/guide/topics/resources/string-resource.html#StringArray) se define en un recurso se parece a esto:

<?xml version = "1.0" encoding = "UTF-8"?>

<resources>

<string-array name="action\_list">

<item> Mercurio </ item>

<item> Venus </ item>

<item> la Tierra </ item>

</ String-> array

</ Pre>

Los [ArrayAdapter](http://developer.android.com/reference/android/widget/ArrayAdapter.html) devueltos por [createFromResource()](http://developer.android.com/reference/android/widget/ArrayAdapter.html#createFromResource%28android.content.Context,%20int,%20int%29) está completo y listo para que usted pueda pasar a [setListNavigationCallbacks()](http://developer.android.com/reference/android/app/ActionBar.html#setListNavigationCallbacks%28android.widget.SpinnerAdapter,%20android.app.ActionBar.OnNavigationListener%29) (en el paso 4 de arriba). Antes de hacerlo, sin embargo, es necesario crear el [OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) .

La implementación de [ActionBar.OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) es donde se manejan los cambios de fragmentos u otras modificaciones a su actividad cuando el usuario selecciona un elemento de la lista desplegable. Sólo hay un método de devolución de llamada para poner en práctica en el oyente: [onNavigationItemSelected()](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html#onNavigationItemSelected%28int,%20long%29) .

El [onNavigationItemSelected()](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html#onNavigationItemSelected%28int,%20long%29) método recibe la posición del elemento de la lista y un identificador de elemento único, proporcionado por los [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) .

He aquí un ejemplo que crea instancias de una aplicación anónima de [OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) , que inserta un [Fragment](http://developer.android.com/reference/android/app/Fragment.html) en el contenedor de diseño identifica por R.id.fragment\_container :

mOnNavigationListener = new OnNavigationListener () {

/ / Obtener las cuerdas que se dan para ArrayAdapter el desplegable de

String [] cadenas = getResources () getStringArray (R.array.action\_list).;

@ Override

public boolean onNavigationItemSelected (posición int, long itemId) {

/ / Crear nuevo fragmento de nuestra clase propia de fragmentos

ListContentFragment newFragment ListContentFragment = new ();

FragmentTransaction pies = openFragmentTransaction ();

/ / Cambiar lo que está en el contenedor fragmento con este fragmento

/ / Y dar el fragmento de un nombre de etiqueta igual a la cadena en la posición seleccionada

ft.replace (R.id.fragment\_container, newFragment, las cadenas [posición]);

/ / Aplicar los cambios

ft.commit ();

return true;

}

};

Esta instancia de [OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) es completo y ahora se puede llamar a [setListNavigationCallbacks()](http://developer.android.com/reference/android/app/ActionBar.html#setListNavigationCallbacks%28android.widget.SpinnerAdapter,%20android.app.ActionBar.OnNavigationListener%29) (en el paso 4), pasando por los [ArrayAdapter](http://developer.android.com/reference/android/widget/ArrayAdapter.html) y esto [OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) .

En este ejemplo, cuando el usuario selecciona un elemento de la lista desplegable, un fragmento se añade a la disposición (en sustitución del fragmento actual en el R.id.fragment\_container vista). El fragmento añadido se da una etiqueta que lo identifica singularmente, que es la misma cadena utilizado para identificar el fragmento en la lista desplegable.

He aquí un vistazo a la ListContentFragment clase que define cada fragmento en este ejemplo:

ListContentFragment public class {Fragmento

private String mText;

@ Override

public void onAttach (actividad Actividad) {

/ / Esta es la devolución de llamada recibió por primera vez, aquí se puede establecer el texto de

/ / Fragmento la definida por la etiqueta especificada durante la transacción fragmento

super.onAttach (actividad);

mText getTag = ();

}

@ Override

pública onCreateView Ver (LayoutInflater inflater, contenedor ViewGroup,

Paquete savedInstanceState) {

/ / Esto se le llama para definir el diseño del fragmento;

/ / Que acabamos de crear una TextView y establezca su texto a ser la etiqueta de fragmento

TextView texto = new TextView (getActivity ());

text.setText (mText);

devolver el texto;

}

}

**El estilo de la barra de acciones**

Si usted ha implementado un diseño personalizado para los widgets de la aplicación, también puede que quiera volver a diseñar algunas de la barra de acción para que coincida con el diseño de su aplicación. Para ello, es necesario utilizar Android en [el estilo y el tema de](http://developer.android.com/guide/topics/ui/themes.html) marco para cambiar el estilo de la barra de acción utilizando las propiedades especiales de estilo.

**Nota:** Para que las imágenes de fondo para cambiar la apariencia en función del estado del botón actual (seleccionado, presiona, no seleccionado), el recurso dibujable que utilice debe ser una [lista del estado dibujable](http://developer.android.com/guide/topics/resources/drawable-resource.html#StateList) .

**Precaución:** Para todos los dibujables de fondo que usted proporciona, asegúrese de usar [parches de nueve dibujables](http://developer.android.com/guide/topics/graphics/2d-graphics.html#nine-patch) para permitir el estiramiento. La imagen de nueve parche debe *ser* inferior a 40px 30px de alto y de ancho (para el activo MDPI).

**Aspecto general**

[android:windowActionBarOverlay](http://developer.android.com/reference/android/R.attr.html#windowActionBarOverlay)

Declara si la barra de acción debe superponer el diseño de la actividad en lugar de compensar la posición de la actividad de distribución (por ejemplo, la aplicación Gallery utiliza el modo de superposición). Esto es false por defecto.

Normalmente, la barra de acción requiere su propio espacio en la pantalla y su diseño de la actividad se llena de lo que queda. Cuando la barra de acción está en el modo de superposición, el diseño de la actividad utiliza todo el espacio disponible y el sistema de la barra de acción se basa en la parte superior. El modo de superposición puede ser útil si usted quiere que su contenido para mantener un tamaño fijo y la posición de la barra de acciones, cuando se oculta y se muestra. También te recomendamos usarlo simplemente como un efecto visual, porque se puede utilizar un fondo semi-transparente de la barra de acción para que el usuario aún se pueden ver parte de su trazado actividad detrás de la barra de acción.

**Nota:** Los [Holo](http://developer.android.com/reference/android/R.style.html#Theme_Holo) familias temáticos señalar a la barra de acción con un fondo semi-transparente de forma predeterminada. Sin embargo, usted puede modificar con sus propios estilos y la [DeviceDefault](http://developer.android.com/reference/android/R.style.html#Theme_DeviceDefault) el tema en diferentes dispositivos podría utilizar un fondo opaco de forma predeterminada.

Cuando el modo de superposición está activada, el diseño de la actividad no tiene conciencia de la barra de acción por la que se en la parte superior de la misma. Por lo tanto, hay que tener cuidado de no colocar toda la información importante o de los componentes de interfaz de usuario en el área superpuesta por la barra de acción. Si es apropiado, se puede hacer referencia al valor de la plataforma de [actionBarSize](http://developer.android.com/reference/android/R.attr.html#actionBarSize) para determinar la altura de la barra de acción, haciendo referencia a que en su diseño XML. Por ejemplo:

<SomeView

...

android: layout\_marginTop = "?android: attr / actionBarSize "/>

También puede recuperar la altura de la barra de acción en tiempo de ejecución con [getHeight()](http://developer.android.com/reference/android/app/ActionBar.html#getHeight%28%29) . Esto refleja la altura de la barra de acción en el momento que se llama, que no puede incluir la barra de acción apilados (debido a las pestañas de navegación) si se llama durante los primeros métodos de la actividad del ciclo de vida. Para ver cómo se puede determinar la altura total en tiempo de ejecución, incluyendo la barra de acción apiladas, ver el [TitlesFragment](http://developer.android.com/resources/samples/HoneycombGallery/src/com/example/android/hcgallery/TitlesFragment.html) clase en la [Galería de nido de abeja](http://developer.android.com/resources/samples/HoneycombGallery/index.html) aplicación de ejemplo.

**Los temas de acción**

[android:actionButtonStyle](http://developer.android.com/reference/android/R.attr.html#actionButtonStyle)

Define un recurso de estilo para los botones de los elementos de acción.

[android:actionBarItemBackground](http://developer.android.com/reference/android/R.attr.html#actionBarItemBackground)

Define un recurso dibujable para el fondo de cada elemento de acción. (Añadido en el nivel de API 14.)

[android:itemBackground](http://developer.android.com/reference/android/R.attr.html#itemBackground)

Define un recurso dibujable para el fondo de cada elemento del menú de desbordamiento.

[android:actionBarDivider](http://developer.android.com/reference/android/R.attr.html#actionBarDivider)

Define un recurso dibujable de la división entre los elementos de acción. (Añadido en el nivel de API 14.)

[android:actionMenuTextColor](http://developer.android.com/reference/android/R.attr.html#actionMenuTextColor)

Define un color para el texto que aparece en un elemento de acción.

[android:actionMenuTextAppearance](http://developer.android.com/reference/android/R.attr.html#actionMenuTextAppearance)

Define un recurso de estilo para el texto que aparece en un elemento de acción.

[android:actionBarWidgetTheme](http://developer.android.com/reference/android/R.attr.html#actionBarWidgetTheme)

Define un recurso temático para los widgets que se inflan en la barra de acción como [puntos de vista de acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionView) . (Añadido en el nivel de API 14.)

**Fichas de navegación**

[android:actionBarTabStyle](http://developer.android.com/reference/android/R.attr.html#actionBarTabStyle)

Define un recurso de estilo para las pestañas en la barra de acción.

[android:actionBarTabBarStyle](http://developer.android.com/reference/android/R.attr.html#actionBarTabBarStyle)

Define un recurso de estilo de la barra delgada que aparece debajo de las pestañas de navegación.

[android:actionBarTabTextStyle](http://developer.android.com/reference/android/R.attr.html#actionBarTabTextStyle)

Define un recurso de estilo para el texto en las pestañas de navegación.

**Listas desplegables**

[android:actionDropDownStyle](http://developer.android.com/reference/android/R.attr.html#actionDropDownStyle)

Define un estilo para la navegación desplegable (por ejemplo, los antecedentes y estilos de texto).

Por ejemplo, aquí hay un archivo que define algunos estilos personalizados para la barra de acción:

<?xml version = "1.0" encoding = "UTF-8"?>

<resources>

<-! El tema aplicado a la aplicación o actividad ->

<Nombre del estilo = "CustomActivityTheme" padre = "@ android: estilo / tema.Holo ">

  <item name="android:actionBarTabTextStyle"> @ estilo / CustomTabTextStyle </ item>

  <item name="android:actionBarDivider"> @ dibujable / ab\_divider </ item>

<item name="android:actionBarItemBackground"> ​​@ dibujable / ab\_item\_background </ item>

</ Style>

<-! Estilo para el texto de la acción ficha de la barra ->

<Nombre del estilo = "CustomTabTextStyle" padre = "@ android: estilo / TextAppearance.Holo ">

<item name="android:textColor"> # 2456c2 </ item>

</style>

</ Recursos>

**Nota: Asegúrese de** que su tema se declara un tema principal en el <style> etiqueta, de la cual hereda todos los estilos no explícitamente declaradas por el tema. Al modificar la barra de acciones, usando un tema importante para los padres es que usted puede simplemente reemplazan los estilos que la barra de acciones que desea cambiar sin volver a aplicar los estilos que desea dejar en paz (por ejemplo, el aspecto del texto o material de relleno en los puntos de acción).

Usted puede aplicar el tema personalizado a toda la aplicación o para las actividades individuales en el archivo de manifiesto de esta manera:

<Aplicación para Android: tema = "@ estilo / CustomActivityTheme" ... />

Para obtener más información sobre el uso de recursos de estilo y el tema de la aplicación, lea [estilos y temas](http://developer.android.com/guide/topics/ui/themes.html) .

**Diseño avanzado**

Si usted necesita un estilo más avanzado de la barra de acción que está disponible con las propiedades anteriores, se puede incluir [android:actionBarStyle](http://developer.android.com/reference/android/R.attr.html#actionBarStyle) y [android:actionBarSplitStyle](http://developer.android.com/reference/android/R.attr.html#actionBarSplitStyle) en el tema de su actividad. Cada una de ellas especifica otro estilo que se pueden definir varias propiedades de la barra de acciones, incluyendo diferentes orígenes con [android:background](http://developer.android.com/reference/android/R.attr.html#background) , [android:backgroundSplit](http://developer.android.com/reference/android/R.attr.html#backgroundSplit) y [android:backgroundStacked](http://developer.android.com/reference/android/R.attr.html#backgroundStacked) . Si reemplaza los estilos de la barra de acciones, asegúrese de que se defina una acción de los padres bar de estilo, tales como [Widget.Holo.ActionBar](http://developer.android.com/reference/android/R.style.html#Widget_Holo_ActionBar) .

Por ejemplo, si usted quiere cambiar el fondo de la barra de acciones, puede utilizar los siguientes estilos:

<?xml version = "1.0" encoding = "UTF-8"?>

<resources>

<-! El tema aplicado a la aplicación o actividad ->

<Nombre del estilo = "CustomActivityTheme" padre = "@ android: estilo / tema.Holo ">

<item name="android:actionBarStyle"> @ estilo / MyActionBar </ item>

<-! Otra actividad y los estilos de barra de acción aquí ->

</ Style>

<-! Estilo de los fondos de la acción de barras ->

<Nombre del estilo = "MyActionBar" padre = "@ android: estilo / Widget.Holo.Actionbar ">

<item name="android:background"> ​​@ dibujable / ab\_background </ item>

<item name="android:backgroundStacked"> @ dibujable / ab\_background </ item>

<item name="android:backgroundSplit"> @ dibujable / ab\_split\_background </ item>

</style>

</ Recursos>

**Barra de acción**

**Vista rápida**

* Una barra de título que incluye el icono de la aplicación y el título de la actividad
* Proporciona acceso a los elementos de menú y los modos de navegación, tales como fichas
* Requiere nivel 11 o mayor de la API

**En este documento**

1. [Adición de la Barra de acciones](http://developer.android.com/guide/topics/ui/actionbar.html#Adding)
   1. [Extracción de la barra de acción](http://developer.android.com/guide/topics/ui/actionbar.html#Removing)
2. [Agregar elementos de acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionItems)
   1. [La elección de sus puntos de acción](http://developer.android.com/guide/topics/ui/actionbar.html#ChoosingActionItems)
   2. [Usar barra de división de acción](http://developer.android.com/guide/topics/ui/actionbar.html#SplitBar)
3. [Utilizando el icono de la aplicación para la navegación](http://developer.android.com/guide/topics/ui/actionbar.html#Home)
   1. [Navegando por](http://developer.android.com/guide/topics/ui/actionbar.html#Up)
4. [Adición de una vista de acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionView)
   1. [Manejo de puntos de vista plegables de acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionViewCollapsing)
5. [Adición de un proveedor de Acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionProvider)
   1. [Usando el ShareActionProvider](http://developer.android.com/guide/topics/ui/actionbar.html#ShareActionProvider)
   2. [Creación de un proveedor de acción personalizada](http://developer.android.com/guide/topics/ui/actionbar.html#CreatingActionProvider)
6. [Agregar fichas de navegación](http://developer.android.com/guide/topics/ui/actionbar.html#Tabs)
7. [Añadir desplegable de navegación](http://developer.android.com/guide/topics/ui/actionbar.html#Dropdown)
8. [El estilo de la barra de acciones](http://developer.android.com/guide/topics/ui/actionbar.html#Style)
   1. [Aspecto general](http://developer.android.com/guide/topics/ui/actionbar.html#GeneralStyles)
   2. [Los temas de acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionItemStyles)
   3. [Fichas de navegación](http://developer.android.com/guide/topics/ui/actionbar.html#NavigationStyles)
   4. [Lista desplegable](http://developer.android.com/guide/topics/ui/actionbar.html#DropDownStyles)
   5. [Diseño avanzado](http://developer.android.com/guide/topics/ui/actionbar.html#AdvancedStyles)

**Clases de clave**

1. [ActionBar](http://developer.android.com/reference/android/app/ActionBar.html)
2. [Menu](http://developer.android.com/reference/android/view/Menu.html)
3. [ActionProvider](http://developer.android.com/reference/android/view/ActionProvider.html)

**Muestras relacionadas**

1. [Galería de nido de abeja](http://developer.android.com/resources/samples/HoneycombGallery/index.html)
2. [Acción de compatibilidad de Abogados](http://developer.android.com/resources/samples/ActionBarCompat/index.html)
3. [Demos de la API](http://developer.android.com/resources/samples/ApiDemos/src/com/example/android/apis/app/index.html#ActionBar)

**Véase también**

1. [Diseño de Android: Barra de acción](http://developer.android.com/design/patterns/actionbar.html)
2. [Menús](http://developer.android.com/guide/topics/ui/menus.html)
3. [Tabletas de apoyo y Móviles](http://developer.android.com/guide/practices/tablets-and-handsets.html)

La barra de acción es una característica de la ventana que identifica la aplicación y la ubicación del usuario, y proporciona las acciones del usuario y los modos de navegación. Se debe utilizar la barra de acción en la mayoría de las actividades que deben presentar de forma destacada las acciones del usuario o la navegación mundial, debido a que la barra de acción ofrece a los usuarios una interfaz coherente en todas las aplicaciones y el sistema se adapta con elegancia la apariencia de la barra de acción para configuraciones de pantalla diferentes. Usted puede controlar los comportamientos y la visibilidad de la barra de acción con el [ActionBar](http://developer.android.com/reference/android/app/ActionBar.html) API, que se agregaron en Android 3.0 (nivel de la API 11).

Los objetivos principales de la barra de acción son los siguientes:

* Proporcionar un espacio dedicado a la identificación de la marca de la aplicación y la ubicación del usuario.

Esto se logra con el icono de la aplicación o logotipo en el lado izquierdo y el título de la actividad. Usted puede optar por retirar el título de la actividad, sin embargo, si la vista actual se identifica por una etiqueta de navegación, tales como la pestaña seleccionada.

* Proporcionar una navegación consistente y ver el refinamiento través de diferentes aplicaciones.

La barra de acción proporciona un sistema integrado de navegación ficha para cambiar entre los [fragmentos](http://developer.android.com/guide/topics/fundamentals/fragments.html) . También ofrece una lista desplegable se puede utilizar como una alternativa modo de navegación o para afinar la vista actual (por ejemplo, para ordenar una lista por diferentes criterios).

* Realiza acciones clave para la actividad (por ejemplo, "búsqueda", "cree", "compartir", etc) destacados y accesibles para el usuario de una manera predecible.

Puede proporcionar acceso instantáneo a las acciones del usuario mediante la colocación de los elementos clave de la [opciones de menú](http://developer.android.com/guide/topics/ui/menus.html#OptionsMenu) directamente en la barra de acción, como "puntos de acción." Los temas de acción también puede proporcionar una "visión de acción", que proporciona un widget integrado por conductas de acción más inmediatos. Los elementos de menú que no sean promovidos a un elemento de acción están disponibles en el menú de desbordamiento, reveló, ya sea el botón de *menú del* dispositivo (si está disponible) o por un "menú de desbordamiento" en la barra de acción (cuando el dispositivo no incluye un botón de *Menú* ).

http://developer.android.com/images/ui/actionbar.png

**Figura 1.** Barra de acciones de la [Galería de nido de abeja](http://developer.android.com/resources/samples/HoneycombGallery/index.html) aplicación (en el auricular del paisaje), que muestra el logo a la izquierda, pestañas de navegación, y un elemento de acción de la derecha (más el botón de menú de desbordamiento).

**Nota:** Si usted está buscando para obtener información sobre la barra de acciones contextuales para la visualización de los elementos contextuales de acción, ver el [menú](http://developer.android.com/guide/topics/ui/menus.html#context-menu) guía.

**Acción de Diseño de Abogados**

Para conocer las directrices de diseño, leer Diseño de Android [Barra de acción del](http://developer.android.com/design/patterns/actionbar.html) guía.

**Restante compatible con versiones anteriores**

Si desea proporcionar una barra de acción en su solicitud *y* siguen siendo compatibles con las versiones de Android mayor que 3,0, lo que necesita para crear la barra de acción en el diseño de su actividad (debido a que el [ActionBar](http://developer.android.com/reference/android/app/ActionBar.html) la clase no está disponible en las versiones anteriores).

Para ayudarle, la [barra de acciones de compatibilidad](http://developer.android.com/resources/samples/ActionBarCompat/index.html) aplicación de ejemplo proporciona un nivel de API y el diseño de la barra de acción que permite a su aplicación para utilizar algunos de los [ActionBar](http://developer.android.com/reference/android/app/ActionBar.html) API y también apoyar a las versiones anteriores de Android mediante la sustitución de la barra de título tradicional con un diseño de barra de acción personalizado.

**Adición de la Barra de acciones**

Empezando con Android 3.0 (nivel de la API 11), la barra de acción está incluido en todas las actividades que utilizan el [Theme.Holo](http://developer.android.com/reference/android/R.style.html#Theme_Holo) tema (o uno de sus descendientes), que es el tema por defecto, cuando ya sea el [targetSdkVersion](http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#target) o [minSdkVersion](http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#min) atributo se establece en "11" o mayor. Por ejemplo:

<Manifiesta ... >

<Usos-SDK de Android: minSdkVersion = "4"

**android: targetSdkVersion = "11"** />

...

</ Manifest>

En este ejemplo, la aplicación requiere una versión mínima del nivel de la API 4 (Android 1.6), pero también apunta a nivel de la API 11 (Android 3.0). De esta manera, cuando la aplicación se ejecuta en Android 3.0 o superior, el sistema aplica el tema holográfico para cada actividad, y por lo tanto, cada actividad incluye la barra de acción.

Si desea utilizar [ActionBar](http://developer.android.com/reference/android/app/ActionBar.html) API, como para añadir los modos de navegación y modificar estilos de barra de acción, debe establecer la [minSdkVersion](http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#min) al "11" o mayor. Si desea que su aplicación para apoyar a las versiones anteriores de Android, hay maneras de utilizar un conjunto limitado de [ActionBar](http://developer.android.com/reference/android/app/ActionBar.html) API en dispositivos compatibles con el nivel API de 11 o superior, mientras que aún se ejecuta en las versiones anteriores. Consulte la información acerca de sidebox restante compatible con versiones anteriores.

**Extracción de la barra de acción**

Si usted no desea que la barra de acción para una actividad particular, definir el tema de la actividad de [Theme.Holo.NoActionBar](http://developer.android.com/reference/android/R.style.html#Theme_Holo_NoActionBar) . Por ejemplo:

<Actividad de Android: tema = "@ android: estilo / tema.Holo.NoActionBar ">

También puede ocultar la barra de acciones en tiempo de ejecución llamando a [hide()](http://developer.android.com/reference/android/app/ActionBar.html#hide%28%29) . Por ejemplo:

Actionbar actionbar = [getActionBar()](http://developer.android.com/reference/android/app/Activity.html#getActionBar%28%29) ;

actionBar.hide ();

Cuando se oculta la barra de acciones, el sistema se ajusta a su diseño de actividad para llenar todo el espacio de pantalla disponible. Usted puede traer a la barra de acción vuelve con [show()](http://developer.android.com/reference/android/app/ActionBar.html#show%28%29) .

Tenga en cuenta que ocultar y quitar la barra de acción hace que su actividad de re-diseño con el fin de explicar el espacio consumido por la barra de acción. Si su actividad regularmente se esconde y muestra la barra de acción (como en la aplicación Android Gallery), es posible que desee utilizar el modo de superposición. Superposición de modo señala a la barra de acción en la parte superior de su diseño de la actividad en lugar de en su propia área de la pantalla. De esta manera, el diseño permanece fija cuando se oculta la barra de acciones y re-aparece. Para activar el modo de superposición, crear un tema para su actividad y establecer [android:windowActionBarOverlay](http://developer.android.com/reference/android/R.attr.html#windowActionBarOverlay) de true . Para obtener más información, consulte la sección acerca de [estilos a la barra de acción](http://developer.android.com/guide/topics/ui/actionbar.html#Style) .

**Consejo:** Si tienes un tema personalizado en el que la actividad que le gustaría quitar la barra de acción, establecer el [android:windowActionBar](http://developer.android.com/reference/android/R.styleable.html#Theme_windowActionBar) propiedad de estilo de false . Sin embargo, si se quita la barra de acción con un tema, a continuación, la ventana no permitirá que la barra de acción en absoluto, por lo que no puede añadirlo más tarde-llamado [getActionBar()](http://developer.android.com/reference/android/app/Activity.html#getActionBar%28%29) devolverá un valor nulo.

**Agregar elementos de acción**

A veces es posible que desee dar a los usuarios acceso inmediato a un elemento de la [menú de opciones](http://developer.android.com/guide/topics/ui/menus.html#OptionsMenu) . Para ello, se puede declarar que el elemento de menú aparecerá en la barra de acción como un "elemento de acción." Un elemento de acción puede incluir un icono y / o un título de texto. Si un elemento de menú no aparece como un elemento de acción, entonces el sistema lo coloca en el menú de desbordamiento. El menú de desbordamiento se revela, ya sea por el botón de *menú del* dispositivo (si es proporcionado por el dispositivo) o un botón adicional en la barra de acción (si el dispositivo no ofrece el botón de *menú).*

http://developer.android.com/images/ui/actionbar-item-withtext.png

**Figura 2.** Dos elementos de acción con títulos de los iconos y el texto y el botón de menú de desbordamiento.

Cuando la primera actividad se inicia, el sistema rellena la barra de acción y el menú de desbordamiento llamando [onCreateOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#onCreateOptionsMenu%28android.view.Menu%29) para su actividad. Como se discutió en el [Menús](http://developer.android.com/guide/topics/ui/menus.html) guía para el desarrollador, es en este método de devolución de llamada que se debe inflar un XML [recurso de menú](http://developer.android.com/guide/topics/resources/menu-resource.html) que define los elementos del menú. Por ejemplo:

@ Override

onCreateOptionsMenu pública booleano (menú Menú) {

MenuInflater inflater getMenuInflater = ();

inflater.inflate (R.menu.main\_activity, menú);

return true;

}

En el archivo XML, se puede solicitar un elemento de menú para que aparezca como un elemento de acción, declarando android:showAsAction="ifRoom" para el <item> elemento. De esta manera, el elemento de menú aparece en la barra de acción para un acceso rápido sólo *si hay espacio* disponible. Si no hay suficiente espacio, el elemento aparece en el menú de desbordamiento.

Si el elemento de menú proporciona un título y un icono-con el android:title y android:icon los atributos, entonces el elemento de acción muestra sólo el icono por defecto. Si desea mostrar el título del texto, añadir "withText" a la android:showAsAction atributo. Por ejemplo:

<?xml version = "1.0" encoding = "UTF-8"?>

<Menú xmlns:android="http://schemas.android.com/apk/res/android">

<Tema android: id = "+ id @ / menu\_save"

android: icon = "@ dibujable / ic\_menu\_save"

android: title = "@ cadena / menu\_save"

**android: showAsAction = "ifRoom | withText"** />

</ Menu>

**Nota:** El "withText" valor es una *pista* a la barra de acción que el título del texto debe aparecer. La barra de acción mostrará el título cuando sea posible, pero tal vez no, si está disponible un icono y la barra de acción está limitada por el espacio.

Cuando el usuario selecciona un elemento de acción, su actividad recibe una llamada a [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) , pasando por la identificación proporcionada por el android:id de atributos de la misma devolución de llamada recibida para todos los elementos en el menú de opciones.

Es importante que usted defina siempre android:title de cada elemento de menú, incluso si usted no declara que el título aparecerá con la acción del elemento-por tres razones:

* Si no hay suficiente espacio en la barra de acción para el elemento de acción, el elemento de menú aparece en el menú de desbordamiento y sólo aparece el título.
* Los lectores de pantalla para usuarios con deficiencias visuales leer el título del elemento de menú.
* Si el elemento de acción aparece con sólo el icono, el usuario puede pulsar el largo tema para revelar una información sobre herramientas que muestra el título del artículo de acción.

El android:icon es siempre opcional, pero recomendable. Por recomendaciones de diseño de iconos, ver las [de acción icono de la barra](http://developer.android.com/guide/practices/ui_guidelines/icon_design_action_bar.html) pautas de diseño.

**Nota:** Si ha agregado el elemento de menú de un fragmento, a través del [Fragment](http://developer.android.com/reference/android/app/Fragment.html) la clase de [onCreateOptionsMenu](http://developer.android.com/reference/android/app/Fragment.html#onCreateOptionsMenu%28android.view.Menu,%20android.view.MenuInflater%29) de devolución de llamada, el sistema llama a la respectiva [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Fragment.html#onOptionsItemSelected%28android.view.MenuItem%29) método para que el fragmento cuando el usuario selecciona uno de los elementos del fragmento. Sin embargo, la actividad tiene la oportunidad de controlar el evento en primer lugar, por lo que el sistema llama [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) sobre la actividad antes de llamar a la devolución de llamada para el mismo fragmento.

También puede declarar un elemento para *"siempre"* aparecen como un elemento de acción, en lugar de colocarse en el menú de desbordamiento cuando el espacio es limitado. En la mayoría de los casos, **no** se **debe** forzar a un elemento que aparezca en la barra de acción mediante el "always" valor. Sin embargo, es posible que tenga un elemento para que aparezca siempre cuando se ofrece una [visión de acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionView) que no ofrece una acción predeterminada para el menú de desbordamiento. Tenga en cuenta que los elementos de acción demasiados puede crear una interfaz de usuario desordenado y causar problemas de diseño en los dispositivos con una pantalla estrecha. Lo mejor es usar en su lugar "ifRoom" para solicitar que un artículo aparezca en la barra de acción, pero permitir que el sistema se mueve en el menú de desbordamiento cuando no hay suficiente espacio.

Para obtener más información sobre cómo crear el menú de opciones que define sus elementos de acción, consulte la [Menús](http://developer.android.com/guide/topics/ui/menus.html#options-menu) guía para el desarrollador.

**La elección de sus puntos de acción**

**Los elementos del menú de aplicaciones frente a los controles de otros**

Como regla general, todos los elementos del [menú de opciones](http://developer.android.com/guide/topics/ui/menus.html#OptionsMenu) (por no hablar de los puntos de acción) deben tener un impacto global en la aplicación, en lugar de afectar a sólo una pequeña parte de la interfaz. Por ejemplo, si usted tiene un diseño de panel multi-y un panel se muestra un vídeo, mientras que otra muestra todos los vídeos, los controles del reproductor de vídeo debe aparecer en el panel que contiene el video (no en la barra de acciones), mientras que la barra de acción podría proporcionar una acción artículos para compartir el video o guardar el vídeo a una lista de favoritos.

Así, incluso antes de decidir si un elemento de menú debe aparecer como un elemento de acción, asegúrese de que el tema tiene un alcance global para la actividad actual. Si no es así, entonces usted debe colocarlo como un botón en el contexto apropiado de la distribución de la actividad.

Usted debe elegir con cuidado qué elementos de su menú de opciones debe aparecer como puntos de acción mediante la evaluación de algunos rasgos fundamentales. En general, cada elemento de acción debe ser *por lo menos uno* de los siguientes:

1. **Utilizado con frecuencia:** Es una acción que los usuarios necesitan siete de cada diez visitas o que lo utilizan varias veces en una fila.

Ejemplos frecuentes acciones: "Nuevo mensaje" en la aplicación de mensajería y "Buscar" en la reproducción de Google.

1. **Importante:** Es una acción que es necesario que los usuarios descubran fácilmente o, si no se utiliza con frecuencia, es importante que sea fácil de realizar en los pocos casos en que los usuarios lo necesitan.

Ejemplo acciones importantes: "Agregar red" en la configuración Wi-Fi y "Cambiar a cámara" en la aplicación Galería.

1. **Típico:** Es una acción que se proporciona normalmente en la barra de acción en aplicaciones similares, por lo que sus usuarios esperan encontrar en el tuyo.

Ejemplos típicos de las acciones: "Actualizar" en una aplicación de correo electrónico o social, y "Nuevo contacto" en la aplicación de la gente.

Si usted cree que más de cuatro de sus elementos de menú se puede justificar como elementos de acción, entonces usted debe considerar cuidadosamente su nivel de importancia relativa y tratar de fijar no más de cuatro como elementos de acción (y lo hacen con el "ifRoom" valor a permitir que el sistema de poner un poco de nuevo en el menú de desbordamiento cuando el espacio es limitado en pantallas más pequeñas). Incluso si hay espacio disponible en una amplia pantalla, no se debe crear una larga secuencia de puntos de acción que el desorden de la interfaz de usuario y aparecen como una barra de herramientas de escritorio, a fin de mantener el número de elementos de acción a un mínimo.

Además, las siguientes acciones nunca deben aparecer como puntos de acción: Ajustes, Ayuda, sugerencias o similares. Siempre que permanezcan en el menú de desbordamiento.

**Nota:** Recuerde que no todos los dispositivos cuentan con un botón de hardware dedicado para la búsqueda, por lo que si es una característica importante en su aplicación, siempre debe aparecer como un elemento de acción (y por lo general como primer tema, sobre todo si se ofrecen con una [vista de la acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionView) ).

**Usar barra de división de acción**

Cuando se ejecuta la aplicación en Android 4.0 (nivel de la API 14) y superior, hay un modo extra disponibles para la barra de acción llamada "barra de acción de división". Cuando se habilita barra de división acción, un bar separado en la parte inferior de la pantalla para mostrar todos los elementos de acción cuando la actividad se está ejecutando en una pantalla estrecha (por ejemplo, un teléfono con orientación vertical). La división de la barra de acción para separar los elementos de acción asegura que una cantidad razonable de espacio disponible para mostrar todos los elementos de acción en una pantalla estrecha, dejando espacio para la navegación y elementos de título en la parte superior.

Para activar la barra de acción de división, sólo tiene que añadir uiOptions="splitActionBarWhenNarrow" a sus [<activity>](http://developer.android.com/guide/topics/manifest/activity-element.html) o [<application>](http://developer.android.com/guide/topics/manifest/application-element.html) elemento manifiesto.

Tenga en cuenta que Android ajusta la apariencia de la barra de acción en una variedad de formas, basado en el tamaño de la pantalla actual. Usar barra de división acción es sólo una opción que se puede habilitar para permitir que la barra de acción para optimizar aún más la experiencia de usuario para diferentes tamaños de pantalla. Al hacerlo, usted también puede permitir que la barra de acción para cerrar las pestañas de navegación en la barra de la acción principal. Es decir, si utiliza [pestañas de navegación](http://developer.android.com/guide/topics/ui/actionbar.html#Tabs) en tu barra de acción, una vez que los puntos de acción se separan en una pantalla estrecha, las fichas de navegación puede ser capaz de encajar en la barra de acción principal en lugar de separarse en la "barra de acción apilados". En concreto, si ha desactivado el icono de la barra de acciones y el título (con [setDisplayShowHomeEnabled(false)](http://developer.android.com/reference/android/app/ActionBar.html#setDisplayShowHomeEnabled%28boolean%29) y [setDisplayShowTitleEnabled(false)](http://developer.android.com/reference/android/app/ActionBar.html#setDisplayShowTitleEnabled%28boolean%29) ), luego del colapso de navegación pestañas en la barra de acción principal, como se muestra en el segundo dispositivo en la figura 3.



**Figura 3.** Las maquetas de barra de acción dividida con pestañas de navegación a la izquierda; con el icono de la aplicación y el título de discapacitados a la derecha.

**Nota:** Aunque el [android:uiOptions](http://developer.android.com/reference/android/R.attr.html#uiOptions) atributo se añadió en Android 4.0 (nivel de la API 14), que con seguridad puede incluir en su solicitud, incluso si su [minSdkVersion](http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#min) se establece en un valor inferior a "14" siga siendo compatible con versiones anteriores de Android. Cuando se ejecuta en las versiones anteriores, el sistema simplemente ignora el atributo XML, ya que no lo entiendo. La única condición para su inclusión en el manifiesto es que se debe compilar la aplicación en contra de una versión de la plataforma que apoya a nivel de API de 14 o superior. Sólo asegúrese de que usted no usa abiertamente otras API en su código de aplicación que no son compatibles con la versión declarada por el [minSdkVersion](http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#min) atributo de sólo los atributos XML de forma segura ignorado por plataformas más antiguas.

**Utilizando el icono de la aplicación para la navegación**

**El uso de un logotipo en lugar del icono**

Por defecto, el sistema utiliza el icono de la aplicación en la barra de acción, según lo especificado por el [android:icon](http://developer.android.com/guide/topics/manifest/application-element.html#icon) atributo en el [<application>](http://developer.android.com/guide/topics/manifest/application-element.html) o [<activity>](http://developer.android.com/guide/topics/manifest/activity-element.html) elemento. Sin embargo, si también se especifica el [android:logo](http://developer.android.com/guide/topics/manifest/application-element.html#logo) atributo, entonces la barra de acción utiliza la imagen del logotipo en lugar del icono.

Un logotipo debe ser generalmente más ancho que el icono, pero no debe incluir el texto innecesario. En general, usted debe utilizar un logo sólo cuando representa su marca en un formato tradicional de que los usuarios a reconocer. Un buen ejemplo es el logotipo de la aplicación de YouTube logo representa la marca de usuario esperado, mientras que el icono de la aplicación es una versión modificada que cumpla con el requisito de plaza.

Por defecto, el icono de la aplicación aparece en la barra de acción en el lado izquierdo. Si lo desea, puede activar el icono de comportarse como un elemento de acción. En respuesta a una acción del usuario sobre el icono, la aplicación debe hacer una de dos cosas:

* Vaya a la aplicación "casa" la actividad, o
* Navegar hacia "arriba" la jerarquía estructural de la aplicación

Cuando el usuario toca el icono, el sistema llama a su actividad de [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) método con el android.R.id.home ID. En respuesta, usted debe comenzar la actividad en casa o llevar al usuario a un paso en la jerarquía estructural de la aplicación.

Si usted responde a la icono de la aplicación por volver a la actividad en el hogar, usted debe incluir el [FLAG\_ACTIVITY\_CLEAR\_TOP](http://developer.android.com/reference/android/content/Intent.html#FLAG_ACTIVITY_CLEAR_TOP) bandera en la [Intent](http://developer.android.com/reference/android/content/Intent.html) . Con este indicador, si la actividad que está empezando ya existe en la tarea actual, a continuación, todas las actividades en la parte superior de la misma se destruyen y se trajo a primer plano. La adición de este indicador es a menudo importante, porque va a "casa" es una acción que es equivalente a "volver" y no debe por lo general crear una nueva instancia de la actividad en el hogar. De lo contrario, podría terminar con una pila larga de actividades en la tarea actual, con varias instancias de la actividad en el hogar.

Por ejemplo, aquí hay una implementación de [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) que devuelve a la aplicación de la "casa" la actividad:

@ Override

public boolean onOptionsItemSelected (elemento MenuItem) {

switch (item.getItemId ()) {

caso de Android.R.id.home:

/ / Icono de la aplicación en la barra de acción de hacer clic, ir a casa

Intención de Intención = nuevo intento (esto, HomeActivity.class);

intent.addFlags (Intent.FLAG\_ACTIVITY\_CLEAR\_TOP);

startActivity (intención);

return true;

default:

volver super.onOptionsItemSelected (punto);

}

}

En caso de que el usuario puede introducir la actividad actual desde otra aplicación, es posible que también desee agregar el [FLAG\_ACTIVITY\_NEW\_TASK](http://developer.android.com/reference/android/content/Intent.html#FLAG_ACTIVITY_NEW_TASK) bandera. Esta bandera se asegura de que, cuando el usuario navega bien "casa" o "arriba", la nueva actividad **no** se agrega a la tarea actual, sino que se inició en una tarea que pertenece a su aplicación. Por ejemplo, si el usuario inicia una actividad en su aplicación a través de la intención invocada por otra aplicación, a continuación, selecciona el icono de la barra de acción para navegar en casa o arriba, el [FLAG\_ACTIVITY\_CLEAR\_TOP](http://developer.android.com/reference/android/content/Intent.html#FLAG_ACTIVITY_CLEAR_TOP) bandera comienza la actividad en una tarea que pertenece a su aplicación (no el actual tarea). El sistema o empieza una nueva tarea con su nueva actividad como la actividad de las raíces o, si existe una tarea existente en el fondo con una instancia de esa actividad, entonces esa tarea se lleva adelante y la actividad de destino recibe [onNewIntent()](http://developer.android.com/reference/android/app/Activity.html#onNewIntent%28android.content.Intent%29) . Así que si su actividad acepta los intentos de otras aplicaciones (que declara la intención de los filtros genéricos), por lo general, debe agregar el [FLAG\_ACTIVITY\_NEW\_TASK](http://developer.android.com/reference/android/content/Intent.html#FLAG_ACTIVITY_NEW_TASK) bandera a la intención de:

intent.addFlags (Intent.FLAG\_ACTIVITY\_CLEAR\_TOP | Intención.FLAG\_ACTIVITY\_NEW\_TASK);

Para obtener más información acerca de estos indicadores y otros comportamientos de la pila de espalda, lea las [tareas y Volver Stack](http://developer.android.com/guide/topics/fundamentals/tasks-and-back-stack.html) guía para el desarrollador.

**Nota:** Si usted está usando el icono para ir a la actividad en el hogar, ten en cuenta que inicio con Android 4.0 (nivel de la API 14), debe habilitar explícitamente el icono como un elemento de acción llamando a [setHomeButtonEnabled(true)](http://developer.android.com/reference/android/app/ActionBar.html#setHomeButtonEnabled%28boolean%29) (en versiones anteriores, la icono ha sido habilitado como un elemento de acción por defecto).

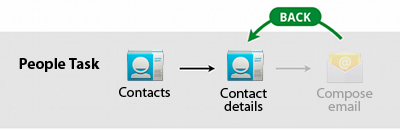
**Navegando por**



**Figura 4.** Icono estándar de la aplicación E-mail (a la izquierda) y el "navegar hacia arriba" icono (derecha). El sistema añade automáticamente el "arriba" del indicador.

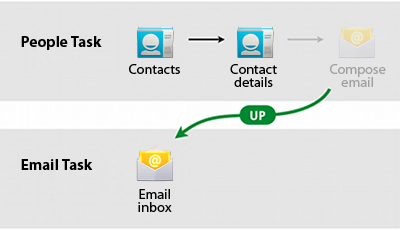
Como complemento a la tradicional "vuelta" de navegación, lo cual lleva al usuario a la pantalla anterior en la tarea de la historia, se puede activar el icono de la barra de acción para ofrecer "arriba" de navegación, que debe llevar al usuario un paso adelante en la jerarquía estructural de la aplicación . Por ejemplo, si la pantalla actual es en algún lugar profundo en la jerarquía de la aplicación, tocando el icono de la aplicación debe navegar hacia arriba un nivel, a los padres de la pantalla actual.

Por ejemplo, la figura 5 se muestra cómo se comporta el botón BACK cuando el usuario navega de una aplicación a una actividad que pertenece a una aplicación diferente (en concreto, al redactar un correo electrónico a una persona seleccionada de la aplicación de Personas).



**Figura 5.** El comportamiento del botón VOLVER después de entrar en la aplicación de correo electrónico de las personas (o contactos) de aplicaciones.

Sin embargo, si el usuario quiere permanecer dentro de la aplicación de correo electrónico después de componer la dirección de correo electrónico, hasta la navegación permite al usuario navegar hacia arriba en la aplicación de correo electrónico, en lugar de volver a la actividad anterior. La figura 6 muestra este escenario, en el que el usuario nuevo entra en la aplicación de correo electrónico, pero presiona el icono de la barra de acción para desplazarse hacia arriba, no hacia atrás.



**Figura 6.** Ejemplo de comportamiento de navegación hacia arriba después de entrar en la aplicación de correo electrónico de la aplicación de la gente.

**Diseño de navegación**

Para más información sobre cómo *arriba* y hacia *atrás* de navegación diferentes, leer Android Diseño de [navegación](http://developer.android.com/design/patterns/navigation.html) guía.

Para activar el icono para un máximo de navegación (que muestra el "arriba" indicador situado junto al icono), llame a [setDisplayHomeAsUpEnabled(true)](http://developer.android.com/reference/android/app/ActionBar.html#setDisplayHomeAsUpEnabled%28boolean%29) en su [ActionBar](http://developer.android.com/reference/android/app/ActionBar.html) :

protected void onCreate (paquete savedInstanceState) {

super.onCreate (savedInstanceState);

setContentView (R.layout.main);

Actionbar actionbar getActionBar = ();

actionBar.setDisplayHomeAsUpEnabled (true);

...

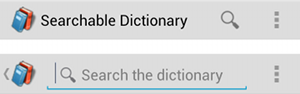
}

Cuando el usuario toca el icono, el sistema llama a su actividad de [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) método con el android.R.id.home ID, como se muestra en la sección anterior acerca de [utilizar el icono de aplicación para la navegación](http://developer.android.com/guide/topics/ui/actionbar.html#Home) .

Recuerde que debe utilizar el [FLAG\_ACTIVITY\_CLEAR\_TOP](http://developer.android.com/reference/android/content/Intent.html#FLAG_ACTIVITY_CLEAR_TOP) bandera en la [Intent](http://developer.android.com/reference/android/content/Intent.html) , de modo que usted no crea una nueva instancia de la actividad de los padres si ya existe uno. Por ejemplo, si usted no utiliza el [FLAG\_ACTIVITY\_CLEAR\_TOP](http://developer.android.com/reference/android/content/Intent.html#FLAG_ACTIVITY_CLEAR_TOP) bandera, a continuación, después de navegar hacia arriba, el botón de nuevo en realidad llevará al usuario "adelante", con respecto a la estructura de la aplicación, lo que sería extraño.

**Nota:** Si hay muchos caminos para que el usuario podría haber tenido para llegar a la actividad actual dentro de la aplicación, el icono de arriba debe desplazarse hacia atrás a lo largo de la ruta de acceso que el usuario realmente seguido para llegar a la actividad actual.

**Adición de una vista de acción**



**Figura 7.** Una barra de acción con el fin de la acción se derrumbó para la búsqueda (arriba), vista la acción luego se amplió con la [SearchView](http://developer.android.com/reference/android/widget/SearchView.html) widget (parte inferior).

Una vista de la acción es un widget que aparece en la barra de acciones como un sustituto para el botón de un elemento de acción. Por ejemplo, si usted tiene un elemento en el menú de opciones de "Buscar", se puede añadir un punto de vista la acción que reemplaza el botón con un [SearchView](http://developer.android.com/reference/android/widget/SearchView.html) widget, como se muestra en la figura 7.

Para declarar un punto de vista de acción para un artículo en su [recurso de menú](http://developer.android.com/guide/topics/resources/menu-resource.html) , utilice el android:actionLayout o android:actionViewClass atributo para especificar un recurso de diseño o de clase widget a utilizar, respectivamente. Por ejemplo:

<?xml version = "1.0" encoding = "UTF-8"?>

<Menú xmlns:android="http://schemas.android.com/apk/res/android">

<Tema android: id = "+ id @ / menu\_search"

android: title = "@ cadena / menu\_search"

android: icon = "@ dibujable / ic\_menu\_search"

android: showAsAction = "ifRoom | collapseActionView"

**android: actionViewClass = "android.widget.SearchView "/>**

</ Menu>

Observe que el android:showAsAction atributo también incluye "collapseActionView" . Esto es opcional y declara que el punto de vista la acción debe ser derrumbado en un botón. Cuando el usuario selecciona el botón, la vista de la acción se expande. De lo contrario, la opinión de la acción es visible por defecto y puede consumir un valioso espacio barra de acción, incluso cuando el usuario no lo está utilizando. Para obtener más información, consulte la siguiente sección sobre [Manejo de puntos de vista plegables de acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionViewCollapsing) .

Si es necesario agregar algunos ganchos de eventos a la vista la acción, puede hacerlo durante el [onCreateOptionsMenu()](http://developer.android.com/reference/android/app/Activity.html#onCreateOptionsMenu%28android.view.Menu%29) de devolución de llamada. Usted puede adquirir los elementos en una vista de la acción llamando a [findItem()](http://developer.android.com/reference/android/view/Menu.html#findItem%28int%29) con el ID del elemento de menú, a continuación, llamar [getActionView()](http://developer.android.com/reference/android/view/MenuItem.html#getActionView%28%29) . Por ejemplo, el widget de búsqueda en el ejemplo anterior se adquiere de esta manera:

@Override

onCreateOptionsMenu pública booleano (menú Menú) {

. getMenuInflater () se inflan (R.menu.options, menú);

SearchView SearchView = (SearchView) menu.findItem (R.id.menu\_search) getActionView ().;

/ / Configurar la información de búsqueda y añadir cualquier detector de eventos

...

retorno super.onCreateOptionsMenu (menú);

}

Para obtener más información sobre cómo utilizar el widget de búsqueda, consulte [Creación de una interfaz de búsqueda](http://developer.android.com/guide/topics/search/search-dialog.html) .

**Manejo de puntos de vista plegables de acción**

**Apoyo a Android 3.0 con una visión de acción**

El "collapseActionView" opción se añadió con Android 4.0 (nivel de la API 14). Sin embargo, si su aplicación es compatible con versiones anteriores, aún debe declarar "collapseActionView" con el fin de apoyar mejor a pantallas más pequeñas. Dispositivos con Android 4.0 y superior se mostrará el punto de vista la acción se derrumbó, mientras que las versiones anteriores funciona como fue diseñado de otra manera.

La adición de este valor, es necesario configurar el destino de generación de Android 4.0 o superior para poder compilar. Las versiones anteriores de Android caso omiso de la "collapseActionView" valor porque no lo entiendo. Sólo asegúrese de no utilizar otras API en su código fuente que no se admiten en la versión declarada por el [minSdkVersion](http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#min) , a menos que usted agregue la comprobación de la versión adecuada en tiempo de ejecución.

Puntos de vista de acción le permiten proporcionar un acceso rápido a las acciones de ricos sin necesidad de cambiar actividades o fragmentos, o sustitución de la barra de acción. Sin embargo, podría no ser apropiada para hacer un punto de vista la acción visible de forma predeterminada. A fin de preservar el espacio de la barra de acción (especialmente cuando se ejecuta en pantallas más pequeñas), puede contraer su punto de vista la acción como un control de elemento de acción. Cuando el usuario selecciona el botón, la vista de la acción aparece en la barra de acción. Cuando se derrumbó, el ​​sistema puede colocar el elemento en el menú de desbordamiento si se ha definido android:showAsAction con "ifRoom" , pero la vista la acción sigue apareciendo en la barra de acción cuando el usuario selecciona el elemento. Usted puede hacer su acción de ver plegable mediante la adición de "collapseActionView" a la android:showAsAction atributo, como se muestra en el XML de arriba.

Debido a que el sistema se amplíe la vista de la acción cuando el usuario selecciona el elemento, por lo que *no es* necesario para responder a el tema en el [onOptionsItemSelected](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) de devolución de llamada. El sistema todavía llama [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) cuando el usuario lo selecciona, pero el sistema siempre se amplíe la vista de la acción a menos que devuelva true (lo que indica que usted ha manejado el caso en su lugar).

El sistema también se derrumba la acción de la vista cuando el usuario selecciona el "arriba" icono en la barra de acción o pulsa el botón BACK.

Si es necesario, puede expandir o contraer la vista de la acción en su propio código llamando [expandActionView()](http://developer.android.com/reference/android/view/MenuItem.html#expandActionView%28%29) y [collapseActionView()](http://developer.android.com/reference/android/view/MenuItem.html#collapseActionView%28%29) en el [MenuItem](http://developer.android.com/reference/android/view/MenuItem.html) .

**Nota:** Aunque el colapso de su punto de vista la acción es opcional, le recomendamos que siempre se derrumban a su punto de vista la acción si se incluye [SearchView](http://developer.android.com/reference/android/widget/SearchView.html) . También tenga en cuenta que algunos dispositivos cuentan con un botón de búsqueda dedicada y debe ampliar su visión de acción de búsqueda si el usuario presiona el botón BUSCAR. Basta con sustituir la actividad de [onKeyUp()](http://developer.android.com/reference/android/app/Activity.html#onKeyUp%28int,%20android.view.KeyEvent%29) el método de devolución de llamada, escuche el [KEYCODE\_SEARCH](http://developer.android.com/reference/android/view/KeyEvent.html#KEYCODE_SEARCH) evento, a continuación, llamar [expandActionView()](http://developer.android.com/reference/android/view/MenuItem.html#expandActionView%28%29) .

Si usted necesita poner al día su actividad en base a la visibilidad de su punto de vista la acción, puede recibir devoluciones de llamada cuando se amplió y se derrumbó por la definición de un [OnActionExpandListener](http://developer.android.com/reference/android/view/MenuItem.OnActionExpandListener.html) y registrarlo con [setOnActionExpandListener()](http://developer.android.com/reference/android/view/MenuItem.html#setOnActionExpandListener%28android.view.MenuItem.OnActionExpandListener%29) . Por ejemplo:

@ Override

onCreateOptionsMenu pública booleano (menú Menú) {

. getMenuInflater () se inflan (R.menu.options, menú);

MenuItem menuitem = menu.findItem (R.id.actionItem);

...

menuItem.setOnActionExpandListener (OnActionExpandListener nuevo () {

@ Override

onMenuItemActionCollapse pública booleano (tema MenuItem) {

/ / Hacer algo cuando se derrumbó

return true; / / Devuelve verdadero colapso de vista de la acción

}

@ Override

public boolean onMenuItemActionExpand (elemento MenuItem) {

/ / Hacer algo cuando se expande

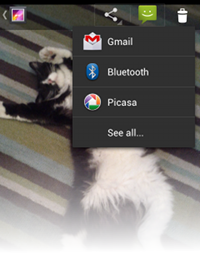
return true; / / Devuelve verdadero punto de vista de ampliar la acción

}

});

}

**Adición de un proveedor de Acción**



**Figura 8.** Captura de pantalla de la aplicación Galería, con la [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) submenú ampliado para mostrar los objetivos de las acciones.

Similar a un [punto de vista la acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionView) , un proveedor de la acción (definido por el [ActionProvider](http://developer.android.com/reference/android/view/ActionProvider.html) clase) reemplaza un elemento de acción, con un diseño personalizado, sino que también toma el control de todos los comportamientos del elemento. Cuando se declara un proveedor de acción para un elemento de menú en la barra de acción, no sólo controla la apariencia del elemento de la barra de acción, con un diseño personalizado, sino que también controla el evento predeterminado para el elemento de menú cuando aparece en el menú de desbordamiento . También puede proporcionar un submenú de cualquiera de la barra de acción o el menú de desbordamiento.

Por ejemplo, el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) es una extensión de [ActionProvider](http://developer.android.com/reference/android/view/ActionProvider.html) que facilita una "parte" de acción, mostrando una lista de objetivos de acciones disponibles en la barra de acción. En lugar de utilizar un elemento de acción tradicional que invoca la [ACTION\_SEND](http://developer.android.com/reference/android/content/Intent.html#ACTION_SEND) intención, se puede declarar una instancia de [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) para manejar un elemento de acción. Este proveedor de la acción presenta una visión de acción con una lista desplegable de aplicaciones que manejan el [ACTION\_SEND](http://developer.android.com/reference/android/content/Intent.html#ACTION_SEND) intención, incluso cuando el elemento de menú aparece en el menú de desbordamiento. Por lo tanto, cuando se utiliza un proveedor de una acción como ésta, usted no tiene que manejar los eventos de usuario en el menú.

Para declarar un proveedor de acción para un elemento de acción, definir el android:actionProviderClass de atributos para el caso, la <item> elemento en su [recurso de menú](http://developer.android.com/guide/topics/resources/menu-resource.html) , utilizando el nombre de clase completo del proveedor de la acción. Por ejemplo:

<?xml version = "1.0" encoding = "UTF-8"?>

<Menú xmlns:android="http://schemas.android.com/apk/res/android">

<Tema android: id = "+ id @ / menu\_share"

android: title = "@ string / share"

android: showAsAction = "ifRoom"

**android: actionProviderClass = "android.widget.ShareActionProvider "/>**

...

</ Menu>

En este ejemplo, el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) se utiliza como el proveedor de acción. En este punto, el proveedor de la acción oficial se hace con el elemento de menú y se ocupa de su aspecto y comportamiento en la barra de acciones y su comportamiento en el menú de desbordamiento. No obstante, debe proporcionar un título de texto para el elemento que se utiliza en el menú de desbordamiento.

Aunque el proveedor de la acción se puede realizar la acción predeterminada para el elemento de menú cuando aparece en el menú de desbordamiento, su actividad (o fragmento) puede reemplazar dicho comportamiento también se controla el evento de clic de la [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) el método de devolución de llamada. Si no controlar el evento en el que la devolución de llamada, entonces el proveedor de la acción recibe el [onPerformDefaultAction()](http://developer.android.com/reference/android/view/ActionProvider.html#onPerformDefaultAction%28%29) de devolución de llamada para controlar el evento. Sin embargo, si el proveedor de acción ofrece un submenú, entonces su actividad no recibirá el [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) de devolución de llamada, ya que el submenú se muestra en lugar de invocar el menú por defecto comportamiento de elemento cuando se selecciona.

**Usando el ShareActionProvider**

Si desea proporcionar un "compartir" la acción en tu barra de acción mediante el aprovechamiento de otras aplicaciones instaladas en el dispositivo (por ejemplo, para compartir una foto con una aplicación de mensajería o social), y luego usando [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) es una manera efectiva de hacerlo, en lugar que la adición de un elemento de acción que invoca el [ACTION\_SEND](http://developer.android.com/reference/android/content/Intent.html#ACTION_SEND) intención. Cuando se utiliza [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) de un elemento de acción, que presenta una visión de acción con una lista desplegable de aplicaciones que manejan el [ACTION\_SEND](http://developer.android.com/reference/android/content/Intent.html#ACTION_SEND) intención (como se muestra en la figura 8).

Toda la lógica de la creación del submenú, rellenarlo con los objetivos de acción, y el manejo de eventos de clic (incluso cuando el tema aparece en el menú de desbordamiento) es implementado por el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) -el único código que necesita para escribir es declarar que el proveedor de acción para la del menú y especifique la calidad de parte.

Por defecto, el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) conserva una clasificación para cada objetivo de cuota sobre la base de la frecuencia con la que el usuario selecciona cada una de ellas. La cuota de los objetivos utilizados con más frecuencia aparecen en la parte superior de la lista desplegable y el objetivo de utilizar con mayor frecuencia aparece directamente en la barra de acciones como el objetivo compartido predeterminado. De forma predeterminada, la información de clasificación se guardan en un archivo privado con un nombre especificado por [DEFAULT\_SHARE\_HISTORY\_FILE\_NAME](http://developer.android.com/reference/android/widget/ShareActionProvider.html#DEFAULT_SHARE_HISTORY_FILE_NAME) . Si utiliza el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) o una extensión de la misma para un solo tipo de acción, entonces usted debe seguir para utilizar este archivo por defecto la historia y no hay nada que usted necesita hacer. Sin embargo, si utiliza [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) o una extensión de la misma para varias acciones con significados semánticamente diferentes, entonces cada [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) debe especificar el archivo de la propia historia con el fin de mantener su propia historia. Para especificar un archivo de historia diferente para el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) , llamada [setShareHistoryFileName()](http://developer.android.com/reference/android/widget/ShareActionProvider.html#setShareHistoryFileName%28java.lang.String%29) y proporcionar un nombre de archivo XML (por ejemplo, "custom\_share\_history.xml" ).

**Nota:** Aunque el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) cuota de rangos objetivos basados ​​en la frecuencia de uso, el comportamiento es extensible y las extensiones de [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) puede llevar a cabo diferentes comportamientos y clasificación basados ​​en el archivo de la historia (si procede).

Para agregar [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) , basta con definir el android:actionProviderClass atributo con "android.widget.ShareActionProvider" , como se muestra en el ejemplo XML anterior. Lo único que queda por hacer es definir la [Intent](http://developer.android.com/reference/android/content/Intent.html) que desea utilizar para el intercambio. Para ello, debe llamar a [getActionProvider()](http://developer.android.com/reference/android/view/MenuItem.html#getActionProvider%28%29) para recuperar el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) que está asociada con un [MenuItem](http://developer.android.com/reference/android/view/MenuItem.html) , a continuación, llamar [setShareIntent()](http://developer.android.com/reference/android/widget/ShareActionProvider.html#setShareIntent%28android.content.Intent%29) .

Si el formato de la intención de acción depende de la opción seleccionada o de otras variables que cambian durante el ciclo de vida de la actividad, debe guardar el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) en un campo de miembro y actualizarlo llamando [setShareIntent()](http://developer.android.com/reference/android/widget/ShareActionProvider.html#setShareIntent%28android.content.Intent%29) según sea necesario. Por ejemplo:

privada mShareActionProvider ShareActionProvider;

...

@ Override

onCreateOptionsMenu pública booleano (menú Menú) {

mShareActionProvider = (ShareActionProvider) menu.findItem (R.id.menu\_share) getActionProvider ().;

/ / Si utiliza más de un ShareActionProvider, cada uno por una acción diferente,

/ / Utilizar la línea siguiente para especificar un archivo de la historia única para cada uno.

/ / MShareActionProvider.setShareHistoryFileName ("custom\_share\_history.xml");

/ / Establecer la intención recurso por defecto

mShareActionProvider.setShareIntent (getDefaultShareIntent ());

return true;

}

/ / Cuando es necesario actualizar la intención de participación de algún otro lugar en la aplicación, llame al

/ / MShareActionProvider. [setShareIntent()](http://developer.android.com/reference/android/widget/ShareActionProvider.html#setShareIntent%28android.content.Intent%29)

El [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) ahora se encarga de toda la interacción del usuario con el tema y *no es* necesario para manejar los eventos de clic de la [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) el método de devolución de llamada.

Para obtener una muestra mediante el proveedor de la acción social, véase [ActionBarShareActionProviderActivity](http://developer.android.com/resources/samples/ApiDemos/src/com/example/android/apis/app/ActionBarShareActionProviderActivity.html) .

**Creación de un proveedor de acción personalizada**

Cuando se desea crear un punto de vista la acción que tiene comportamiento dinámico y una acción predeterminada en el menú de desbordamiento, que se extiende [ActionProvider](http://developer.android.com/reference/android/view/ActionProvider.html) para definir las conductas es una buena solución. Creación de su proveedor de la propia acción que ofrece un componente organizado y reutilizables, en lugar de manejar las transformaciones de acción diferentes elementos y comportamientos en el fragmento o código de actividad. Como se muestra en la sección anterior, Android proporciona una implementación de [ActionProvider](http://developer.android.com/reference/android/view/ActionProvider.html) para las acciones de participación: el [ShareActionProvider](http://developer.android.com/reference/android/widget/ShareActionProvider.html) .

Para crear su cuenta, simplemente se extiende el [ActionProvider](http://developer.android.com/reference/android/view/ActionProvider.html) clase y poner en práctica sus métodos de devolución de llamada, según corresponda. Lo más importante, se deben implementar las siguientes acciones:

[ActionProvider()](http://developer.android.com/reference/android/view/ActionProvider.html#ActionProvider%28android.content.Context%29)

Este constructor se pasa a la aplicación [Context](http://developer.android.com/reference/android/content/Context.html) , que debe guardar en un campo de miembro para utilizar en los métodos de devolución de llamada de otros.

[onCreateActionView()](http://developer.android.com/reference/android/view/ActionProvider.html#onCreateActionView%28%29)

Aquí es donde se define el punto de vista de acción para el elemento. Utilice el [Context](http://developer.android.com/reference/android/content/Context.html) adquiere desde el constructor para crear instancias de un [LayoutInflater](http://developer.android.com/reference/android/view/LayoutInflater.html) e inflar el diseño de vista la acción de un recurso XML, a continuación, conectar detectores de eventos. Por ejemplo:

pública onCreateActionView Vista () {

/ / Inflar el punto de vista la acción que se muestre en la barra de acción.

LayoutInflater layoutInflater = LayoutInflater.from (mContext);

Ver vista = layoutInflater.inflate (R.layout.action\_provider, null);

ImageButton botón = (ImageButton) view.findViewById (R.id.button);

button.setOnClickListener (Ver nueva.OnClickListener () {

@ Override

public void onClick (View v) {

/ / Hacer algo ...

}

});

volver ver;

}

[onPerformDefaultAction()](http://developer.android.com/reference/android/view/ActionProvider.html#onPerformDefaultAction%28%29)

El sistema llama a esto cuando el elemento de menú se selecciona del menú de desbordamiento y el proveedor de la acción debe llevar a cabo una acción predeterminada para el elemento de menú.

Sin embargo, si su proveedor de acción proporciona un submenú, a través de la [onPrepareSubMenu()](http://developer.android.com/reference/android/view/ActionProvider.html#onPrepareSubMenu%28android.view.SubMenu%29) de devolución de llamada, luego el submenú aparece incluso cuando el elemento del menú se encuentra en el menú de desbordamiento. Por lo tanto, [onPerformDefaultAction()](http://developer.android.com/reference/android/view/ActionProvider.html#onPerformDefaultAction%28%29) no se llama cuando hay un submenú.

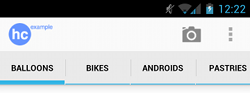
**Nota:** Una actividad o un fragmento que implementa [onOptionsItemSelected()](http://developer.android.com/reference/android/app/Activity.html#onOptionsItemSelected%28android.view.MenuItem%29) puede reemplazar el comportamiento del proveedor acción predeterminado controlando el evento de elemento seleccionado (y volviendo verdadero), en cuyo caso, el sistema no requiere [onPerformDefaultAction()](http://developer.android.com/reference/android/view/ActionProvider.html#onPerformDefaultAction%28%29) .

Para un ejemplo la extensión de [ActionProvider](http://developer.android.com/reference/android/view/ActionProvider.html) , consulte [ActionBarSettingsActionProviderActivity](http://developer.android.com/resources/samples/ApiDemos/src/com/example/android/apis/app/ActionBarSettingsActionProviderActivity.html) .

**Agregar fichas de navegación**

http://developer.android.com/images/ui/actionbar.png

**Figura 9.** Captura de pantalla de las pestañas de la barra de acciones [Galería de nido de abeja](http://developer.android.com/resources/samples/HoneycombGallery/index.html) aplicación.



**Figura 10.** Captura de pantalla de las pestañas en la barra de acción apilados en una pantalla estrecha.

Cuando se desea proporcionar fichas de navegación en una actividad, uso de las pestañas de la barra de acción es una gran opción (en lugar de utilizar [TabWidget](http://developer.android.com/reference/android/widget/TabWidget.html) ), porque el sistema adapta las fichas de la barra de acciones para la pantalla diferente a los tamaños de colocación de la barra de acción principal, cuando la pantalla es lo suficientemente amplia, o en una barra separada (conocida como la "barra de acción apilada") cuando la pantalla es demasiado estrecha, como se muestra en las figuras 9 y 10.

Para cambiar entre los fragmentos utilizando las fichas, debe realizar una transacción fragmento cada vez que se selecciona una ficha. Si no estás familiarizado con la forma de cambiar fragmentos utilizando [FragmentTransaction](http://developer.android.com/reference/android/app/FragmentTransaction.html) , primero lea la [Fragmentos](http://developer.android.com/guide/topics/fundamentals/fragments.html) guía para el desarrollador.

Para empezar, su diseño debe incluir un [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html) en el que se coloca cada [Fragment](http://developer.android.com/reference/android/app/Fragment.html) asociado con una ficha. Asegúrese de que el [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html) tiene un identificador de recursos para que pueda hacer referencia a ella desde la pestaña de intercambio de código. Por otra parte, si el contenido de la ficha se llenará el diseño de la actividad (con exclusión de la barra de acciones), entonces su actividad no necesita un diseño en absoluto (ni siquiera tiene que llamar [setContentView()](http://developer.android.com/reference/android/app/Activity.html#setContentView%28android.view.View%29) ). En su lugar, puede colocar cada fragmento de la raíz por defecto [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html) , que puede consultar con el android.R.id.content ID (se puede ver este identificador utilizado en el código de ejemplo siguiente, durante las transacciones de fragmentos).

Una vez que determine que los fragmentos aparecen en el diseño, el procedimiento básico para agregar fichas es la siguiente:

1. Implementar el [ActionBar.TabListener](http://developer.android.com/reference/android/app/ActionBar.TabListener.html) interfaz. Devoluciones de llamada de esta interfaz de responder a los eventos del usuario en las fichas para que pueda intercambiar fragmentos.
2. Por cada ficha que desea agregar, una instancia de un [ActionBar.Tab](http://developer.android.com/reference/android/app/ActionBar.Tab.html) y establecer el [ActionBar.TabListener](http://developer.android.com/reference/android/app/ActionBar.TabListener.html) llamando [setTabListener()](http://developer.android.com/reference/android/app/ActionBar.Tab.html#setTabListener%28android.app.ActionBar.TabListener%29) . También se establece el título de la ficha y / o el icono con [setText()](http://developer.android.com/reference/android/app/ActionBar.Tab.html#setText%28int%29) y / o [setIcon()](http://developer.android.com/reference/android/app/ActionBar.Tab.html#setIcon%28android.graphics.drawable.Drawable%29) .
3. Añadir cada pestaña a la barra de acción llamando a [addTab()](http://developer.android.com/reference/android/app/ActionBar.html#addTab%28android.app.ActionBar.Tab%29) .

Al mirar el [ActionBar.TabListener](http://developer.android.com/reference/android/app/ActionBar.TabListener.html) la interfaz, el aviso de que los métodos de devolución de llamada proporcionar sólo la [ActionBar.Tab](http://developer.android.com/reference/android/app/ActionBar.Tab.html) que se ha seleccionado y una [FragmentTransaction](http://developer.android.com/reference/android/app/FragmentTransaction.html) para que usted pueda realizar transacciones-que los fragmentos no dice nada acerca de lo que usted debe cambiar fragmento de dentro o fuera. Por lo tanto, debe definir su propia asociación entre cada [ActionBar.Tab](http://developer.android.com/reference/android/app/ActionBar.Tab.html) y el apropiado [Fragment](http://developer.android.com/reference/android/app/Fragment.html) que representa (a fin de realizar la transacción fragmento apropiado). Hay varias maneras de definir la asociación, en función de su diseño. En el siguiente ejemplo, el [ActionBar.TabListener](http://developer.android.com/reference/android/app/ActionBar.TabListener.html) la aplicación proporciona un constructor de manera que cada nueva pestaña utiliza su propia instancia del oyente. Cada instancia del detector define varios campos que son necesarios para llevar a cabo más adelante una transacción en el fragmento correspondiente.

Por ejemplo, aquí es cómo se puede implementar el [ActionBar.TabListener](http://developer.android.com/reference/android/app/ActionBar.TabListener.html) de tal manera que cada pestaña utiliza su propia instancia del oyente:

<T pública estática TabListener clase extiende Fragment> implementa actionbar.TabListener {

Fragmento mFragment privada;

mActivity privado final;

MTag privada final String;

final privado Clase <T> mCLASS;

/ \*\* Constructor utilizado cada vez que se crea una nueva pestaña.

\* @ Param La actividad de la Actividad de acogida, que se utiliza para crear una instancia del fragmento

\* @ Param etiqueta etiqueta de identificación para el fragmento

\* @ Param ClZ El fragmento de la clase, que se utiliza para crear una instancia del fragmento

\* /

TabListener público (la actividad de la actividad, la etiqueta de cadena, Clase <T> ClZ) {

mActivity = actividad;

MTAG = etiqueta;

mCLASS = ClZ;

}

/ \* Las siguientes son cada uno de los [ActionBar.TabListener](http://developer.android.com/reference/android/app/ActionBar.TabListener.html) devoluciones de llamada \* /

public void onTabSelected (ficha Tab, FragmentTransaction pies) {

/ / Comprobar si el fragmento ya se ha inicializado

if (mFragment == null) {

/ / Si no, crear una instancia y agregarlo a la actividad

mFragment = Fragment.instantiate (mActivity, mClass.getName ());

ft.add (android.R.id.content, mFragment, MTag);

} else {

/ / Si existe, sólo tiene que conectar con el fin de mostrar

ft.attach (mFragment);

}

}

public void onTabUnselected (ficha Tab, FragmentTransaction pies) {

if (mFragment!= Null) {

/ / Desata el fragmento, porque otro se va a adjuntar

ft.detach (mFragment);

}

}

public void onTabReselected (ficha Tab, FragmentTransaction pies) {

/ / El usuario selecciona la pestaña ya seleccionada. Por lo general, no hacer nada.

}

}

**Precaución: No debe** llamar a [commit()](http://developer.android.com/reference/android/app/FragmentTransaction.html#commit%28%29) para la transacción fragmento en cada una de estas devoluciones de llamada, el sistema que te llama y se puede producir una excepción si se llama a sí mismo. También **no se puede** agregar fragmentos de estas transacciones a la pila de la espalda.

En este ejemplo, el oyente se conecta simplemente ( [attach()](http://developer.android.com/reference/android/app/FragmentTransaction.html#attach%28android.app.Fragment%29) ) un fragmento de la actividad de diseño, o si no se crea una instancia, se crea el fragmento y agrega ( [add()](http://developer.android.com/reference/android/app/FragmentTransaction.html#add%28android.app.Fragment,%20java.lang.String%29) ) para el diseño (como un niño de la android.R.id.content grupo de opinión), cuando en la ficha correspondiente se selecciona y separa ( [detach()](http://developer.android.com/reference/android/app/FragmentTransaction.html#detach%28android.app.Fragment%29) ) cuando la ficha no está seleccionada.

El [ActionBar.TabListener](http://developer.android.com/reference/android/app/ActionBar.TabListener.html) aplicación es la mayor parte del trabajo. Todo lo que queda es la creación de cada uno de [ActionBar.Tab](http://developer.android.com/reference/android/app/ActionBar.Tab.html) y agregarlo a la [ActionBar](http://developer.android.com/reference/android/app/ActionBar.html) . Además, debe llamar a [setNavigationMode(NAVIGATION\_MODE\_TABS)](http://developer.android.com/reference/android/app/ActionBar.html#setNavigationMode%28int%29) para hacer las fichas visibles. Usted también puede desactivar el título de la actividad llamando [setDisplayShowTitleEnabled(false)](http://developer.android.com/reference/android/app/ActionBar.html#setDisplayShowTitleEnabled%28boolean%29) si los títulos de las fichas en realidad indican la vista actual.

Por ejemplo, el código siguiente agrega dos pestañas utilizando el oyente se ha definido anteriormente:

@ Override

protected void onCreate (paquete savedInstanceState) {

super.onCreate (savedInstanceState);

/ / Tenga en cuenta que setContentView () no se utiliza, debido a que usamos la raíz

/ / Androide.R.id.content como contenedor para cada fragmento

/ / Configuración de barra de acción para las pestañas

Actionbar actionbar getActionBar = ();

actionBar.setNavigationMode (ActionBar.NAVIGATION\_MODE\_TABS);

actionBar.setDisplayShowTitleEnabled (false);

Tab tab = actionBar.newTab ()

. SetText (R.string.artist)

. SetTabListener (nuevo TabListener <ArtistFragment> (

esto, "artista", ArtistFragment.class));

actionBar.addTab (ficha);

tab = actionBar.newTab ()

. SetText (R.string.album)

. SetTabListener (nuevo TabListener <AlbumFragment> (

esto, "álbum", AlbumFragment.class));

actionBar.addTab (ficha);

}

**Nota:** La implementación de arriba para [ActionBar.TabListener](http://developer.android.com/reference/android/app/ActionBar.TabListener.html) es una de varias técnicas posibles. Puedes ver más de este estilo en la [API Demos](http://developer.android.com/resources/samples/ApiDemos/src/com/example/android/apis/app/FragmentTabs.html) aplicación.

Si su actividad se detiene, debe conservar la ficha actualmente seleccionada con el [estado de la instancia salvo](http://developer.android.com/guide/topics/fundamentals/activities.html#SavingActivityState) por lo que puede abrir la ficha correspondiente cuando el usuario vuelve. Cuando es el momento de guardar el estado, se puede consultar la ficha seleccionada actualmente con [getSelectedNavigationIndex()](http://developer.android.com/reference/android/app/ActionBar.html#getSelectedNavigationIndex%28%29) . Esto devuelve la posición de índice de la pestaña seleccionada.

**Precaución:** Es importante que guarde el estado de cada fragmento según sea necesario, de modo que cuando los usuarios cambian los fragmentos con las pestañas y luego regresar a un fragmento anterior, se ve como lo hizo cuando se fueron. Para información sobre cómo guardar el estado de su fragmento, ver el [Fragmentos](http://developer.android.com/guide/topics/fundamentals/fragments.html) guía para el desarrollador.

**Nota:** En algunos casos, el sistema Android mostrará las pestañas barra de acciones como una lista desplegable con el fin de garantizar el mejor ajuste en la barra de acción.

**Añadir desplegable de navegación**

Como otro modo de navegación (o filtrado) dentro de su actividad, la barra de acción ofrece un sistema incorporado en la lista desplegable. Por ejemplo, la lista desplegable puede ofrecer diferentes modos en que se ordena el contenido de la actividad.

El procedimiento básico para permitir desplegable de navegación es:

1. Crear un [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) que proporciona la lista de elementos seleccionables para la lista desplegable y la disposición a utilizar en la elaboración de cada elemento de la lista.
2. Implementar [ActionBar.OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) para definir el comportamiento que se produce cuando el usuario selecciona un elemento de la lista.
3. Activar el modo de navegación de la barra de acción con [setNavigationMode()](http://developer.android.com/reference/android/app/ActionBar.html#setNavigationMode%28int%29) . Por ejemplo:
4. Actionbar actionbar getActionBar = ();
5. actionBar.setNavigationMode (ActionBar.NAVIGATION\_MODE\_LIST);

**Nota:** Usted debe realizar esta actividad durante el de [onCreate()](http://developer.android.com/reference/android/app/Activity.html#onCreate%28android.os.Bundle%29) método.

1. Establecer la devolución de llamada de la lista desplegable con [setListNavigationCallbacks()](http://developer.android.com/reference/android/app/ActionBar.html#setListNavigationCallbacks%28android.widget.SpinnerAdapter,%20android.app.ActionBar.OnNavigationListener%29) . Por ejemplo:
2. actionBar.setListNavigationCallbacks (mSpinnerAdapter, mNavigationCallback);

Este método tiene sus [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) y [ActionBar.OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) .

Esa es la configuración básica. Sin embargo, la aplicación de las [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) y [ActionBar.OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) es donde la mayor parte del trabajo está hecho. Hay muchas maneras que usted puede implementar estos para definir la funcionalidad de su desplegable de navegación y ejecución de los distintos tipos de [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) está fuera del alcance de este documento (que debe referirse a la [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) referencia de clase para más información). Sin embargo, a continuación es un ejemplo simple de un [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) y [ActionBar.OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) para empezar (haga clic en el título para revelar la muestra).

[**http://developer.android.com/assets/images/triangle-closed.pngSpinnerAdapter ejemplo y OnNavigationListener**](http://developer.android.com/guide/topics/ui/actionbar.html)

[SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) es un adaptador que proporciona datos para un control giratorio, como la lista desplegable en la barra de acción. [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) es una interfaz que se puede implementar, pero Android incluye algunas implementaciones útiles que se pueden ampliar, como [ArrayAdapter](http://developer.android.com/reference/android/widget/ArrayAdapter.html) y [SimpleCursorAdapter](http://developer.android.com/reference/android/widget/SimpleCursorAdapter.html) . Por ejemplo, aquí está una manera fácil de crear un [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) mediante el uso de [ArrayAdapter](http://developer.android.com/reference/android/widget/ArrayAdapter.html) aplicación, que utiliza una matriz de cadenas como la fuente de datos:

SpinnerAdapter mSpinnerAdapter = ArrayAdapter.createFromResource (esto, R.array.action\_list,

androide.R.layout.simple\_spinner\_dropdown\_item);

El [createFromResource()](http://developer.android.com/reference/android/widget/ArrayAdapter.html#createFromResource%28android.content.Context,%20int,%20int%29) método toma tres parámetros: la aplicación [Context](http://developer.android.com/reference/android/content/Context.html) , la identificación de recursos para la matriz de cadenas, y la disposición a utilizar para cada elemento de la lista.

Una [matriz de cadena](http://developer.android.com/guide/topics/resources/string-resource.html#StringArray) se define en un recurso se parece a esto:

<?xml version = "1.0" encoding = "UTF-8"?>

<resources>

<string-array name="action\_list">

<item> Mercurio </ item>

<item> Venus </ item>

<item> la Tierra </ item>

</ String-> array

</ Pre>

Los [ArrayAdapter](http://developer.android.com/reference/android/widget/ArrayAdapter.html) devueltos por [createFromResource()](http://developer.android.com/reference/android/widget/ArrayAdapter.html#createFromResource%28android.content.Context,%20int,%20int%29) está completo y listo para que usted pueda pasar a [setListNavigationCallbacks()](http://developer.android.com/reference/android/app/ActionBar.html#setListNavigationCallbacks%28android.widget.SpinnerAdapter,%20android.app.ActionBar.OnNavigationListener%29) (en el paso 4 de arriba). Antes de hacerlo, sin embargo, es necesario crear el [OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) .

La implementación de [ActionBar.OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) es donde se manejan los cambios de fragmentos u otras modificaciones a su actividad cuando el usuario selecciona un elemento de la lista desplegable. Sólo hay un método de devolución de llamada para poner en práctica en el oyente: [onNavigationItemSelected()](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html#onNavigationItemSelected%28int,%20long%29) .

El [onNavigationItemSelected()](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html#onNavigationItemSelected%28int,%20long%29) método recibe la posición del elemento de la lista y un identificador de elemento único, proporcionado por los [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) .

He aquí un ejemplo que crea instancias de una aplicación anónima de [OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) , que inserta un [Fragment](http://developer.android.com/reference/android/app/Fragment.html) en el contenedor de diseño identifica por R.id.fragment\_container :

mOnNavigationListener = new OnNavigationListener () {

/ / Obtener las cuerdas que se dan para ArrayAdapter el desplegable de

String [] cadenas = getResources () getStringArray (R.array.action\_list).;

@ Override

public boolean onNavigationItemSelected (posición int, long itemId) {

/ / Crear nuevo fragmento de nuestra clase propia de fragmentos

ListContentFragment newFragment ListContentFragment = new ();

FragmentTransaction pies = openFragmentTransaction ();

/ / Cambiar lo que está en el contenedor fragmento con este fragmento

/ / Y dar el fragmento de un nombre de etiqueta igual a la cadena en la posición seleccionada

ft.replace (R.id.fragment\_container, newFragment, las cadenas [posición]);

/ / Aplicar los cambios

ft.commit ();

return true;

}

};

Esta instancia de [OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) es completo y ahora se puede llamar a [setListNavigationCallbacks()](http://developer.android.com/reference/android/app/ActionBar.html#setListNavigationCallbacks%28android.widget.SpinnerAdapter,%20android.app.ActionBar.OnNavigationListener%29) (en el paso 4), pasando por los [ArrayAdapter](http://developer.android.com/reference/android/widget/ArrayAdapter.html) y esto [OnNavigationListener](http://developer.android.com/reference/android/app/ActionBar.OnNavigationListener.html) .

En este ejemplo, cuando el usuario selecciona un elemento de la lista desplegable, un fragmento se añade a la disposición (en sustitución del fragmento actual en el R.id.fragment\_container vista). El fragmento añadido se da una etiqueta que lo identifica singularmente, que es la misma cadena utilizado para identificar el fragmento en la lista desplegable.

He aquí un vistazo a la ListContentFragment clase que define cada fragmento en este ejemplo:

ListContentFragment public class {Fragmento

private String mText;

@ Override

public void onAttach (actividad Actividad) {

/ / Esta es la devolución de llamada recibió por primera vez, aquí se puede establecer el texto de

/ / Fragmento la definida por la etiqueta especificada durante la transacción fragmento

super.onAttach (actividad);

mText getTag = ();

}

@ Override

pública onCreateView Ver (LayoutInflater inflater, contenedor ViewGroup,

Paquete savedInstanceState) {

/ / Esto se le llama para definir el diseño del fragmento;

/ / Que acabamos de crear una TextView y establezca su texto a ser la etiqueta de fragmento

TextView texto = new TextView (getActivity ());

text.setText (mText);

devolver el texto;

}

}

**El estilo de la barra de acciones**

Si usted ha implementado un diseño personalizado para los widgets de la aplicación, también puede que quiera volver a diseñar algunas de la barra de acción para que coincida con el diseño de su aplicación. Para ello, es necesario utilizar Android en [el estilo y el tema de](http://developer.android.com/guide/topics/ui/themes.html) marco para cambiar el estilo de la barra de acción utilizando las propiedades especiales de estilo.

**Nota:** Para que las imágenes de fondo para cambiar la apariencia en función del estado del botón actual (seleccionado, presiona, no seleccionado), el recurso dibujable que utilice debe ser una [lista del estado dibujable](http://developer.android.com/guide/topics/resources/drawable-resource.html#StateList) .

**Precaución:** Para todos los dibujables de fondo que usted proporciona, asegúrese de usar [parches de nueve dibujables](http://developer.android.com/guide/topics/graphics/2d-graphics.html#nine-patch) para permitir el estiramiento. La imagen de nueve parche debe *ser* inferior a 40px 30px de alto y de ancho (para el activo MDPI).

**Aspecto general**

[android:windowActionBarOverlay](http://developer.android.com/reference/android/R.attr.html#windowActionBarOverlay)

Declara si la barra de acción debe superponer el diseño de la actividad en lugar de compensar la posición de la actividad de distribución (por ejemplo, la aplicación Gallery utiliza el modo de superposición). Esto es false por defecto.

Normalmente, la barra de acción requiere su propio espacio en la pantalla y su diseño de la actividad se llena de lo que queda. Cuando la barra de acción está en el modo de superposición, el diseño de la actividad utiliza todo el espacio disponible y el sistema de la barra de acción se basa en la parte superior. El modo de superposición puede ser útil si usted quiere que su contenido para mantener un tamaño fijo y la posición de la barra de acciones, cuando se oculta y se muestra. También te recomendamos usarlo simplemente como un efecto visual, porque se puede utilizar un fondo semi-transparente de la barra de acción para que el usuario aún se pueden ver parte de su trazado actividad detrás de la barra de acción.

**Nota:** Los [Holo](http://developer.android.com/reference/android/R.style.html#Theme_Holo) familias temáticos señalar a la barra de acción con un fondo semi-transparente de forma predeterminada. Sin embargo, usted puede modificar con sus propios estilos y la [DeviceDefault](http://developer.android.com/reference/android/R.style.html#Theme_DeviceDefault) el tema en diferentes dispositivos podría utilizar un fondo opaco de forma predeterminada.

Cuando el modo de superposición está activada, el diseño de la actividad no tiene conciencia de la barra de acción por la que se en la parte superior de la misma. Por lo tanto, hay que tener cuidado de no colocar toda la información importante o de los componentes de interfaz de usuario en el área superpuesta por la barra de acción. Si es apropiado, se puede hacer referencia al valor de la plataforma de [actionBarSize](http://developer.android.com/reference/android/R.attr.html#actionBarSize) para determinar la altura de la barra de acción, haciendo referencia a que en su diseño XML. Por ejemplo:

<SomeView

...

android: layout\_marginTop = "?android: attr / actionBarSize "/>

También puede recuperar la altura de la barra de acción en tiempo de ejecución con [getHeight()](http://developer.android.com/reference/android/app/ActionBar.html#getHeight%28%29) . Esto refleja la altura de la barra de acción en el momento que se llama, que no puede incluir la barra de acción apilados (debido a las pestañas de navegación) si se llama durante los primeros métodos de la actividad del ciclo de vida. Para ver cómo se puede determinar la altura total en tiempo de ejecución, incluyendo la barra de acción apiladas, ver el [TitlesFragment](http://developer.android.com/resources/samples/HoneycombGallery/src/com/example/android/hcgallery/TitlesFragment.html) clase en la [Galería de nido de abeja](http://developer.android.com/resources/samples/HoneycombGallery/index.html) aplicación de ejemplo.

**Los temas de acción**

[android:actionButtonStyle](http://developer.android.com/reference/android/R.attr.html#actionButtonStyle)

Define un recurso de estilo para los botones de los elementos de acción.

[android:actionBarItemBackground](http://developer.android.com/reference/android/R.attr.html#actionBarItemBackground)

Define un recurso dibujable para el fondo de cada elemento de acción. (Añadido en el nivel de API 14.)

[android:itemBackground](http://developer.android.com/reference/android/R.attr.html#itemBackground)

Define un recurso dibujable para el fondo de cada elemento del menú de desbordamiento.

[android:actionBarDivider](http://developer.android.com/reference/android/R.attr.html#actionBarDivider)

Define un recurso dibujable de la división entre los elementos de acción. (Añadido en el nivel de API 14.)

[android:actionMenuTextColor](http://developer.android.com/reference/android/R.attr.html#actionMenuTextColor)

Define un color para el texto que aparece en un elemento de acción.

[android:actionMenuTextAppearance](http://developer.android.com/reference/android/R.attr.html#actionMenuTextAppearance)

Define un recurso de estilo para el texto que aparece en un elemento de acción.

[android:actionBarWidgetTheme](http://developer.android.com/reference/android/R.attr.html#actionBarWidgetTheme)

Define un recurso temático para los widgets que se inflan en la barra de acción como [puntos de vista de acción](http://developer.android.com/guide/topics/ui/actionbar.html#ActionView) . (Añadido en el nivel de API 14.)

**Fichas de navegación**

[android:actionBarTabStyle](http://developer.android.com/reference/android/R.attr.html#actionBarTabStyle)

Define un recurso de estilo para las pestañas en la barra de acción.

[android:actionBarTabBarStyle](http://developer.android.com/reference/android/R.attr.html#actionBarTabBarStyle)

Define un recurso de estilo de la barra delgada que aparece debajo de las pestañas de navegación.

[android:actionBarTabTextStyle](http://developer.android.com/reference/android/R.attr.html#actionBarTabTextStyle)

Define un recurso de estilo para el texto en las pestañas de navegación.

**Listas desplegables**

[android:actionDropDownStyle](http://developer.android.com/reference/android/R.attr.html#actionDropDownStyle)

Define un estilo para la navegación desplegable (por ejemplo, los antecedentes y estilos de texto).

Por ejemplo, aquí hay un archivo que define algunos estilos personalizados para la barra de acción:

<?xml version = "1.0" encoding = "UTF-8"?>

<resources>

<-! El tema aplicado a la aplicación o actividad ->

<Nombre del estilo = "CustomActivityTheme" padre = "@ android: estilo / tema.Holo ">

  <item name="android:actionBarTabTextStyle"> @ estilo / CustomTabTextStyle </ item>

  <item name="android:actionBarDivider"> @ dibujable / ab\_divider </ item>

<item name="android:actionBarItemBackground"> ​​@ dibujable / ab\_item\_background </ item>

</ Style>

<-! Estilo para el texto de la acción ficha de la barra ->

<Nombre del estilo = "CustomTabTextStyle" padre = "@ android: estilo / TextAppearance.Holo ">

<item name="android:textColor"> # 2456c2 </ item>

</style>

</ Recursos>

**Nota: Asegúrese de** que su tema se declara un tema principal en el <style> etiqueta, de la cual hereda todos los estilos no explícitamente declaradas por el tema. Al modificar la barra de acciones, usando un tema importante para los padres es que usted puede simplemente reemplazan los estilos que la barra de acciones que desea cambiar sin volver a aplicar los estilos que desea dejar en paz (por ejemplo, el aspecto del texto o material de relleno en los puntos de acción).

Usted puede aplicar el tema personalizado a toda la aplicación o para las actividades individuales en el archivo de manifiesto de esta manera:

<Aplicación para Android: tema = "@ estilo / CustomActivityTheme" ... />

Para obtener más información sobre el uso de recursos de estilo y el tema de la aplicación, lea [estilos y temas](http://developer.android.com/guide/topics/ui/themes.html) .

**Diseño avanzado**

Si usted necesita un estilo más avanzado de la barra de acción que está disponible con las propiedades anteriores, se puede incluir [android:actionBarStyle](http://developer.android.com/reference/android/R.attr.html#actionBarStyle) y [android:actionBarSplitStyle](http://developer.android.com/reference/android/R.attr.html#actionBarSplitStyle) en el tema de su actividad. Cada una de ellas especifica otro estilo que se pueden definir varias propiedades de la barra de acciones, incluyendo diferentes orígenes con [android:background](http://developer.android.com/reference/android/R.attr.html#background) , [android:backgroundSplit](http://developer.android.com/reference/android/R.attr.html#backgroundSplit) y [android:backgroundStacked](http://developer.android.com/reference/android/R.attr.html#backgroundStacked) . Si reemplaza los estilos de la barra de acciones, asegúrese de que se defina una acción de los padres bar de estilo, tales como [Widget.Holo.ActionBar](http://developer.android.com/reference/android/R.style.html#Widget_Holo_ActionBar) .

Por ejemplo, si usted quiere cambiar el fondo de la barra de acciones, puede utilizar los siguientes estilos:

<?xml version = "1.0" encoding = "UTF-8"?>

<resources>

<-! El tema aplicado a la aplicación o actividad ->

<Nombre del estilo = "CustomActivityTheme" padre = "@ android: estilo / tema.Holo ">

<item name="android:actionBarStyle"> @ estilo / MyActionBar </ item>

<-! Otra actividad y los estilos de barra de acción aquí ->

</ Style>

<-! Estilo de los fondos de la acción de barras ->

<Nombre del estilo = "MyActionBar" padre = "@ android: estilo / Widget.Holo.Actionbar ">

<item name="android:background"> ​​@ dibujable / ab\_background </ item>

<item name="android:backgroundStacked"> @ dibujable / ab\_background </ item>

<item name="android:backgroundSplit"> @ dibujable / ab\_split\_background </ item>

</style>

</ Recursos>

**Diálogos**

**En este documento**

1. [Mostrar un cuadro de diálogo](http://developer.android.com/guide/topics/ui/dialogs.html#ShowingADialog)
2. [Despedir a un diálogo](http://developer.android.com/guide/topics/ui/dialogs.html#DismissingADialog)
3. [Creación de un AlertDialog](http://developer.android.com/guide/topics/ui/dialogs.html#AlertDialog)
   1. [Agregar botones](http://developer.android.com/guide/topics/ui/dialogs.html#AddingButtons)
   2. [Adición de una lista](http://developer.android.com/guide/topics/ui/dialogs.html#AddingAList)
4. [Creación de un ProgressDialog](http://developer.android.com/guide/topics/ui/dialogs.html#ProgressDialog)
   1. [Mostrando una barra de progreso](http://developer.android.com/guide/topics/ui/dialogs.html#ShowingAProgressBar)
5. [Creación de un cuadro de diálogo personalizado](http://developer.android.com/guide/topics/ui/dialogs.html#CustomDialog)

**Clases de clave**

1. [Dialog](http://developer.android.com/reference/android/app/Dialog.html)
2. [AlertDialog](http://developer.android.com/reference/android/app/AlertDialog.html)
3. [DialogFragment](http://developer.android.com/reference/android/app/DialogFragment.html)

**Tutoriales relacionados**

1. [Hola DatePicker](http://developer.android.com/resources/tutorials/views/hello-datepicker.html)
2. [Hola TimePicker](http://developer.android.com/resources/tutorials/views/hello-timepicker.html)

**Véase también**

1. [Android Diseño: Cuadros de diálogo](http://developer.android.com/design/building-blocks/dialogs.html)

Un diálogo es por lo general una pequeña ventana que aparece al frente de la actividad actual. La actividad subyacente pierde el foco y el cuadro de diálogo acepta toda la interacción del usuario. Los diálogos se utilizan normalmente para las notificaciones que deben interrumpir el usuario y para realizar tareas cortas que se relacionan directamente con la aplicación en curso (por ejemplo, una barra de progreso o un inicio de sesión).

El [Dialog](http://developer.android.com/reference/android/app/Dialog.html) es la clase base para crear cuadros de diálogo. Sin embargo, normalmente no debe crear una instancia de [Dialog](http://developer.android.com/reference/android/app/Dialog.html) directamente. En su lugar, usted debe utilizar una de las siguientes subclases:

[AlertDialog](http://developer.android.com/reference/android/app/AlertDialog.html)

Un diálogo que puede manejar cero, uno, dos o tres botones, y / o una lista de elementos seleccionables que pueden incluir casillas o botones. El AlertDialog es capaz de construir interfaces de usuario más diálogo y es el tipo de diálogo propuesto. Consulte [Creación de un AlertDialog](http://developer.android.com/guide/topics/ui/dialogs.html#AlertDialog) a continuación.

[ProgressDialog](http://developer.android.com/reference/android/app/ProgressDialog.html)

Un diálogo que muestra una rueda de progreso o barra de progreso. Debido a que es una extensión de la AlertDialog, también es compatible con los botones. Consulte [Creación de un ProgressDialog](http://developer.android.com/guide/topics/ui/dialogs.html#ProgressDialog) a continuación.

[DatePickerDialog](http://developer.android.com/reference/android/app/DatePickerDialog.html)

Un cuadro de diálogo que permite al usuario seleccionar una fecha. Ver la [DatePicker Hola](http://developer.android.com/resources/tutorials/views/hello-datepicker.html) tutorial.

[TimePickerDialog](http://developer.android.com/reference/android/app/TimePickerDialog.html)

Un diálogo que permite al usuario seleccionar una hora. Ver la [TimePicker Hola](http://developer.android.com/resources/tutorials/views/hello-timepicker.html) tutorial.

Si usted desea personalizar su propio diálogo, puede ampliar la base [Dialog](http://developer.android.com/reference/android/app/Dialog.html) objeto o cualquiera de las subclases indicadas anteriormente y definir un nuevo diseño. Vea la sección de [Creación de un cuadro de diálogo personalizado](http://developer.android.com/guide/topics/ui/dialogs.html#CustomDialog) a continuación.

**Diseño de diálogo**

Para conocer las directrices de diseño, leer Diseño de Android [Diálogos](http://developer.android.com/design/building-blocks/dialogs.html) guía.

**Mostrar un cuadro de diálogo**

Un cuadro de diálogo siempre se crea y se muestra como parte de una [Activity](http://developer.android.com/reference/android/app/Activity.html) . Normalmente, debe crear diálogos dentro de su actividad de [onCreateDialog(int)](http://developer.android.com/reference/android/app/Activity.html#onCreateDialog%28int%29) el método de devolución de llamada. Cuando se utiliza esta devolución de llamada, el sistema Android gestiona automáticamente el estado de cada cuadro de diálogo y se engancha a la actividad, por lo que es efectivamente el "dueño" de cada diálogo. Como tal, cada cuadro de diálogo hereda ciertas propiedades de la actividad. Por ejemplo, cuando un cuadro de diálogo está abierto, la tecla Menú revela el menú de opciones definido para la actividad y las teclas de volumen modificar el flujo de audio utilizado por la actividad.

**Nota:** Si usted decide crear un diálogo fuera de la onCreateDialog() método, no se conectará a una actividad. Puede, sin embargo, se atribuyen a una actividad con [setOwnerActivity(Activity)](http://developer.android.com/reference/android/app/Dialog.html#setOwnerActivity%28android.app.Activity%29) .

Cuando se desea mostrar un cuadro de diálogo, llame a [showDialog(int)](http://developer.android.com/reference/android/app/Activity.html#showDialog%28int%29) y pasarle un entero que identifica de forma única el diálogo que se desea mostrar.

Cuando un cuadro de diálogo se solicita por primera vez, llama Android [onCreateDialog(int)](http://developer.android.com/reference/android/app/Activity.html#onCreateDialog%28int%29) de su actividad, que es donde se debe crear una instancia del [Dialog](http://developer.android.com/reference/android/app/Dialog.html) . Este método de devolución de llamada se pasa el mismo ID que pasó a [showDialog(int)](http://developer.android.com/reference/android/app/Activity.html#showDialog%28int%29) . Después de crear el cuadro de diálogo, devolver el objeto al final del método.

Antes de que el diálogo, Android también llama al método de devolución de llamada opcional [onPrepareDialog(int, Dialog)](http://developer.android.com/reference/android/app/Activity.html#onPrepareDialog%28int,%20android.app.Dialog%29) . Definir este método si desea cambiar las propiedades del cuadro de diálogo cada vez que se abre. Este método es llamado cada vez que se abre un cuadro de diálogo, mientras que [onCreateDialog(int)](http://developer.android.com/reference/android/app/Activity.html#onCreateDialog%28int%29) sólo se llama la primera vez que se abre un cuadro de diálogo. Si no se define [onPrepareDialog()](http://developer.android.com/reference/android/app/Activity.html#onPrepareDialog%28int,%20android.app.Dialog%29) , entonces el diálogo seguirá siendo la misma que la vez anterior que se abrió. Este método también se pasa el ID del diálogo, junto con el objeto de diálogo que ha creado en [onCreateDialog()](http://developer.android.com/reference/android/app/Activity.html#onCreateDialog%28int%29) .

La mejor manera de definir el [onCreateDialog(int)](http://developer.android.com/reference/android/app/Activity.html#onCreateDialog%28int%29) y [onPrepareDialog(int, Dialog)](http://developer.android.com/reference/android/app/Activity.html#onPrepareDialog%28int,%20android.app.Dialog%29) los métodos de devolución de llamada es con una sentencia *switch* que controla el parámetro de *identificación* que se pasa al método. Cada *caso* debe comprobar si hay una identificación única de diálogo y luego crear y definir el cuadro de diálogo correspondiente. Por ejemplo, imagina un juego que utiliza dos diálogos distintos: uno para indicar que el juego se ha detenido y otro para indicar que el juego ha terminado. En primer lugar, definir un identificador de número entero para cada cuadro de diálogo:

static final int DIALOG\_PAUSED\_ID = 0;

static final int DIALOG\_GAMEOVER\_ID = 1;

Luego, definir la [onCreateDialog(int)](http://developer.android.com/reference/android/app/Activity.html#onCreateDialog%28int%29) de devolución de llamada con un caso por cada interruptor ID:

protegida de diálogo onCreateDialog (int id) {

Diálogo de diálogo;

switch (id) {

caso DIALOG\_PAUSED\_ID:

/ / Hacer el trabajo para definir el cuadro de diálogo de pausa

[ break ; ]

caso DIALOG\_GAMEOVER\_ID:

/ / Hacer el trabajo para definir el juego a través de diálogo

[ break ; ]

default:

dialog = null;

}

volver de diálogo;

}

**Nota:** En este ejemplo, no hay código dentro de las declaraciones de casos debido a que el procedimiento para la definición de su diálogo está fuera del alcance de esta sección. Vea la sección siguiente sobre [Creación de un AlertDialog](http://developer.android.com/guide/topics/ui/dialogs.html#AlertDialog) , ofrece el código apropiado para este ejemplo.

Cuando sea el momento para mostrar uno de los cuadros de diálogo, llame a [showDialog(int)](http://developer.android.com/reference/android/app/Activity.html#showDialog%28int%29) con el ID de un cuadro de diálogo:

ShowDialog (DIALOG\_PAUSED\_ID);

**Despedir a un diálogo**

Cuando esté listo para cerrar el cuadro de diálogo, se puede despedir a llamando a [dismiss()](http://developer.android.com/reference/android/app/Dialog.html#dismiss%28%29) en el objeto de diálogo. Si es necesario, también puede llamar a [dismissDialog(int)](http://developer.android.com/reference/android/app/Activity.html#dismissDialog%28int%29) de la actividad, que de hecho llama a [dismiss()](http://developer.android.com/reference/android/app/Dialog.html#dismiss%28%29) en el cuadro de diálogo para usted.

Si está utilizando [onCreateDialog(int)](http://developer.android.com/reference/android/app/Activity.html#onCreateDialog%28int%29) para administrar el estado de sus diálogos (como se discutió en la sección anterior), entonces cada vez que se despidió de su diálogo, el estado del objeto de diálogo se mantiene por la actividad. Si usted decide que usted ya no tendrá este objeto o que es importante que el estado está desactivada, entonces usted debe llamar [removeDialog(int)](http://developer.android.com/reference/android/app/Activity.html#removeDialog%28int%29) . Esto eliminará todas las referencias internas con el objeto y si el diálogo se muestra, se lo desestime.

**Uso de despedir a los oyentes**

Si desea que su solicitud para realizar algunos procedimientos el momento en que un diálogo es despedido, entonces usted debe colocar un detector en despedir a su diálogo.

En primer lugar definir el [DialogInterface.OnDismissListener](http://developer.android.com/reference/android/content/DialogInterface.OnDismissListener.html) interfaz. Esta interfaz tiene sólo un método, [onDismiss(DialogInterface)](http://developer.android.com/reference/android/content/DialogInterface.OnDismissListener.html#onDismiss%28android.content.DialogInterface%29) , que se llamará cuando el diálogo se cierra. Después, simplemente pase su puesta en práctica de OnDismissListener [setOnDismissListener()](http://developer.android.com/reference/android/app/Dialog.html#setOnDismissListener%28android.content.DialogInterface.OnDismissListener%29) .

Sin embargo, tenga en cuenta que los cuadros de diálogo también puede ser "cancelado". Este es un caso especial que indica el diálogo se cancela explícitamente por el usuario. Esto ocurrirá si el usuario presiona el botón "atrás" para cerrar el cuadro de diálogo, o si el cuadro de diálogo de forma explícita llama a [cancel()](http://developer.android.com/reference/android/app/Dialog.html#cancel%28%29) (tal vez de un botón "Cancelar" en el diálogo). Cuando un cuadro de diálogo se cancela, el OnDismissListener todavía será notificado, pero si usted quisiera ser informado de que el diálogo fue cancelado de forma explícita (y no rechazó normalmente), entonces usted debe registrar una [DialogInterface.OnCancelListener](http://developer.android.com/reference/android/content/DialogInterface.OnCancelListener.html) con [setOnCancelListener()](http://developer.android.com/reference/android/app/Dialog.html#setOnCancelListener%28android.content.DialogInterface.OnCancelListener%29) .

**Creación de un AlertDialog**

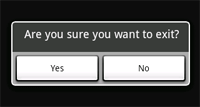
Un [AlertDialog](http://developer.android.com/reference/android/app/AlertDialog.html) es una extensión del [Dialog](http://developer.android.com/reference/android/app/Dialog.html) de clase. Es capaz de construir interfaces de usuario más diálogo y es el tipo de diálogo propuesto. Usted debe utilizar para los cuadros de diálogo que utilizan alguna de las siguientes características:

* a[title] {
* Un mensaje de texto
* Uno, dos, o tres botones
* Una lista de elementos seleccionables (con casillas o botones de radio opcional)

Para crear un AlertDialog, utilice el [AlertDialog.Builder](http://developer.android.com/reference/android/app/AlertDialog.Builder.html) subclase. Obtener un constructor con [AlertDialog.Builder(Context)](http://developer.android.com/reference/android/app/AlertDialog.Builder.html#AlertDialog.Builder%28android.content.Context%29) y luego usar los métodos públicos de la clase para definir todas las propiedades AlertDialog. Después de que hayas terminado con el Creador, recuperar el objeto AlertDialog con [create()](http://developer.android.com/reference/android/app/AlertDialog.Builder.html#create%28%29) .

Los temas siguientes se muestra cómo definir varias propiedades de la AlertDialog utilizando el AlertDialog.Constructor de la clase. Si utiliza cualquiera de las siguiente código de ejemplo dentro de su [onCreateDialog()](http://developer.android.com/reference/android/app/Activity.html#onCreateDialog%28int%29) el método de devolución de llamada, usted puede devolver el objeto resultante de diálogo para mostrar el cuadro de diálogo.

**Agregar botones**



Para crear un AlertDialog con los botones de lado a lado como el que se muestra en la imagen a la derecha, utilice el set...Button() métodos:

AlertDialog.Constructor constructor = new AlertDialog.Generador (este);

builder.setMessage ("¿Está seguro que desea salir?")

. SetCancelable (falso)

. SetPositiveButton ("Sí", DialogInterface nuevo.OnClickListener () {

public void onClick (DialogInterface de diálogo, int id) {

MyActivity.this.finish ();

}

})

. SetNegativeButton ("No", DialogInterface nuevo.OnClickListener () {

public void onClick (DialogInterface de diálogo, int id) {

dialog.cancel ();

}

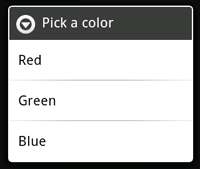
});

AlertDialog alerta builder.create = ();

En primer lugar, añadir un mensaje para el diálogo con [setMessage(CharSequence)](http://developer.android.com/reference/android/app/AlertDialog.Builder.html#setMessage%28java.lang.CharSequence%29) . A continuación, comenzará el método de encadenamiento y establecer el diálogo para *no* ser *cancelable* (por lo que el usuario no puede cerrar el diálogo con el botón de retroceso) con [setCancelable(boolean)](http://developer.android.com/reference/android/app/AlertDialog.Builder.html#setCancelable%28boolean%29) . Para cada botón, utilice uno de los set...Button() los métodos, tales como [setPositiveButton()](http://developer.android.com/reference/android/app/AlertDialog.Builder.html#setPositiveButton%28java.lang.CharSequence,%20android.content.DialogInterface.OnClickListener%29) , que acepta el nombre del botón y un [DialogInterface.OnClickListener](http://developer.android.com/reference/android/content/DialogInterface.OnClickListener.html) que define la acción a realizar cuando el usuario selecciona el botón.

**Nota:** Sólo se puede añadir uno de cada tipo de botón a la AlertDialog. Es decir, no se puede tener más de un resultado "positivo" el botón. Esto limita el número de botones posibles a tres: positivo, neutro y negativo. Estos nombres son técnicamente irrelevante para la funcionalidad real de los botones, sino que debe ayudar a no perder de vista que uno hace lo que pase.

**Adición de una lista**



Para crear un AlertDialog con una lista de elementos seleccionables como el que se muestra a la derecha, utilice los setItems() método:

CharSequence final [] = {elementos "Rojo", "Verde", "Azul"};

AlertDialog.Constructor constructor = new AlertDialog.Generador (este);

builder.setTitle ("Elija un color");

builder.setItems (artículos, DialogInterface nueva.OnClickListener () {

public void onClick (DialogInterface de diálogo, int elemento) {

Toast.makeText (getApplicationContext (), los artículos [tema], toast.. LENGTH\_SHORT) show ();

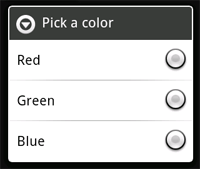
}

});

AlertDialog alerta builder.create = ();

En primer lugar, añadir un título al cuadro de diálogo con [setTitle(CharSequence)](http://developer.android.com/reference/android/app/AlertDialog.Builder.html#setTitle%28java.lang.CharSequence%29) . A continuación, agregue una lista de elementos seleccionables con [setItems()](http://developer.android.com/reference/android/app/AlertDialog.Builder.html#setItems%28java.lang.CharSequence%5B%5D,%20android.content.DialogInterface.OnClickListener%29) , que acepte el arreglo de elementos a mostrar y un [DialogInterface.OnClickListener](http://developer.android.com/reference/android/content/DialogInterface.OnClickListener.html) que define la acción a realizar cuando el usuario selecciona un elemento.

**Adición de casillas de verificación y botones de radio**



Para crear una lista de ítems de opción múltiple (casillas) o un solo ítems de opción (botones de radio) en el interior del cuadro de diálogo, utilice los [setMultiChoiceItems()](http://developer.android.com/reference/android/app/AlertDialog.Builder.html#setMultiChoiceItems%28android.database.Cursor,%20java.lang.String,%20java.lang.String,%20android.content.DialogInterface.OnMultiChoiceClickListener%29) y [setSingleChoiceItems()](http://developer.android.com/reference/android/app/AlertDialog.Builder.html#setSingleChoiceItems%28int,%20int,%20android.content.DialogInterface.OnClickListener%29) los métodos, respectivamente. Si se crea una de estas listas seleccionables en el [onCreateDialog()](http://developer.android.com/reference/android/app/Activity.html#onCreateDialog%28int%29) método de devolución de llamada, Android gestiona el estado de la lista para usted. Mientras que la actividad está activa, el diálogo recuerda los artículos que fueron seleccionados previamente, pero cuando el usuario sale de la actividad, la selección se ha perdido.

**Nota:** Para guardar la selección cuando el usuario se va o se detiene la actividad, que adecuadamente debe guardar y restaurar la configuración de todo el [ciclo de vida de la actividad](http://developer.android.com/guide/topics/fundamentals/activities.html#Lifecycle) . Para guardar de forma permanente las selecciones, incluso cuando el proceso de la actividad es completamente apagado, usted necesita para guardar los ajustes con una de las [de almacenamiento de datos](http://developer.android.com/guide/topics/data/data-storage.html) técnicas.

Para crear un AlertDialog con una sola lista de ítems de selección como la que aparece a la derecha, usar el mismo código del ejemplo anterior, pero sustituyendo los setItems() método con [setSingleChoiceItems()](http://developer.android.com/reference/android/app/AlertDialog.Builder.html#setSingleChoiceItems%28int,%20int,%20android.content.DialogInterface.OnClickListener%29) :

CharSequence final [] = {elementos "Rojo", "Verde", "Azul"};

AlertDialog.Constructor constructor = new AlertDialog.Generador (este);

builder.setTitle ("Elija un color");

builder.setSingleChoiceItems (artículos, DialogInterface -1, nueva.OnClickListener () {

public void onClick (DialogInterface de diálogo, int elemento) {

Toast.makeText (getApplicationContext (), los artículos [tema], toast LENGTH\_SHORT) show ();

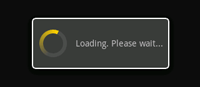
}

});

AlertDialog alerta builder.create = ();

El segundo parámetro en las [setSingleChoiceItems()](http://developer.android.com/reference/android/app/AlertDialog.Builder.html#setSingleChoiceItems%28java.lang.CharSequence%5B%5D,%20int,%20android.content.DialogInterface.OnClickListener%29) el método es un valor entero para el *checkedItem,* lo que indica la posición de la lista de base cero del elemento por defecto seleccionado. Use "-1" para indicar que no hay ningún elemento debe ser seleccionada por defecto.

**Creación de un ProgressDialog**



Un [ProgressDialog](http://developer.android.com/reference/android/app/ProgressDialog.html) es una extensión de la [AlertDialog](http://developer.android.com/reference/android/app/AlertDialog.html) clase que puede mostrar una animación de progreso en la forma de una rueda giratoria, para una tarea con el progreso que es indefinido, o una barra de progreso, para una tarea que tiene una progresión definida. El diálogo también puede proporcionar botones, como una de cancelar una descarga.

La apertura de un diálogo de progreso puede ser tan simple como llamar a [ProgressDialog.show()](http://developer.android.com/reference/android/app/ProgressDialog.html#show%28android.content.Context,%20java.lang.CharSequence,%20java.lang.CharSequence%29) . Por ejemplo, el diálogo de progreso muestra a la derecha se puede conseguir fácilmente sin lograr el diálogo a través de la [onCreateDialog(int)](http://developer.android.com/reference/android/app/Activity.html#onCreateDialog%28int%29) de devolución de llamada, como se muestra aquí:

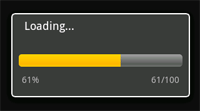
De diálogo ProgressDialog = ProgressDialog.show (MyActivity.this, "",

carga Por favor, espere ... ", true);

El primer parámetro es la aplicación [Context](http://developer.android.com/reference/android/content/Context.html) , el segundo es un título para el cuadro de diálogo (se deja vacío), el tercero es el mensaje, y el último parámetro es si el progreso es indeterminado (esto sólo es relevante cuando se crea una barra de progreso, que es discutido en la próxima sección).

El estilo por defecto de un diálogo de progreso es la rueda que gira. Si desea crear una barra de progreso que muestra el progreso de la carga con una granularidad, el código se requiere un poco más, como se explica en la siguiente sección.

**Mostrando una barra de progreso**



Para mostrar la progresión con una barra de progreso animada:

1. Inicializar el ProgressDialog con el constructor de la clase, [ProgressDialog(Context)](http://developer.android.com/reference/android/app/ProgressDialog.html#ProgressDialog%28android.content.Context%29) .
2. Defina el estilo de los avances a "STYLE\_HORIZONTAL" con [setProgressStyle(int)](http://developer.android.com/reference/android/app/ProgressDialog.html#setProgressStyle%28int%29) y establecer otras propiedades, tales como el mensaje.
3. Cuando esté listo para mostrar el cuadro de diálogo, llame a [show()](http://developer.android.com/reference/android/app/Dialog.html#show%28%29) o devolver el ProgressDialog de ​​la [onCreateDialog(int)](http://developer.android.com/reference/android/app/Activity.html#onCreateDialog%28int%29) de devolución de llamada.
4. Usted puede incrementar la cantidad de progreso que se muestra en la barra o bien llamando al [setProgress(int)](http://developer.android.com/reference/android/app/ProgressDialog.html#setProgress%28int%29) con un valor para el porcentaje total completado hasta ahora o [incrementProgressBy(int)](http://developer.android.com/reference/android/app/ProgressDialog.html#incrementProgressBy%28int%29) con un valor incremental de añadir al porcentaje total realizado hasta ahora.

Por ejemplo, la configuración podría tener este aspecto:

ProgressDialog progressDialog;

progressDialog = new ProgressDialog (mContext);

progressDialog.setProgressStyle (ProgressDialog.STYLE\_HORIZONTAL);

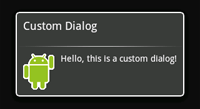
progressDialog.setMessage ("Cargando ...");

progressDialog.setCancelable (false);

La configuración es simple. La mayor parte del código necesario para crear un diálogo de progreso está realmente involucrado en el proceso que lo actualiza. Usted podría encontrar que es necesario crear un segundo hilo de la aplicación de este trabajo y luego informar de los avances de nuevo a la interfaz de usuario hilo de la actividad con un [Handler](http://developer.android.com/reference/android/os/Handler.html) objeto. Si usted no está familiarizado con el uso de las discusiones adicionales con un controlador, consulte el siguiente ejemplo de actividad que utiliza un segundo hilo para incrementar un diálogo de progreso logrado por la actividad.

[http://developer.android.com/assets/images/triangle-closed.png**Ejemplo ProgressDialog con un segundo hilo**](http://developer.android.com/guide/topics/ui/dialogs.html)

**Creación de un cuadro de diálogo personalizado**



Si desea un diseño personalizado para un cuadro de diálogo, puede crear su propio diseño de la ventana de diálogo con el diseño y los elementos de widgets. Después de haber definido el diseño, pase el objeto View raíz o identificador de recursos de diseño para [setContentView(View)](http://developer.android.com/reference/android/app/Dialog.html#setContentView%28android.view.View%29) .

Por ejemplo, para crear el diálogo que se muestra a la derecha:

1. Crear un diseño de XML guardado como custom\_dialog.xml :
2. <LinearLayout xmlns: android = "http://schemas.android.com/apk/res/android"
3. android: id = "+ id @ / layout\_root"
4. android: orientation = "horizontal"
5. android: layout\_width = "fill\_parent"
6. android: layout\_height = "fill\_parent"
7. Android: padding = "10DP"
8. >
9. <ImageView android: id = "+ id @ / imagen"
10. android: layout\_width = "wrap\_content"
11. android: layout\_height = "fill\_parent"
12. android: layout\_marginRight = "10DP"
13. />
14. <TextView android: id = "+ id @ / texto"
15. android: layout\_width = "wrap\_content"
16. android: layout\_height = "fill\_parent"
17. android: textColor = "# FFF"
18. />
19. </ LinearLayout>

Este XML define un [ImageView](http://developer.android.com/reference/android/widget/ImageView.html) y un [TextView](http://developer.android.com/reference/android/widget/TextView.html) dentro de un [LinearLayout](http://developer.android.com/reference/android/widget/LinearLayout.html) .

1. Establezca el diseño por encima de vista de contenido como el diálogo y definir el contenido de la ImageView y elementos TextView:

Contexto mContext getApplicationContext = ();

De diálogo de diálogo = nuevo cuadro de diálogo (mContext);

dialog.setContentView (R.layout.custom\_dialog);

dialog.setTitle ("cuadro de diálogo personalizado");

Vista de Texto text = (TextView) dialog.findViewById (R.id.text);

text.setText ("Hola, este es un diálogo personalizado!");

ImageView la imagen = (ImageView) dialog.findViewById (R.id.image);

image.setImageResource (R.drawable.android);

Después de crear una instancia del diálogo, poner tu diseño personalizado, como ver el contenido del cuadro de diálogo con [setContentView(int)](http://developer.android.com/reference/android/app/Dialog.html#setContentView%28int%29) , pasándole el identificador de recursos de diseño. Ahora que el diálogo tiene un diseño definido, puede capturar objetos Vista desde el diseño con [findViewById(int)](http://developer.android.com/reference/android/app/Dialog.html#findViewById%28int%29) y modificar su contenido.

 Eso es todo. Ahora puede mostrar el diálogo como se describe en [un cuadro de diálogo Mostrando](http://developer.android.com/guide/topics/ui/dialogs.html#ShowingADialog) .

Un diálogo hecho con la clase de diálogo de base debe tener un título. Si usted no llama [setTitle()](http://developer.android.com/reference/android/app/Dialog.html#setTitle%28java.lang.CharSequence%29) , entonces el espacio utilizado por el título se queda vacío, pero aún visible. Si usted no desea un título para nada, entonces usted debe crear su diálogo personalizado utilizando el [AlertDialog](http://developer.android.com/reference/android/app/AlertDialog.html) clase. Sin embargo, debido a que un AlertDialog es más fácil de crear con la [AlertDialog.Builder](http://developer.android.com/reference/android/app/AlertDialog.Builder.html) la clase, usted no tiene acceso a la [setContentView(int)](http://developer.android.com/reference/android/app/Dialog.html#setContentView%28int%29) método utilizado anteriormente. En su lugar, debe utilizar [setView(View)](http://developer.android.com/reference/android/app/AlertDialog.Builder.html#setView%28android.view.View%29) . Este método acepta un [View](http://developer.android.com/reference/android/view/View.html) objeto, por lo que necesita para inflar el objeto de diseño raíz de Vista a partir de XML.

Para inflar el esquema XML, recuperar el [LayoutInflater](http://developer.android.com/reference/android/view/LayoutInflater.html) con [getLayoutInflater()](http://developer.android.com/reference/android/app/Activity.html#getLayoutInflater%28%29) (o [getSystemService()](http://developer.android.com/reference/android/content/Context.html#getSystemService%28java.lang.String%29) ), y luego llamar a [inflate(int, ViewGroup)](http://developer.android.com/reference/android/view/LayoutInflater.html#inflate%28int,%20android.view.ViewGroup%29) , donde el primer parámetro es el identificador de recursos de diseño y el segundo es el ID de la vista de la raíz. En este punto, puede utilizar el diseño de inflado para encontrar objetos Ver en el diseño y definir el contenido de la ImageView y elementos TextView. A continuación, una instancia de la AlertDialog.Builder y configurar el tamaño de inflado para el diálogo con [setView(View)](http://developer.android.com/reference/android/app/AlertDialog.Builder.html#setView%28android.view.View%29) .

He aquí un ejemplo, la creación de un diseño personalizado en un AlertDialog:

AlertDialog.Generador de constructor;

AlertDialog alertDialog;

Contexto mContext getApplicationContext = ();

LayoutInflater inflater = (LayoutInflater) mContext.getSystemService (LAYOUT\_INFLATER\_SERVICE);

Ver layout = inflater.inflate (R.layout.custom\_dialog,

(ViewGroup) findViewById (R.id.layout\_root));

Vista de Texto text = (TextView) layout.findViewById (R.id.text);

text.setText ("Hola, este es un diálogo personalizado!");

ImageView la imagen = (ImageView) layout.findViewById (R.id.image);

image.setImageResource (R.drawable.android);

constructor = new AlertDialog.Builder (mContext);

builder.setView (diseño);

alertDialog builder.create = ();

El uso de un AlertDialog para su diseño personalizado le permite tomar ventaja de las características incorporadas de AlertDialog como botones, listas seleccionables gestionados, un título, un icono y así sucesivamente.

Para obtener más información, consulte la documentación de referencia para el [Dialog](http://developer.android.com/reference/android/app/Dialog.html) y la [AlertDialog.Builder](http://developer.android.com/reference/android/app/AlertDialog.Builder.html) clases.

**Notificaciones**

Existen varios tipos de situaciones puede ocurrir que requieren que se notifique al usuario acerca de un evento que ocurre en su aplicación. Algunos eventos requieren que el usuario responda y otros no. Por ejemplo:

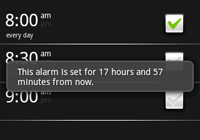
* Cuando un evento como guardar un archivo está completo, un pequeño mensaje aparecerá para confirmar que las de ahorro se ha realizado correctamente.
* Si la aplicación se ejecuta en segundo plano y necesita la atención del usuario, la aplicación debe crear una notificación de que permite que el usuario responda a su conveniencia.
* Si la aplicación está realizando el trabajo que el usuario debe esperar a que (como la carga de un archivo), la aplicación debe mostrar el progreso se cierne una rueda o un bar.

Cada una de estas tareas de notificación se puede lograr utilizando una técnica diferente:

* Una [notificación de Toast](http://developer.android.com/guide/topics/ui/notifiers/index.html#Toast) , para los mensajes breves que provienen del fondo.
* Una [notificación de la barra de estado](http://developer.android.com/guide/topics/ui/notifiers/index.html#StatusBar) , para los recordatorios persistentes que provienen de los antecedentes y solicitar la respuesta del usuario.
* Una [Notificación de diálogo](http://developer.android.com/guide/topics/ui/notifiers/index.html#Dialog) , de las notificaciones relacionadas con la actividad.

Este documento resume cada una de estas técnicas para notificar al usuario e incluye enlaces a la documentación completa.

**Brindis de notificación**

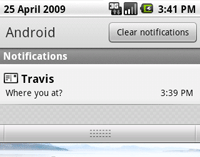


Una notificación brindis es un mensaje que aparece en la superficie de la ventana. Sólo llena el espacio requerido para el mensaje y la actividad actual del usuario permanece visible e interactivo. La notificación automáticamente se desvanece dentro y fuera, y no aceptan eventos de interacción. Debido a que un brindis puede ser creado a partir de un fondo [Service](http://developer.android.com/reference/android/app/Service.html) , al parecer, incluso si la aplicación no es visible.

Un brindis por que es mejor para mensajes de texto cortos, como "archivo guardado," cuando estás bastante seguro de que el usuario está prestando atención a la pantalla. Un brindis no puede aceptar eventos de interacción del usuario, si lo desea el usuario para responder y tomar medidas, considere usar una [notificación de la barra de estado](http://developer.android.com/guide/topics/ui/notifiers/index.html#StatusBar) en su lugar.

Para obtener más información, consulte [Notificaciones de Toast](http://developer.android.com/guide/topics/ui/notifiers/toasts.html) .

**Notificación de estado de Abogados**

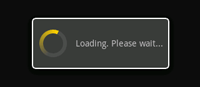


La notificación de la barra de estado añade un icono a la barra de estado del sistema (con la opción de clave de pizarra-el texto del mensaje) y un mensaje más amplio en el "Notificaciones" de la ventana. Cuando el usuario selecciona el mensaje expandido, Android se dispara una [Intent](http://developer.android.com/reference/android/content/Intent.html) que se define por la notificación (por lo general para poner en marcha una [Activity](http://developer.android.com/reference/android/app/Activity.html) ). También puede configurar la notificación para alertar al usuario con un sonido, una vibración y luces intermitentes en el dispositivo.

Este tipo de notificación es ideal cuando la aplicación está trabajando en un fondo [Service](http://developer.android.com/reference/android/app/Service.html) y debe notificar al usuario acerca de un evento. Si usted necesita para alertar al usuario acerca de un evento que se produce, mientras que su actividad se encuentra todavía en el enfoque, considere el uso de una [Notificación de diálogo](http://developer.android.com/guide/topics/ui/notifiers/index.html#Dialog) en su lugar.

Para obtener más información, consulte [Notificaciones de barra de estado](http://developer.android.com/guide/topics/ui/notifiers/notifications.html) .

**Notificación de diálogo**



Un diálogo es por lo general una pequeña ventana que aparece al frente de la actividad actual. La actividad subyacente pierde el foco y el cuadro de diálogo acepta toda la interacción del usuario. Los diálogos se utilizan normalmente para las notificaciones y las actividades de cortos que se relacionan directamente con la aplicación en curso.

Usted debe utilizar un cuadro de diálogo cuando se necesita mostrar una barra de progreso o un mensaje corto que requiere la confirmación del usuario (por ejemplo, una alerta con "OK" y los botones "Cancelar"). Puede utilizar también utilizan los cuadros de diálogo como parte integrante de la interfaz de usuario de su aplicación y para otros fines, además de las notificaciones. Para una discusión completa sobre todos los tipos disponibles de los cuadros de diálogo, incluyendo sus usos para las notificaciones, se refieren a [cuadros de diálogo](http://developer.android.com/guide/topics/ui/dialogs.html) .

**Notificaciones de Toast**

**Vista Rapida**

1. Un brindis es un mensaje que aparece en la superficie de la pantalla por un momento, pero que no tiene el enfoque (o una pausa en la actividad actual), por lo que no puede aceptar la entrada del usuario
2. Usted puede personalizar el diseño de una tostada para incluir imágenes

**En este documento**

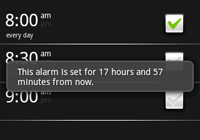
1. [Los puntos básicos.](http://developer.android.com/guide/topics/ui/notifiers/toasts.html#Basics)
2. [Colocación de su brindis](http://developer.android.com/guide/topics/ui/notifiers/toasts.html#Positioning)
3. [Creación de una vista personalizada Toast](http://developer.android.com/guide/topics/ui/notifiers/toasts.html#CustomToastView)

**Clases de clave**

1. [Toast](http://developer.android.com/reference/android/widget/Toast.html)

Una notificación brindis es un mensaje que aparece en la superficie de la ventana. Sólo llena el espacio requerido para el mensaje y la actividad actual del usuario permanece visible e interactivo. La notificación automáticamente se desvanece dentro y fuera, y no aceptan eventos de interacción.

La siguiente imagen muestra una notificación de tostadas ejemplo de la aplicación de alarma. Una vez que la alarma está activada, se muestra un brindis al asegurar que la alarma se creó.



Un brindis se puede crear y se muestran a partir de una [Activity](http://developer.android.com/reference/android/app/Activity.html) o [Service](http://developer.android.com/reference/android/app/Service.html) . Si se crea una notificación de una tostada de un servicio, que aparece en frente de la actividad en la actualidad en el foco.

Si la respuesta del usuario a la notificación se requiere considerar el uso de una [Notificación de la barra de estado](http://developer.android.com/guide/topics/ui/notifiers/notifications.html) .

**Los puntos básicos.**

En primer lugar, una instancia de un [Toast](http://developer.android.com/reference/android/widget/Toast.html) objeto con uno de los [makeText()](http://developer.android.com/reference/android/widget/Toast.html#makeText%28android.content.Context,%20int,%20int%29) métodos. Este método tiene tres parámetros: la aplicación [Context](http://developer.android.com/reference/android/content/Context.html) , el mensaje de texto, y la duración para el brindis. Devuelve un objeto de Toast inicializado correctamente. Puede mostrar la notificación brindis con [show()](http://developer.android.com/reference/android/widget/Toast.html#show%28%29) , como se muestra en el siguiente ejemplo:

Context = getApplicationContext ();

Texto CharSequence = "Hola brindis!";

int = duración Toast.LENGTH\_SHORT;

Toast tostadas = Toast.makeText (contexto, texto, duración);

toast.show ();

Este ejemplo demuestra todo lo necesario para la mayoría de las notificaciones de toast. Rara vez se necesita nada más. Usted puede, sin embargo, desea colocar el brindis de forma diferente o incluso utilizar su propio diseño en lugar de un simple mensaje de texto. Las secciones siguientes describen cómo puede hacer estas cosas.

También se pueden encadenar sus métodos y evitar aferrarse al objeto de Toast, así:

. Toast.makeText (contexto, texto, duración) show ();

**Colocación de su brindis**

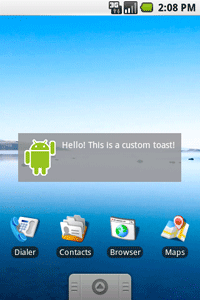
Una notificación tostadas normal aparece en la parte inferior de la pantalla, centrado horizontalmente. Puede cambiar esta posición con el [setGravity(int, int, int)](http://developer.android.com/reference/android/widget/Toast.html#setGravity%28int,%20int,%20int%29) método. Este acepta tres parámetros: una [Gravity](http://developer.android.com/reference/android/view/Gravity.html) constante, un x-posición de desplazamiento, y una posición y desplazamiento.

Por ejemplo, si usted decide que el brindis debe aparecer en la esquina superior izquierda, se puede establecer la gravedad de esta manera:

toast.setGravity (Gravity.TOP | gravedad.Izquierda, 0, 0);

Si usted quiere empujar a la posición a la derecha, aumente el valor del segundo parámetro. Para empujar hacia abajo, aumentar el valor del último parámetro.

**Creación de una vista personalizada Toast**



Si un mensaje de texto simple no es suficiente, puede crear un diseño personalizado para su notificación de toast. Para crear un diseño personalizado, definir un diseño de Vista, en XML o en el código de aplicación, y aprobar la raíz [View](http://developer.android.com/reference/android/view/View.html) objeto a la [setView(View)](http://developer.android.com/reference/android/widget/Toast.html#setView%28android.view.View%29) método.

Por ejemplo, puede crear el diseño para el brindis visible en la pantalla a la derecha con el siguiente código XML (guardado como *toast\_layout.xml):*

<LinearLayout xmlns: android = "http://schemas.android.com/apk/res/android"

android: id = "+ id @ / toast\_layout\_root"

android: orientation = "horizontal"

android: layout\_width = "fill\_parent"

android: layout\_height = "fill\_parent"

Android: padding = "10DP"

android: background = "# DAAA"

>

<ImageView android: id = "+ id @ / imagen"

android: layout\_width = "wrap\_content"

android: layout\_height = "fill\_parent"

android: layout\_marginRight = "10DP"

/>

<TextView android: id = "+ id @ / texto"

android: layout\_width = "wrap\_content"

android: layout\_height = "fill\_parent"

android: textColor = "# FFF"

/>

</ LinearLayout>

Observe que el ID del elemento LinearLayout es "toast\_layout". Debe utilizar este ID para inflar el diseño del XML, como se muestra aquí:

LayoutInflater inflater getLayoutInflater = ();

Ver layout = inflater.inflate (R.layout.toast\_layout,

(ViewGroup) findViewById (R.id.toast\_layout\_root));

ImageView la imagen = (ImageView) layout.findViewById (R.id.image);

image.setImageResource (R.drawable.android);

Vista de Texto text = (TextView) layout.findViewById (R.id.text);

text.setText ("¡Hola! Este es un brindis personalizado! ");

Toast tostadas = new Toast (getApplicationContext ());

toast.setGravity (Gravity.CENTER\_VERTICAL, 0, 0);

toast.setDuration (Toast.LENGTH\_LONG);

toast.setView (diseño);

toast.show ();

En primer lugar, recuperar el [LayoutInflater](http://developer.android.com/reference/android/view/LayoutInflater.html) con [getLayoutInflater()](http://developer.android.com/reference/android/app/Activity.html#getLayoutInflater%28%29) (o [getSystemService()](http://developer.android.com/reference/android/content/Context.html#getSystemService%28java.lang.String%29) ), y luego inflar el diseño de XML usando [inflate(int, ViewGroup)](http://developer.android.com/reference/android/view/LayoutInflater.html#inflate%28int,%20android.view.ViewGroup%29) . El primer parámetro es el identificador de recursos de diseño y el segundo es la vista de la raíz. Usted puede utilizar este diseño de inflado para encontrar más objetos en el diseño de Vista, por lo que ahora capturar y definir el contenido de la ImageView y elementos TextView. Por último, crear un nuevo Toast con [Toast(Context)](http://developer.android.com/reference/android/widget/Toast.html#Toast%28android.content.Context%29) y establecer algunas propiedades de la tostada, como la gravedad y duración. A continuación, llame [setView(View)](http://developer.android.com/reference/android/widget/Toast.html#setView%28android.view.View%29) y pasarle el diseño de inflado. Ahora puede mostrar el brindis con su diseño personalizado llamando a [show()](http://developer.android.com/reference/android/widget/Toast.html#show%28%29) .

**Nota:** No utilice el constructor público para hacer un brindis a menos que usted se va a definir el diseño con [setView(View)](http://developer.android.com/reference/android/widget/Toast.html#setView%28android.view.View%29) . Si usted no tiene un diseño personalizado que se utilizará, debe utilizar [makeText(Context, int, int)](http://developer.android.com/reference/android/widget/Toast.html#makeText%28android.content.Context,%20int,%20int%29) para crear el toast>

**Notificaciones barra de estado**

**Vista Rapida**

* La notificación de la barra de estado permite a su aplicación para notificar al usuario de un evento sin interrumpir su actividad actual
* Puede adjuntar la intención de su notificación de que el sistema se iniciará cuando el usuario hace clic en

**En este documento**

1. [Los puntos básicos.](http://developer.android.com/guide/topics/ui/notifiers/notifications.html#Basics)
2. [En respuesta a las notificaciones](http://developer.android.com/guide/topics/ui/notifiers/notifications.html#HandlingNotifications)
3. [Gestión de las notificaciones](http://developer.android.com/guide/topics/ui/notifiers/notifications.html#ManageYourNotifications)
4. [Creación de una notificación](http://developer.android.com/guide/topics/ui/notifiers/notifications.html#CreateANotification)
   1. [Actualización de la notificación](http://developer.android.com/guide/topics/ui/notifiers/notifications.html#Updating)
   2. [Adición de un sonido](http://developer.android.com/guide/topics/ui/notifiers/notifications.html#Sound)
   3. [Adición de las vibraciones](http://developer.android.com/guide/topics/ui/notifiers/notifications.html#Vibration)
   4. [Adición de luces intermitentes](http://developer.android.com/guide/topics/ui/notifiers/notifications.html#Lights)
   5. [Más características](http://developer.android.com/guide/topics/ui/notifiers/notifications.html#More)
5. [Crear un diseño personalizado de notificación](http://developer.android.com/guide/topics/ui/notifiers/notifications.html#CustomExpandedView)

**Clases de clave**

1. [Notification](http://developer.android.com/reference/android/app/Notification.html)
2. [NotificationManager](http://developer.android.com/reference/android/app/NotificationManager.html)

**Véase también**

1. [Android Diseño: Notificaciones](http://developer.android.com/design/patterns/notifications.html)

La notificación de la barra de estado añade un icono a la barra de estado del sistema (con la opción de clave de pizarra-el texto del mensaje) y un mensaje de notificación en la ventana de notificaciones. Cuando el usuario selecciona la notificación, Android se dispara una [Intent](http://developer.android.com/reference/android/content/Intent.html) que se define por la [Notification](http://developer.android.com/reference/android/app/Notification.html) (por lo general para poner en marcha una [Activity](http://developer.android.com/reference/android/app/Activity.html) ). También puede configurar la notificación para alertar al usuario con un sonido, una vibración y luces intermitentes en el dispositivo.

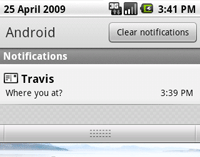
La notificación de la barra de estado debe ser utilizado para cualquier caso en que un servicio de fondo tiene que advertir al usuario acerca de un evento que requiere una respuesta. Un servicio en segundo plano **nunca se debe poner en** marcha una actividad por cuenta propia con el fin de la interacción del usuario. El servicio en lugar debería crear una notificación de la barra de estado que pondrá en marcha la actividad cuando sea seleccionado por el usuario.

La Figura 1 muestra la barra de estado con un icono de notificación en el lado izquierdo.

http://developer.android.com/images/status_bar.png

**FIGURA I.** Barra de estado con una notificación.

La Figura 2 muestra el mensaje de la notificación en la ventana de notificaciones.



**FIGURA 2.** La ventana de notificaciones.

**Notificación de diseño**

Para conocer las directrices de diseño, lea Android Diseño de [Notificaciones](http://developer.android.com/design/patterns/notifications.html) guía.

**Los puntos básicos.**

Una [Activity](http://developer.android.com/reference/android/app/Activity.html) o [Service](http://developer.android.com/reference/android/app/Service.html) puede iniciar una notificación de la barra de estado. Debido a que una actividad puede llevar a cabo acciones sólo mientras se está ejecutando en primer plano y su ventana tiene el foco, por lo general se crean las notificaciones de estado de un servicio de bar. De esta manera, la notificación se puede crear desde el fondo, mientras el usuario está usando otra aplicación o mientras el dispositivo está dormido. Para crear una notificación, debe utilizar dos clases: [Notification](http://developer.android.com/reference/android/app/Notification.html) y [NotificationManager](http://developer.android.com/reference/android/app/NotificationManager.html) .

Utilice una instancia de la [Notification](http://developer.android.com/reference/android/app/Notification.html) clase para definir las propiedades de la notificación de la barra de estado, como el icono de la barra de estado, el mensaje de notificación, y los ajustes adicionales, tales como un sonido para reproducir. El [NotificationManager](http://developer.android.com/reference/android/app/NotificationManager.html) es un servicio del sistema Android que realiza y gestiona todas las notificaciones de la barra de estado. No es una instancia de la [NotificationManager](http://developer.android.com/reference/android/app/NotificationManager.html) directamente. Con el fin de darle a su [Notification](http://developer.android.com/reference/android/app/Notification.html) , debe recuperar una referencia a la [NotificationManager](http://developer.android.com/reference/android/app/NotificationManager.html) con [getSystemService()](http://developer.android.com/reference/android/app/Activity.html#getSystemService%28java.lang.String%29) y luego, cuando se quiere notificar al usuario, que pasan a su [Notification](http://developer.android.com/reference/android/app/Notification.html) a [notify()](http://developer.android.com/reference/android/app/NotificationManager.html#notify%28int,%20android.app.Notification%29) .

Para crear una notificación de la barra de estado:

1. Obtener una referencia a la [NotificationManager](http://developer.android.com/reference/android/app/NotificationManager.html) :
2. Cadena ns = Contexto.NOTIFICATION\_SERVICE;
3. MNotificationManager NotificationManager = (NotificationManager) getSystemService (ns);
4. Una instancia de la [Notification](http://developer.android.com/reference/android/app/Notification.html) :
5. int icon = R.drawable.notification\_icon;
6. CharSequence tickerText = "Hola";
7. de largo cuando = System.currentTimeMillis ();
8. Notificación de la notificación = nueva notificación (icono, tickerText, cuándo);
9. Definir mensaje de la notificación y [PendingIntent](http://developer.android.com/reference/android/app/PendingIntent.html) :
10. Context = getApplicationContext ();
11. CharSequence contentTitle = "Mi notificación";
12. CharSequence contentText = "Hello World!";
13. NotificationIntent Intención = nuevo intento (esto, MyClass.class);
14. ContentIntent PendingIntent = PendingIntent.getActivity (este, 0, notificationIntent, 0);
15. notification.setLatestEventInfo (contexto, contentTitle, contentText, contentIntent);
16. Pasar la [Notification](http://developer.android.com/reference/android/app/Notification.html) a la [NotificationManager](http://developer.android.com/reference/android/app/NotificationManager.html) :
17. private static final int HELLO\_ID = 1;
18. mNotificationManager.notify (HELLO\_ID, la notificación);

Eso es todo. El usuario ha sido notificado.

**En respuesta a las notificaciones**

Una parte central de la experiencia del usuario con una notificación gira en torno a la forma en que interactúa con el flujo de la interfaz de usuario de la aplicación. Debe implementar este correctamente para proporcionar una experiencia de usuario coherente en su aplicación.

Dos ejemplos típicos de las notificaciones se proporcionan por el calendario, que puede enviar notificaciones de eventos, y correo electrónico, que puede enviar notificaciones cuando llegan nuevos mensajes. Estos representan los dos modelos recomendados para el manejo de las notificaciones: o lanzarse a una actividad que es independiente de la aplicación principal, o el lanzamiento de una instancia completamente nueva de la aplicación que muestra el punto adecuado para la notificación.

El escenario siguiente muestra cómo la pila de la actividad deben trabajar en estos dos flujos típicos de notificación, en primer lugar el manejo de una notificación de calendario:

1. El usuario está creando un nuevo evento en el Calendario. Se dan cuenta que necesitan para copiar parte de un mensaje de correo electrónico en este evento.
2. El usuario selecciona Inicio> Correo electrónico.
3. Mientras que en el correo electrónico, recibirá una notificación de calendario para una próxima reunión.
4. Así que elige la notificación, que los lleva a una actividad dedicada Calendario que muestra breves detalles de la próxima reunión.
5. El usuario ha visto suficiente para saber que tienen una próxima reunión para arriba, así que pulse el botón *Atrás.* Ahora se volvió a correo electrónico, que es donde estaban cuando se tomó la notificación.

Manejo de una notificación por correo electrónico:

1. El usuario se encuentra actualmente en redactar un mensaje de correo electrónico, y tiene que ver una fecha en su calendario.
2. El usuario selecciona Inicio> Calendario.
3. Mientras que en el calendario, que recibirá una notificación de correo electrónico acerca de un nuevo mensaje.
4. Se selecciona la notificación, que los lleva a un correo electrónico con los detalles del mensaje se muestra. Este ha sustituido a lo que estaban haciendo previamente (escribiendo un e-mail), pero el mensaje todavía está guardado en sus proyectos.
5. El usuario presiona *Back* una vez para ir a la lista de mensajes (el flujo típico en la aplicación de correo electrónico), y pulse *Atrás* para volver al calendario como lo dejaron.

En un estilo de correo electrónico de notificación, la interfaz de usuario puesto en marcha por el aviso muestra que la principal aplicación en un estado que representa a la notificación. Por ejemplo, cuando la aplicación de correo electrónico pasa a primer plano a partir de su notificación, o bien muestra la lista de conversión o de una conversación específica en función de si hay varios o sólo un nuevo correo electrónico. Para lograr esto, queremos sustituir por completo lo que sea actual estado de la aplicación con una pila nueva actividad que representa el estado nueva notificación.

El siguiente código muestra cómo mostrar este tipo de notificación. De mayor interés es el makeMessageIntentStack() método, que construye una serie de intentos que representan a la pila de la aplicación de la nueva actividad de este estado. (Si está utilizando fragmentos, puede que tenga que inicializar el fragmento y el estado de aplicación de modo que si se pulsa *Back* cambiará la interfaz de usuario de nuevo a su estado primario.) El núcleo de este es el [Intent.makeRestartActivityTask()](http://developer.android.com/reference/android/content/Intent.html#makeRestartActivityTask%28android.content.ComponentName%29) método, que construye la actividad de las raíces de la pila con las banderas apropiadas, tales como [Intent.FLAG\_ACTIVITY\_CLEAR\_TASK](http://developer.android.com/reference/android/content/Intent.html#FLAG_ACTIVITY_CLEAR_TASK) .

/ \*\*

\* Este método crea una matriz de objetos que representan a la Intención

\* La actividad de la pila para el mensaje entrante que el estado de los detalles

\* La aplicación debe estar en marcha cuando a partir de una notificación.

\* /

Intención estática [] makeMessageIntentStack (contexto contexto, a partir CharSequence,

CharSequence msg) {

/ / Una convención típica para las notificaciones es poner en marcha el usuario profunda

/ / En una aplicación que representa los datos en la notificación; a

/ / Lograr esto, podemos construir una serie de intentos para insertar la parte posterior

/ / Pila pila de la historia por encima del elemento que se muestra.

Intención [] = intentos nuevo intento [4];

/ / En primer lugar: la actividad raíz de ApiDemos.

/ / Esta es una manera conveniente de hacer la intención apropiada para poner en marcha y

/ / Restablecer la tarea de la aplicación.

intentos [0] = Intent.makeRestartActivityTask (nuevo ComponentName (contexto,

com.example.android.apis.ApiDemos.class));

// App

intentos [1] = nuevo intento (el contexto, com.example.android.apis.ApiDemos.class);

intentos [1]. putExtra ("com.example.android.apis.Path "," App ");

/ / "App / Notificación"

intentos [2] = nuevo intento (el contexto, com.example.android.apis.ApiDemos.class);

intentos [2]. putExtra ("com.example.android.apis.Path "," App / Notificación ");

/ / Ahora la actividad para mostrar al usuario. También rellenar los datos que

/ / Debe mostrar.

intentos [3] = nuevo intento (contexto, IncomingMessageView.class);

intentos [3]. putExtra (IncomingMessageView.KEY\_FROM, desde);

intentos [3]. putExtra (IncomingMessageView.KEY\_MESSAGE, msg);

volver intenciones;

}

/ \*\*

\* La notificación es el icono y la entrada ampliada asociado en el

Barra de estado

\* /

showAppNotification vacío () {

/ / Buscar el servicio de gestor de notificaciones

Nm NotificationManager = (NotificationManager) getSystemService (NOTIFICATION\_SERVICE);

/ / Los detalles de nuestro mensaje falso

CharSequence from = "Joe";

CharSequence mensaje;

interruptor ((new Random (). nextInt ())% 3) {

caso 0: mensaje = "ru hambre? Estoy muerto de hambre "; break;

case 1: message = "u im cercano"; break;

default: mensaje = ".. kthx meet u cul8r para la cena"; break;

}

/ / El PendingIntent para poner en marcha nuestra actividad, si el usuario selecciona esta

notificación Nótese el uso de FLAG\_CANCEL\_CURRENT de modo que, si hay

/ / Ya es un intento de adaptación activa en espera, que anula y sustituye

/ / Con la nueva matriz de Intenciones.

PendingIntent contentIntent = PendingIntent.getActivities (este, 0,

makeMessageIntentStack (esto, de, mensaje), PendingIntent.FLAG\_CANCEL\_CURRENT);

/ / El texto clave de pizarra, se utiliza una cadena con formato por lo que nuestro mensaje pueda ser localizado

Cadena tickerText = getString (R.string.imcoming\_message\_ticker\_text, mensaje);

/ / Construir el objeto de notificación.

Notificación entr = nueva notificación (R.drawable.stat\_sample, tickerText,

System.currentTimeMillis ());

/ / Establecer la información de las opiniones que se muestran en el panel de notificación.

notif.setLatestEventInfo (esto, de, mensaje, contentIntent);

/ / Vamos a hacer que esta notificación el sonido por defecto, la vibración y led.

/ / Tenga en cuenta que si usted desea alguno de estos comportamientos, que siempre debe tener

/ / Una preferencia por el usuario para desactivarlas.

notif.defaults = Notificación.DEFAULT\_ALL;

/ / Tenga en cuenta que usamos R.layout.incoming\_message\_panel como el ID de

/ / La notificación de la. Podría ser cualquier número entero que quieras, pero el uso que

/ / La convención de usar un identificador de recurso para una cadena relacionada con la

/ / La notificación de la. Siempre será un número único dentro de su

/ / Aplicación.

nm.notify (R.string.imcoming\_message\_ticker\_text, entr);

}

En un estilo de calendario de la notificación, la interfaz de usuario puesto en marcha por la notificación es una actividad exclusiva que no es parte del flujo normal de la aplicación. Por ejemplo, cuando el usuario recibe una notificación de calendario, la elección de que la notificación se inicia una actividad especial que muestra una lista de próximos eventos en el calendario - este punto de vista sólo está disponible a partir de la notificación, no a través de la interfaz del calendario de un usuario normal.

El código para publicar este tipo de notificación es muy directo, es como el anterior, pero el [PendingIntent](http://developer.android.com/reference/android/app/PendingIntent.html) es sólo para una sola actividad, nuestra actividad de notificación específico.

/ \*\*

\* La notificación es el icono y la entrada ampliada asociado en el

Barra de estado

\* /

showInterstitialNotification vacío () {

/ / Buscar el servicio de gestor de notificaciones

Nm NotificationManager = (NotificationManager) getSystemService (NOTIFICATION\_SERVICE);

/ / Los detalles de nuestro mensaje falso

CharSequence from = "Dianne";

CharSequence mensaje;

interruptor ((new Random (). nextInt ())% 3) {

caso 0: mensaje = "estoy listo para algo de cena"; break;

case 1: message = "? ¿qué tailandés de la manzana" [ break ; ]

default: mensaje = ". cumplir u soon EXCLUIR b tarde!" [ break ; ]

}

/ / El PendingIntent para poner en marcha nuestra actividad, si el usuario selecciona esta

notificación Nótese el uso de FLAG\_CANCEL\_CURRENT de modo que, si hay

/ / Ya es un intento de adaptación activa en espera, que anula y sustituye

/ / Con el nuevo intento.

Intención de Intención = nuevo intento (esto, IncomingMessageInterstitial.class);

intent.putExtra (IncomingMessageView.KEY\_FROM, desde);

intent.putExtra (IncomingMessageView.KEY\_MESSAGE, mensaje);

intent.addFlags (Intent.FLAG\_ACTIVITY\_NEW\_TASK | Intención.FLAG\_ACTIVITY\_CLEAR\_TASK);

ContentIntent PendingIntent = PendingIntent.getActivity (este, 0,

intención, PendingIntent.FLAG\_CANCEL\_CURRENT);

/ / El texto clave de pizarra, se utiliza una cadena con formato por lo que nuestro mensaje pueda ser localizado

Cadena tickerText = getString (R.string.imcoming\_message\_ticker\_text, mensaje);

/ / Construir el objeto de notificación.

Notificación entr = nueva notificación (R.drawable.stat\_sample, tickerText,

System.currentTimeMillis ());

/ / Establecer la información de las opiniones que se muestran en el panel de notificación.

notif.setLatestEventInfo (esto, de, mensaje, contentIntent);

/ / Vamos a hacer que esta notificación el sonido por defecto, la vibración y led.

/ / Tenga en cuenta que si usted desea alguno de estos comportamientos, que siempre debe tener

/ / Una preferencia por el usuario para desactivarlas.

notif.defaults = Notificación.DEFAULT\_ALL;

/ / Tenga en cuenta que usamos R.layout.incoming\_message\_panel como el ID de

/ / La notificación de la. Podría ser cualquier número entero que quieras, pero el uso que

/ / La convención de usar un identificador de recurso para una cadena relacionada con la

/ / La notificación de la. Siempre será un número único dentro de su

/ / Aplicación.

nm.notify (R.string.imcoming\_message\_ticker\_text, entr);

}

Esto no es suficiente, sin embargo. Normalmente Android considera todas las actividades dentro de una aplicación para formar parte del flujo de la interfaz de usuario de dicha aplicación, por lo que simplemente el lanzamiento de la actividad de este tipo puede hacer que se mezcla con la aplicación normal de pila en forma no deseada. Para hacer que se comporte correctamente, en la declaración de manifiesto para la actividad de la establecer. La declaración de plena actividad para esta muestra es la siguiente:

<Actividad de android: name = "app..IncomingMessageInterstitial "

android: label = "Usted tiene mensajes"

Android: tema = "@ estilo / ThemeHoloDialog"

android: launchMode = "singleTask"

android: taskAffinity = ""

android: excludeFromRecents = "true">

</ Actividad>

Usted debe tener cuidado al poner en marcha otras actividades de esta actividad inicial, porque esto no es una parte de alto nivel de la aplicación, no aparece en recientes al respecto, y debe ser relanzado en cualquier momento desde la notificación con los nuevos datos que mostrar. Este enfoque es mejor para asegurarse de que cualquier actividad en marcha desde que se puso en marcha en su propia tarea. Al hacer esto se debe tener cuidado para asegurarse de que esta nueva tarea se relaciona bien con el estado actual de la tarea de salir de la aplicación de. Esto es esencialmente el mismo que el cambio a la aplicación principal como se describe para la notificación de estilo correo electrónico mostrado antes. Dada la makeMessageIntentStack() el método mostrado anteriormente, el manejo de un clic a continuación, sería algo como esto:

/ \*\*

\* Lleve a cabo un cambio a la aplicación. Una pila nueva actividad se ha iniciado, sustituyendo

\* Lo que se está ejecutando actualmente, y esta actividad ha terminado.

\* /

void switchToApp () {

/ / Vamos a lanzar la aplicación que muestra lo que el usuario eligió. En este sencillo

/ / Ejemplo, es justo lo que la notificación nos ha dado.

CharSequence de getIntent = (). GetCharSequenceExtra (IncomingMessageView.KEY\_FROM);

CharSequence msg = getIntent (). GetCharSequenceExtra (IncomingMessageView.KEY\_MESSAGE);

/ / Construir la pila nueva actividad, inicie y termine de esta interfaz de usuario.

Intención [] = IncomingMessage.makeMessageIntentStack pila (esto, a partir, msg);

startActivities (pila);

&Finalizar

}

**Gestión de las notificaciones**

El [NotificationManager](http://developer.android.com/reference/android/app/NotificationManager.html) es un servicio del sistema que gestiona todas las notificaciones. Debe recuperar una referencia a él con la [getSystemService()](http://developer.android.com/reference/android/app/Activity.html#getSystemService%28java.lang.String%29) método. Por ejemplo:

Cadena ns = Contexto.NOTIFICATION\_SERVICE;

MNotificationManager NotificationManager = (NotificationManager) getSystemService (ns);

Cuando usted quiere entregar su notificación barra de estado, pase la [Notification](http://developer.android.com/reference/android/app/Notification.html) a la [NotificationManager](http://developer.android.com/reference/android/app/NotificationManager.html) con [notify(int, Notification)](http://developer.android.com/reference/android/app/NotificationManager.html#notify%28int,%20android.app.Notification%29) . El primer parámetro es el identificador único para la notificación y el segundo es la [Notification](http://developer.android.com/reference/android/app/Notification.html) objeto. El ID identifica de forma única la notificación en su aplicación. El ID es necesaria si es necesario actualizar la notificación, o (si su aplicación gestiona diferentes tipos de notificaciones) seleccione la acción apropiada cuando el usuario vuelve a su aplicación a través de la intención de definir en la notificación.

Para borrar la notificación de la barra de estado cuando el usuario selecciona desde la ventana de notificaciones, agregar el "FLAG\_AUTO\_CANCEL" bandera a su [Notification](http://developer.android.com/reference/android/app/Notification.html) . También puede borrar manualmente con la [cancel(int)](http://developer.android.com/reference/android/app/NotificationManager.html#cancel%28int%29) , pasándole el identificador de la notificación, o desactivar todas las notificaciones con [cancelAll()](http://developer.android.com/reference/android/app/NotificationManager.html#cancelAll%28%29) .

**Creación de una notificación**

Una [Notification](http://developer.android.com/reference/android/app/Notification.html) objeto define los detalles del mensaje de notificación que se muestra en la barra de estado y la ventana de notificaciones, y cualquier configuración de alerta, tales como los sonidos y las luces intermitentes.

La notificación de la barra de estado *requiere que* todos de los siguientes:

* Un icono de la barra de estado
* Un título y un mensaje, a menos que se define un [diseño de la notificación personalizada](http://developer.android.com/guide/topics/ui/notifiers/notifications.html#CustomExpandedView)
* Un [PendingIntent](http://developer.android.com/reference/android/app/PendingIntent.html) , que se disparó cuando la notificación se ha seleccionado

Los ajustes opcionales para la notificación de la barra de estado incluyen:

* Un mensaje clave de pizarra de texto para la barra de estado
* Un sonido de alerta
* Un ajuste de vibración
* Un LED parpadeante ajuste

El motor de arranque-kit para una nueva notificación incluye la [Notification(int, CharSequence, long)](http://developer.android.com/reference/android/app/Notification.html#Notification%28int,%20java.lang.CharSequence,%20long%29) y el constructor de [setLatestEventInfo(Context, CharSequence, CharSequence, PendingIntent)](http://developer.android.com/reference/android/app/Notification.html#setLatestEventInfo%28android.content.Context,%20java.lang.CharSequence,%20java.lang.CharSequence,%20android.app.PendingIntent%29) método. Estos definen todos los ajustes necesarios para una notificación. El siguiente fragmento de código muestra una configuración básica de la notificación:

icono de int = R.drawable.notification\_icon; / / icono de los recursos

CharSequence tickerText = "Hola"; / / ticker de texto

de largo cuando = System.currentTimeMillis (); / / tiempo de la notificación

Context = getApplicationContext (); / / contexto de aplicación

CharSequence contentTitle = "Mi notificación"; / / mensaje de título

CharSequence contentText = "Hello World!"Mensaje de texto

NotificationIntent Intención = nuevo intento (esto, MyClass.class);

ContentIntent PendingIntent = PendingIntent.getActivity (este, 0, notificationIntent, 0);

/ / Las siguientes dos líneas inicializar la notificación, utilizando las configuraciones por encima de

Notificación de la notificación = nueva notificación (icono, tickerText, cuándo);

notification.setLatestEventInfo (contexto, contentTitle, contentText, contentIntent);

**Actualización de la notificación**

Puede actualizar la información en su notificación de la barra de estado como los acontecimientos se siguen produciendo en su aplicación. Por ejemplo, cuando un nuevo mensaje de texto SMS llega antes de los mensajes anteriores han sido leídos, con las actualizaciones de la aplicación de mensajería la notificación existente para mostrar el número total de mensajes nuevos recibidos. Esta práctica de la actualización de una notificación existente es mucho mejor que la adición de nuevas notificaciones, ya que evita el desorden en la ventana de notificaciones.

Debido a que cada notificación se identifica por el [NotificationManager](http://developer.android.com/reference/android/app/NotificationManager.html) con un identificador de número entero, puede revisar la notificación llamando [setLatestEventInfo()](http://developer.android.com/reference/android/app/Notification.html#setLatestEventInfo%28android.content.Context,%20java.lang.CharSequence,%20java.lang.CharSequence,%20android.app.PendingIntent%29) con los nuevos valores, cambiar algunos valores de los campos de la notificación y, a continuación, llamar a [notify()](http://developer.android.com/reference/android/app/NotificationManager.html#notify%28int,%20android.app.Notification%29) de nuevo.

Usted puede revisar cada propiedad con los campos de miembro de objeto (excepto para el [Context](http://developer.android.com/reference/android/content/Context.html) y el título de la notificación y texto). Usted siempre debe revisar el mensaje de texto cuando se actualiza la notificación llamando [setLatestEventInfo()](http://developer.android.com/reference/android/app/Notification.html#setLatestEventInfo%28android.content.Context,%20java.lang.CharSequence,%20java.lang.CharSequence,%20android.app.PendingIntent%29) con los nuevos valores para *contentTitle* y *contentText.* Luego llame a [notify()](http://developer.android.com/reference/android/app/NotificationManager.html#notify%28int,%20android.app.Notification%29) para actualizar la notificación. (Por supuesto, si usted ha creado un [diseño de la notificación personalizada](http://developer.android.com/guide/topics/ui/notifiers/notifications.html#CustomExpandedView) , a continuación, la actualización de estos valores título y el texto no tiene ningún efecto.)

**Adición de un sonido**

Usted puede alertar al usuario con el sonido de notificación por defecto (que se define por el usuario) o con un sonido especificado por la aplicación.

Para usar el sonido predeterminado del usuario, añadir "DEFAULT\_SOUND" para el campo *por defecto:*

notification.defaults | Notificación =.DEFAULT\_SOUND;

Para utilizar un sonido diferente a las notificaciones, pasar una referencia de Uri para el campo de *sonido.* El siguiente ejemplo utiliza un archivo de audio conocido guardan en la tarjeta SD del dispositivo:

notification.sound = Uri.parse ("file :/ / / sdcard/notification/ringer.mp3");

En el siguiente ejemplo, el archivo de audio se elige en el interior de [MediaStore](http://developer.android.com/reference/android/provider/MediaStore.Audio.Media.html) 's [ContentProvider](http://developer.android.com/reference/android/content/ContentProvider.html) :

notification.sound = Uri.withAppendedPath (Audio.medios de comunicaciónINTERNAL\_CONTENT\_URI, "6");

En este caso, la identificación exacta del archivo de medios ("6") es conocido y se añade al contenido [Uri](http://developer.android.com/reference/android/net/Uri.html) . Si no conoce el ID exacto, usted debe consultar a todos los medios disponibles en el [MediaStore](http://developer.android.com/reference/android/provider/MediaStore.html) con un [ContentResolver](http://developer.android.com/reference/android/content/ContentResolver.html) . Vea el [contenido de los proveedores de](http://developer.android.com/guide/topics/providers/content-providers.html) la documentación para obtener más información sobre el uso de un ContentResolver.

Si quieres que el sonido se repetirá continuamente hasta que el usuario responde a la notificación o la notificación se cancela, agregue [FLAG\_INSISTENT](http://developer.android.com/reference/android/app/Notification.html#FLAG_INSISTENT) al campo de *las banderas.*

**Nota:** Si el campo *por defecto* incluye [DEFAULT\_SOUND](http://developer.android.com/reference/android/app/Notification.html#DEFAULT_SOUND) , entonces el sonido por defecto anula cualquier sonido se define en el campo de *sonido.*

**Adición de las vibraciones**

Usted puede alertar al usuario con el patrón por defecto la vibración o con un patrón de vibración se define por su aplicación.

Para utilizar el patrón predeterminado, agregue [DEFAULT\_VIBRATE](http://developer.android.com/reference/android/app/Notification.html#DEFAULT_VIBRATE) al campo *por defecto:*

notification.defaults | Notificación =.DEFAULT\_VIBRATE;

Para definir su patrón de vibración propia, pasar una matriz de valores de *largo* para el campo *de vibración:*

tiempo [] = {vibre 0100200300};

notification.vibrate = vibrar;

La matriz de larga define el patrón alterno para la longitud de la vibración de vez en cuando (en milisegundos). El primer valor es el tiempo de espera (apagado) antes de empezar, el segundo valor es la longitud de la primera vibración, la tercera es la siguiente longitud apagado, y así sucesivamente. El patrón puede ser tan largo como quieras, pero no puede ser programado para repetir.

**Nota:** Si el campo *por defecto* incluye [DEFAULT\_VIBRATE](http://developer.android.com/reference/android/app/Notification.html#DEFAULT_VIBRATE) , entonces la vibración por defecto anula cualquier tipo de vibración se define en el campo *de vibración.*

**Adición de luces intermitentes**

Para alertar al usuario mediante el parpadeo de las luces LED, se puede implementar el patrón de luz por defecto (si está disponible), o definir su propio color y el patrón de las luces.

Para utilizar el ajuste de la luz por defecto, añadir [DEFAULT\_LIGHTS](http://developer.android.com/reference/android/app/Notification.html#DEFAULT_LIGHTS) al campo *por defecto:*

notification.defaults | Notificación =.DEFAULT\_LIGHTS;

Para definir su propio color y el patrón, defina un valor para el campo *ledARGB* (por el color), el campo *ledOffMS* (período de tiempo, en milisegundos, para mantener la luz apagada), los *ledOnMS* (período de tiempo, en milisegundos, que mantener la luz encendida), y también añadir [FLAG\_SHOW\_LIGHTS](http://developer.android.com/reference/android/app/Notification.html#FLAG_SHOW_LIGHTS) al campo de *las banderas:*

notification.ledARGB = 0xff00ff00;

notification.ledOnMS = 300;

notification.ledOffMS = 1000;

notification.flags | = Notificación.FLAG\_SHOW\_LIGHTS;

En este ejemplo, la luz verde parpadea varias veces durante 300 milisegundos y se apaga durante un segundo. No todos los colores en el espectro con el apoyo de los LEDs del equipo, y no todas las impresoras admiten los mismos colores, por lo que el equipo estima que a lo mejor de su capacidad. El verde es el color más común de notificación.

**Más características**

Usted puede agregar más características a sus notificaciones mediante [Notification](http://developer.android.com/reference/android/app/Notification.html) campos y banderas. Algunas características útiles incluyen los siguientes:

[FLAG\_AUTO\_CANCEL](http://developer.android.com/reference/android/app/Notification.html#FLAG_AUTO_CANCEL) bandera

Agregue esto a la esfera *banderas* para cancelar automáticamente la notificación después de que se ha seleccionado desde la ventana de notificaciones.

[FLAG\_INSISTENT](http://developer.android.com/reference/android/app/Notification.html#FLAG_INSISTENT) bandera

Añadir esto a la *banderas de* campo para repetir el audio hasta que el usuario responde.

[FLAG\_ONGOING\_EVENT](http://developer.android.com/reference/android/app/Notification.html#FLAG_ONGOING_EVENT) bandera

Añadir esto a la *banderas de* campo para agrupar la notificación prevista en el "curso" del título en la ventana de notificaciones. Esto indica que la aplicación está en curso - los procesos se siguen ejecutando en segundo plano, incluso cuando la aplicación no es visible (por ejemplo, con la música o una llamada telefónica).

[FLAG\_NO\_CLEAR](http://developer.android.com/reference/android/app/Notification.html#FLAG_NO_CLEAR) bandera

Agregue esto a la esfera *banderas* para indicar que la notificación *no* debe ser aprobado por la "notificaciones claras" botón. Esto es particularmente útil si la notificación es en curso.

[number](http://developer.android.com/reference/android/app/Notification.html#number) de campo

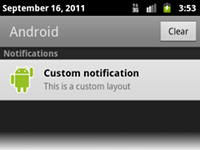
Este valor indica el número actual de los acontecimientos representados por la notificación. El número apropiado se superpone en la parte superior del icono de la barra de estado. Si tiene intención de utilizar este campo, entonces usted debe comenzar con "1" cuando la notificación se crea por primera vez. (Si cambia el valor de cero a algo más grande durante una actualización, el número no se muestra.)

[iconLevel](http://developer.android.com/reference/android/app/Notification.html#iconLevel) campo

Este valor indica el nivel actual de un [LevelListDrawable](http://developer.android.com/reference/android/graphics/drawable/LevelListDrawable.html) que se utiliza para el icono de notificación. Puede animar el icono en la barra de estado al cambiar este valor que se correlaciona con el dibujable está definido en un LevelListDrawable. Ver la [LevelListDrawable](http://developer.android.com/reference/android/graphics/drawable/LevelListDrawable.html) de referencia para más información.

Vea la [Notification](http://developer.android.com/reference/android/app/Notification.html) referencia de clase para obtener más información acerca de las características adicionales que se pueden personalizar para su aplicación.

**Crear un diseño personalizado de notificación**



**Figura 3.** Notificación con un diseño personalizado.

Por defecto, la notificación que aparece en la ventana de notificaciones incluye un título y el texto del mensaje. Estos se definen por los parámetros *contentTitle* y *contentText* del [setLatestEventInfo()](http://developer.android.com/reference/android/app/Notification.html#setLatestEventInfo%28android.content.Context,%20java.lang.CharSequence,%20java.lang.CharSequence,%20android.app.PendingIntent%29) método. Sin embargo, también puede definir un diseño personalizado de la notificación mediante [RemoteViews](http://developer.android.com/reference/android/widget/RemoteViews.html) . La Figura 3 muestra un ejemplo de un diseño de notificación personalizada. Es similar a la notificación por defecto, pero en realidad está creada con un diseño personalizado XML.

Para definir su propio diseño para la notificación, una instancia de un [RemoteViews](http://developer.android.com/reference/android/widget/RemoteViews.html) objeto que se infla un archivo de diseño personalizado, a continuación, pasar los [RemoteViews](http://developer.android.com/reference/android/widget/RemoteViews.html) al campo *contentView* de su notificación.

Crear una presentación de la notificación personalizada se entiende mejor con un ejemplo:

1. Crear el diseño de XML para la notificación. Por ejemplo, el siguiente diseño se llama custom\_notification.xml :
2. <RelativeLayout xmlns: android = "http://schemas.android.com/apk/res/android"
3. android: id = "+ id @ / layout"
4. android: layout\_width = "fill\_parent"
5. android: layout\_height = "fill\_parent"
6. Android: padding = "10DP">
7. <ImageView android: id = "+ id @ / imagen"
8. android: layout\_width = "wrap\_content"
9. android: layout\_height = "fill\_parent"
10. android: layout\_alignParentLeft = "true"
11. android: layout\_marginRight = "10DP" />
12. <TextView android: id = "+ id @ / title"
13. android: layout\_width = "wrap\_content"
14. android: layout\_height = "wrap\_content"
15. android: layout\_toRightOf = "@ id / imagen"
16. style = "@ estilo / NotificationTitle" />
17. <TextView android: id = "+ id @ / texto"
18. android: layout\_width = "wrap\_content"
19. android: layout\_height = "wrap\_content"
20. android: layout\_toRightOf = "@ id / imagen"
21. android: layout\_below = "@ id / title"
22. style = "@ estilo / NotificationText" />
23. </ RelativeLayout>

Tenga en cuenta que los dos [TextView](http://developer.android.com/reference/android/widget/TextView.html) elementos incluyen el style atributo. Es importante que utilice los recursos de estilo para el texto de las notificaciones personalizadas, ya que el color de fondo de la notificación puede variar entre los diferentes dispositivos y versiones de la plataforma. Empezando con Android 2.3 (nivel de la API 9), el sistema define un estilo para el texto que utiliza para los diseños predeterminados de notificación. Por lo tanto, usted debe aplicar ese estilo cuando se ejecuta en Android 2.3 o superior para asegurarse de que su texto sea visible contra el fondo.

Por ejemplo, para usar los colores de texto estándar en las versiones anteriores a la 2.3 para Android, usted debe utilizar los siguientes estilos de res/values/styles.xml :

<?xml version="1.0" encoding="utf-8"?>

<resources>

<style name="NotificationText">

<item name="android:textColor">?android: attr / textColorPrimary </ item>

</style>

<style name="NotificationTitle">

<item name="android:textColor">?android: attr / textColorPrimary </ item>

<item name="android:textStyle"> negrita </ ​​item>

</style>

<- Si desea un color ligeramente diferente de un texto,

considerar el uso de?android: attr / textColorSecondary ->

</ Recursos>

A continuación, para aplicar colores predeterminados del sistema para las notificaciones en Android 2.3 y superior, utilice los siguientes estilos de res/values-v9/styles.xml :

<?xml version="1.0" encoding="utf-8"?>

<resources>

<Nombre del estilo = "NotificationText" padre = "android: TextAppearance.StatusBar.EventContent "/>

<Nombre del estilo = "NotificationTitle" padre = "android: TextAppearance.StatusBar.EventContent.Cargo:

</ Recursos>

Ahora, cuando se ejecuta en Android 2.3 (nivel de la API 9) o superior, el texto en la vista personalizada a utilizar los mismos colores que hace el sistema de notificaciones por defecto. Esto es importante porque las últimas versiones de Android en realidad cambiar el color de fondo de las notificaciones a ser oscuro. La herencia de estilos del sistema se asegura de que su texto se haga la luz en estos casos, sino también si el fondo es un color inesperado otra parte, el texto también cambiará según el caso.

1. Ahora, en el código de la aplicación, utilice los métodos RemoveViews para definir la imagen y el texto. A continuación, pase el objeto RemoteViews al campo *contentView* de la notificación, como se muestra en este ejemplo:
2. RemoteViews contentView = new RemoteViews (getPackageName (), R.layout.custom\_notification\_layout);
3. contentView.setImageViewResource (R.id.image, R.drawable.notification\_image);
4. contentView.setTextViewText (R.id.title, "Notificación personalizada");
5. contentView.setTextViewText (R.id.text, "Este es un diseño personalizado");
6. notification.contentView = contentView;

Como se muestra aquí, pasar el nombre de la aplicación del paquete y el identificador de recursos de diseño para el constructor RemoteViews. A continuación, definir el contenido de la ImageView y TextView, utilizando el [setImageViewResource()](http://developer.android.com/reference/android/widget/RemoteViews.html#setImageViewResource%28int,%20int%29) y [setTextViewText()](http://developer.android.com/reference/android/widget/RemoteViews.html#setTextViewText%28int,%20java.lang.CharSequence%29) . En cada caso, pasar el identificador de referencia del objeto View apropiado que desea establecer, junto con el valor de esa opinión. Finalmente, el objeto RemoteViews se pasa a la notificación en el campo *contentView.*

1. Debido a que usted no necesita la [setLatestEventInfo()](http://developer.android.com/reference/android/app/Notification.html#setLatestEventInfo%28android.content.Context,%20java.lang.CharSequence,%20java.lang.CharSequence,%20android.app.PendingIntent%29) método cuando se utiliza una vista personalizada, debe definir la intención de la notificación con el campo *contentIntent,* como en este ejemplo:
2. NotificationIntent Intención = nuevo intento (esto, MyClass.class);
3. ContentIntent PendingIntent = PendingIntent.getActivity (este, 0, notificationIntent, 0);
4. notification.contentIntent = contentIntent;
5. La notificación se puede enviar como de costumbre:

mNotificationManager.notify (CUSTOM\_VIEW\_ID, la notificación);

El [RemoteViews](http://developer.android.com/reference/android/widget/RemoteViews.html) clase también incluye métodos que puede utilizar para agregar fácilmente un [Chronometer](http://developer.android.com/reference/android/widget/Chronometer.html) o [ProgressBar](http://developer.android.com/reference/android/widget/ProgressBar.html) en el diseño de su notificacion. Para obtener más información acerca de la creación de diseños personalizados para su notificación, se refieren a la [RemoteViews](http://developer.android.com/reference/android/widget/RemoteViews.html) referencia de clase.

**Precaución:** Cuando se crea un diseño de la notificación personalizada, debe tener especial cuidado para asegurarse de que sus funciones de diseño personalizados correctamente en dispositivos diferentes orientaciones y resoluciones. Si bien este consejo se aplica a todos los diseños de vista creada en Android, es especialmente importante en este caso porque su patrimonio real de diseño es muy restringido. Así que no hagas que tu diseño personalizado demasiado complejo y asegúrese de probar en varias configuraciones.

**Arrastrar y soltar**

**Vista Rapida**

* Permitir a los usuarios mover los datos dentro de su actividad de diseño el uso de gestos gráficos.
* Soporta, además de las operaciones de movimiento de datos.
* Sólo funciona en una sola aplicación.
* Requiere de la API 11.

**En este documento**

1. [Descripción general](http://developer.android.com/guide/topics/ui/drag-drop.html#AboutDragging)
   1. [El proceso de arrastre / caída](http://developer.android.com/guide/topics/ui/drag-drop.html#DragDropLifecycle)
   2. [El detector de eventos de arrastrar y método de devolución de llamada](http://developer.android.com/guide/topics/ui/drag-drop.html#AboutDragListeners)
   3. [Arrastre eventos](http://developer.android.com/guide/topics/ui/drag-drop.html#AboutDragEvent)
   4. [La sombra de arrastre](http://developer.android.com/guide/topics/ui/drag-drop.html#AboutDragShadowBuilder)
2. [El diseño de una operación de arrastrar y soltar](http://developer.android.com/guide/topics/ui/drag-drop.html#DesignDragOperation)
   1. [Inicio de una resistencia](http://developer.android.com/guide/topics/ui/drag-drop.html#StartDrag)
   2. [En respuesta a un inicio de arrastre](http://developer.android.com/guide/topics/ui/drag-drop.html#HandleStart)
   3. [Gestión de eventos durante el arrastre](http://developer.android.com/guide/topics/ui/drag-drop.html#HandleDuring)
   4. [En respuesta a una caída](http://developer.android.com/guide/topics/ui/drag-drop.html#HandleDrop)
   5. [En respuesta a un callejón sin arrastre](http://developer.android.com/guide/topics/ui/drag-drop.html#HandleEnd)
   6. [En respuesta a arrastrar eventos: un ejemplo](http://developer.android.com/guide/topics/ui/drag-drop.html#RespondEventSample)

**Clases de clave**

1. [View](http://developer.android.com/reference/android/view/View.html)
2. [OnLongClickListener](http://developer.android.com/reference/android/view/View.OnLongClickListener.html)
3. [OnDragListener](http://developer.android.com/reference/android/view/View.OnDragListener.html)
4. [DragEvent](http://developer.android.com/reference/android/view/DragEvent.html)
5. [DragShadowBuilder](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html)
6. [ClipData](http://developer.android.com/reference/android/content/ClipData.html)
7. [ClipDescription](http://developer.android.com/reference/android/content/ClipDescription.html)

**Muestras relacionadas**

1. [Galería de nido de abeja](http://developer.android.com/resources/samples/HoneycombGallery/index.html) .
2. [DragAndDropDemo.java](http://developer.android.com/resources/samples/ApiDemos/src/com/example/android/apis/view/DragAndDropDemo.html) y [DraggableDot.java](http://developer.android.com/resources/samples/ApiDemos/src/com/example/android/apis/view/DraggableDot.html) en [Demos Api](http://developer.android.com/resources/samples/ApiDemos/index.html) .

**Véase también**

1. [Proveedores del contenido](http://developer.android.com/guide/topics/providers/content-providers.html)
2. [Sucesos de Entrada](http://developer.android.com/guide/topics/ui/ui-events.html)

Con el Android arrastrar / soltar el marco, puede permitir a los usuarios mover los datos de una vista a otra vista en el diseño actual con una carga gráfica y el gesto de caída. El marco incluye algunos métodos de arrastre de la clase de eventos, los oyentes de arrastre, y el ayudante y clases.

Aunque el marco está diseñado principalmente para el movimiento de datos, se puede utilizar para las acciones de la interfaz de usuario de otros. Por ejemplo, podría crear una aplicación que mezcla los colores cuando el usuario arrastra el icono de un color sobre otro icono. El resto de este tema, sin embargo, se describe el marco en términos de movimiento de datos.

**Descripción general**

Una operación de arrastrar y soltar se inicia cuando el usuario realiza un gesto que reconocemos como una señal para empezar a arrastrar los datos. En respuesta, la aplicación le indica al sistema que la resistencia se está iniciando. El sistema vuelve a llamar a su aplicación para obtener una representación de los datos que se arrastran. Como el dedo del usuario se mueve esta representación (una "sombra de arrastre") sobre la estructura actual, el sistema envía arrastrar eventos a los objetos detectores de eventos de arrastre y arrastre los métodos de devolución de llamada de eventos asociados a las [View](http://developer.android.com/reference/android/view/View.html) los objetos en el diseño. Una vez que el usuario suelta la sombra de arrastre, el sistema termina la operación de arrastre.

Para crear un evento de arrastre objeto detector ("oyentes") de una clase que implementa [View.OnDragListener](http://developer.android.com/reference/android/view/View.OnDragListener.html) . Se establece el evento de arrastre objeto detector para una vista con el objeto Ver los [setOnDragListener()](http://developer.android.com/reference/android/view/View.html#setOnDragListener%28android.view.View.OnDragListener%29) método. Cada objeto de Vista también tiene un [onDragEvent()](http://developer.android.com/reference/android/view/View.html#onDragEvent%28android.view.DragEvent%29) el método de devolución de llamada. Ambos de estos se describen con más detalle en la sección [El oyente evento de arrastre y el método de devolución de llamada](http://developer.android.com/guide/topics/ui/drag-drop.html#AboutDragListeners) .

**Nota:** Por razones de simplicidad, las secciones siguientes se refieren a la rutina que recibe los eventos de arrastre como el "oyente de eventos de arrastre", a pesar de que en realidad puede ser un método de devolución de llamada.

Cuando se inicia un lastre, que incluyen tanto los datos que se mueven y los metadatos que describen estos datos como parte de la llamada al sistema. Durante el arrastre, el sistema envía los eventos de arrastrar a los oyentes de eventos de arrastre o los métodos de devolución de llamada de cada vista en el diseño. Los oyentes o los métodos de devolución de llamada se pueden utilizar los metadatos para decidir si desean aceptar los datos cuando se ha caído. Si el usuario coloca los datos a través de un objeto View, y el oyente ese objeto View o el método de devolución de llamada ya ha dicho que el sistema que quiere aceptar la caída, el sistema envía los datos al oyente o al método de devolución de llamada en un evento de arrastre.

Su aplicación le indica al sistema para iniciar un arrastre llamando a la [startDrag()](http://developer.android.com/reference/android/view/View.html#startDrag%28android.content.ClipData,%20android.view.View.DragShadowBuilder,%20java.lang.Object,%20int%29) método. Esto le indica al sistema para iniciar el envío de los eventos de arrastre. El método también envía los datos que se arrastran.

Usted puede llamar a [startDrag()](http://developer.android.com/reference/android/view/View.html#startDrag%28android.content.ClipData,%20android.view.View.DragShadowBuilder,%20java.lang.Object,%20int%29) para cualquier Ver adjunto en el diseño actual. El sistema sólo utiliza el objeto View para acceder a la configuración global de su trazado.

Una vez que la aplicación llama [startDrag()](http://developer.android.com/reference/android/view/View.html#startDrag%28android.content.ClipData,%20android.view.View.DragShadowBuilder,%20java.lang.Object,%20int%29) , el resto del proceso utiliza los eventos que el sistema envía a los objetos de vista de su diseño actual.

**El proceso de arrastre / caída**

Básicamente, hay cuatro pasos o estados en el proceso de arrastrar y soltar:

*Empezado*

En respuesta al gesto del usuario para iniciar una resistencia, la aplicación llama [startDrag()](http://developer.android.com/reference/android/view/View.html#startDrag%28android.content.ClipData,%20android.view.View.DragShadowBuilder,%20java.lang.Object,%20int%29) para indicar al sistema para iniciar un arrastre. Los argumentos [startDrag()](http://developer.android.com/reference/android/view/View.html#startDrag%28android.content.ClipData,%20android.view.View.DragShadowBuilder,%20java.lang.Object,%20int%29) proporcionar los datos que se arrastran, los metadatos para estos datos, y una devolución de llamada para dibujar la sombra de arrastre.

El primer sistema responde llamando de nuevo a su solicitud para obtener una sombra de arrastre. A continuación, muestra la sombra de arrastre en el dispositivo.

A continuación, el sistema envía un evento de arrastre con el tipo de acción [ACTION\_DRAG\_STARTED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_STARTED) a los oyentes de arrastre de eventos para todos los objetos de vista de la presentación actual. Para continuar recibiendo los eventos de arrastre, incluyendo un evento de colocación es posible, un detector de eventos de arrastrar debe devolver true . Esto registra el detector en el sistema. Sólo los detectores registrados continuar recibiendo los eventos de arrastre. En este punto, los oyentes pueden también cambiar la apariencia de su objeto View para mostrar que el oyente puede aceptar un evento de colocación.

Si el detector de eventos de arrastrar devuelve false , entonces no recibirá los eventos de arrastre para la operación actual hasta que el sistema envía un evento de arrastre con el tipo de acción [ACTION\_DRAG\_ENDED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENDED) . Mediante el envío de false , el que escucha le dice al sistema que no está interesado en la operación de arrastre y no quiere aceptar los datos arrastrados.

*Continuo*

El usuario continúa el arrastre. A medida que la sombra de la resistencia se cruza el cuadro delimitador de un objeto View, el sistema envía uno o más eventos de arrastrar al oyente el objeto View de arrastre evento (si está registrado para recibir los eventos). El oyente puede optar por alterar la apariencia de su objeto Vista en respuesta al evento. Por ejemplo, si el suceso indica que la sombra de arrastre ha entrado en el cuadro delimitador de la vista (tipo de acción [ACTION\_DRAG\_ENTERED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENTERED) ), el oyente puede reaccionar, poniendo de relieve su opinión.

*Abandonado*

El usuario libera a la sombra de arrastre dentro del cuadro delimitador de una vista que se pueden aceptar los datos. El sistema envía de detectores del objeto Ver un evento de arrastre con el tipo de acción [ACTION\_DROP](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DROP) . El evento de arrastre contiene los datos que se pasan al sistema en la llamada a [startDrag()](http://developer.android.com/reference/android/view/View.html#startDrag%28android.content.ClipData,%20android.view.View.DragShadowBuilder,%20java.lang.Object,%20int%29) que se inició la operación. El oyente se espera que regrese boolean true que el sistema si el código para la aceptación de la caída de éxito.

Tenga en cuenta que este paso sólo se produce si el usuario coloca la sombra de arrastre dentro de la caja de una vista cuyo detector se registra para recibir eventos de arrastre. Si el usuario suelta la sombra de arrastre en cualquier otra situación, no [ACTION\_DROP](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DROP) evento de arrastre se envía.

*Terminado*

Después de que el usuario suelta la sombra de arrastre, y después el sistema envía (si es necesario) un evento de arrastre con acción tipo [ACTION\_DROP](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DROP) , el sistema envía un evento de arrastre con el tipo de acción [ACTION\_DRAG\_ENDED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENDED) para indicar que la operación de arrastre se ha terminado. Esto se hace sin importar donde el usuario libera a la sombra de arrastre. El evento se envía a todos los oyentes que se registra para recibir eventos de arrastre, incluso si el oyente recibe el [ACTION\_DROP](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DROP) evento.

Cada uno de estos cuatro pasos se describe con más detalle en la sección de [diseño de una operación de arrastrar y soltar](http://developer.android.com/guide/topics/ui/drag-drop.html#DesignDragOperation) .

**El detector de eventos de arrastrar y método de devolución de llamada**

Un punto de vista recibe los eventos de arrastre, ya sea con un detector de eventos drag que implementa [View.OnDragListener](http://developer.android.com/reference/android/view/View.OnDragListener.html) o con su [onDragEvent(DragEvent)](http://developer.android.com/reference/android/view/View.html#onDragEvent%28android.view.DragEvent%29) el método de devolución de llamada. Cuando el sistema llama el método o el oyente, pasa a un [DragEvent](http://developer.android.com/reference/android/view/DragEvent.html) objeto.

Es probable que desee utilizar el detector en la mayoría de los casos. Cuando le interfaces de usuario de diseño, por lo general no lo hacen clases subclases Vista, pero utilizando el método de devolución de llamada te obliga a hacer esto con el fin de reemplazar el método. En comparación, se puede implementar una clase de oyente y luego usarlo con varios objetos de vista diferente. También se puede aplicar como una clase en línea anónimo. Para configurar el detector para un objeto View, llame [setOnDragListener()](http://developer.android.com/reference/android/view/View.html#setOnDragListener%28android.view.View.OnDragListener%29) .

Usted puede tener tanto un oyente y un método de devolución de llamada para el objeto View. Si esto ocurre, el primer sistema llama al oyente. El sistema no llama al método de devolución de llamada a menos que el oyente vuelve false .

La combinación de la [onDragEvent(DragEvent)](http://developer.android.com/reference/android/view/View.html#onDragEvent%28android.view.DragEvent%29) método y [View.OnDragListener](http://developer.android.com/reference/android/view/View.OnDragListener.html) es análoga a la combinación de la [onTouchEvent()](http://developer.android.com/reference/android/view/View.html#onTouchEvent%28android.view.MotionEvent%29) y [View.OnTouchListener](http://developer.android.com/reference/android/view/View.OnTouchListener.html) utilizarse con eventos táctiles.

**Arrastre eventos**

El sistema envía un evento de arrastre en forma de un [DragEvent](http://developer.android.com/reference/android/view/DragEvent.html) objeto. El objeto contiene un tipo de acción que le dice al oyente lo que está sucediendo en el proceso de arrastrar / soltar. El objeto contiene otros datos, dependiendo del tipo de acción.

Para obtener el tipo de acción, un oyente llama [getAction()](http://developer.android.com/reference/android/view/DragEvent.html#getAction%28%29) . Hay seis valores posibles, definidos por las constantes en la [DragEvent](http://developer.android.com/reference/android/view/DragEvent.html) clase. Estos se enumeran en [la tabla 1](http://developer.android.com/guide/topics/ui/drag-drop.html#table1) .

El [DragEvent](http://developer.android.com/reference/android/view/DragEvent.html) objeto también contiene los datos de que su aplicación previstas en el sistema en la llamada a [startDrag()](http://developer.android.com/reference/android/view/View.html#startDrag%28android.content.ClipData,%20android.view.View.DragShadowBuilder,%20java.lang.Object,%20int%29) . Algunos de los datos son válidos sólo para ciertos tipos de acciones. Los datos que son válidas para cada tipo de acción se resumen en la [tabla 2](http://developer.android.com/guide/topics/ui/drag-drop.html#table2) . También se ha descrito en detalle con el evento para el que es válida en la sección de [diseño de una operación de arrastrar y soltar](http://developer.android.com/guide/topics/ui/drag-drop.html#DesignDragOperation) .

**Tabla 1.** Tipos de acción DragEvent

|  |  |
| --- | --- |
| **getAction () el valor** | **Sentido** |
| [ACTION\_DRAG\_STARTED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_STARTED) | Escucha un objeto View de arrastre de eventos recibe este tipo de eventos la acción justo después de la aplicación llama a [startDrag()](http://developer.android.com/reference/android/view/View.html#startDrag%28android.content.ClipData,%20android.view.View.DragShadowBuilder,%20java.lang.Object,%20int%29) y se pone una sombra de arrastre. |
| [ACTION\_DRAG\_ENTERED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENTERED) | Escucha un objeto View de arrastre de eventos recibe este tipo de acción de sucesos cuando la sombra de arrastre acaba de entrar en el cuadro delimitador de la Vista. Este es el tipo de evento la primera acción del oyente recibe, cuando la sombra de arrastre entra en el cuadro de límite. Si el oyente quiere seguir recibiendo los eventos de arrastre para esta operación, debe devolver boolean true en el sistema. |
| [ACTION\_DRAG\_LOCATION](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_LOCATION) | Escucha un objeto View de arrastre de eventos recibe este tipo de acción de evento después de recibir un [ACTION\_DRAG\_ENTERED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENTERED) evento, mientras que la sombra de arrastre que esté dentro del rectángulo de selección de la vista. |
| [ACTION\_DRAG\_EXITED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_EXITED) | Escucha un objeto View de arrastre de eventos recibe este tipo de acción de evento después de recibir un [ACTION\_DRAG\_ENTERED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENTERED) y al menos un [ACTION\_DRAG\_LOCATION](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_LOCATION) caso y luego de que el usuario se ha movido a la sombra de arrastrar fuera del cuadro delimitador de la Vista. |
| [ACTION\_DROP](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DROP) | Escucha un objeto View de arrastre de eventos recibe este tipo de acción de evento cuando el usuario suelta la sombra de arrastre sobre el objeto View. Este tipo de acción sólo se envía al oyente un objeto View de si el oyente volvió booleano true en respuesta a la [ACTION\_DRAG\_STARTED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_STARTED) evento de arrastre. Este tipo de acción no se envía si el usuario suelta la sombra de arrastre en una vista cuya escucha no está registrado, o si el usuario suelta la sombra de arrastre en todo lo que no es parte de la presentación actual.  El oyente se espera que regrese booleano true si se procesa correctamente la caída. De lo contrario, debe devolver false . |
| [ACTION\_DRAG\_ENDED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENDED) | Escucha un objeto View de arrastre de eventos recibe este tipo de eventos de acción cuando el sistema está por terminar la operación de arrastre. Este tipo de acción no está necesariamente precedido por un [ACTION\_DROP](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DROP) evento. Si el sistema envió una [ACTION\_DROP](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DROP) , recibiendo el [ACTION\_DRAG\_ENDED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENDED) tipo de acción no implica que la operación de descenso con éxito. El oyente debe llamar [getResult()](http://developer.android.com/reference/android/view/DragEvent.html#getResult%28%29) para obtener el valor que fue devuelto en respuesta a [ACTION\_DROP](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DROP) . Si un [ACTION\_DROP](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DROP) evento no fue enviado, a continuación, [getResult()](http://developer.android.com/reference/android/view/DragEvent.html#getResult%28%29) devuelve false . |

**Tabla 2.** Los datos válidos DragEvent por tipo de acción

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| [**getAction()**](http://developer.android.com/reference/android/view/DragEvent.html#getAction%28%29) **el valor** | [**getClipDescription()**](http://developer.android.com/reference/android/view/DragEvent.html#getClipDescription%28%29) **el valor** | [**getLocalState()**](http://developer.android.com/reference/android/view/DragEvent.html#getLocalState%28%29) **el valor** | [**getX()**](http://developer.android.com/reference/android/view/DragEvent.html#getX%28%29) **el valor** | [**getY()**](http://developer.android.com/reference/android/view/DragEvent.html#getY%28%29) **el valor** | [**getClipData()**](http://developer.android.com/reference/android/view/DragEvent.html#getClipData%28%29) **el valor** | [**getResult()**](http://developer.android.com/reference/android/view/DragEvent.html#getResult%28%29) **el valor** |
| [ACTION\_DRAG\_STARTED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_STARTED) | X | X | X |  |  |  |
| [ACTION\_DRAG\_ENTERED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENTERED) | X | X | X | X |  |  |
| [ACTION\_DRAG\_LOCATION](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_LOCATION) | X | X | X | X |  |  |
| [ACTION\_DRAG\_EXITED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_EXITED) | X | X |  |  |  |  |
| [ACTION\_DROP](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DROP) | X | X | X | X | X |  |
| [ACTION\_DRAG\_ENDED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENDED) | X | X |  |  |  | X |

El [getAction()](http://developer.android.com/reference/android/view/DragEvent.html#getAction%28%29) , [describeContents()](http://developer.android.com/reference/android/view/DragEvent.html#describeContents%28%29) , [writeToParcel()](http://developer.android.com/reference/android/view/DragEvent.html#writeToParcel%28android.os.Parcel,%20int%29) y [toString()](http://developer.android.com/reference/android/view/DragEvent.html#toString%28%29) métodos siempre devuelven datos válidos.

Si un método no contiene datos válidos para un tipo de acción en particular, devuelve null o 0, dependiendo de su tipo de resultado.

**La sombra de arrastre**

Durante una operación de arrastrar y soltar, el sistema muestra una imagen que el usuario arrastra. Para el movimiento de datos, esta imagen representa los datos que se arrastran. Para otras operaciones, la imagen representa algún aspecto de la operación de arrastre.

La imagen se llama una sombra de arrastre. Usted lo crea con los métodos que se declara para una [View.DragShadowBuilder](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html) objeto, y luego pasarlo al sistema cuando se inicia una fricción con [startDrag()](http://developer.android.com/reference/android/view/View.html#startDrag%28android.content.ClipData,%20android.view.View.DragShadowBuilder,%20java.lang.Object,%20int%29) . Como parte de su respuesta a [startDrag()](http://developer.android.com/reference/android/view/View.html#startDrag%28android.content.ClipData,%20android.view.View.DragShadowBuilder,%20java.lang.Object,%20int%29) , el sistema llama a los métodos de devolución de llamada que ha definido en el [View.DragShadowBuilder](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html) para obtener una sombra de arrastre.

La [View.DragShadowBuilder](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html) clase tiene dos constructores:

[View.DragShadowBuilder(View)](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html#View.DragShadowBuilder%28android.view.View%29)

Este constructor acepta alguno de ellos de la aplicación [View](http://developer.android.com/reference/android/view/View.html) objetos. El constructor almacena el objeto de Vista en el [View.DragShadowBuilder](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html) objeto, por lo que durante la devolución de llamada se puede acceder a él a medida que la construcción de tu sombra de arrastre. No tiene que estar asociado con la Vista (si existe) que el usuario ha seleccionado para iniciar la operación de arrastre.

Si utiliza este constructor, usted no tiene para ampliar [View.DragShadowBuilder](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html) o invalidar sus métodos. Por defecto, usted recibirá una sombra de arrastre que tiene el mismo aspecto que la vista se pasa como argumento, centrado en la ubicación donde el usuario está en contacto con la pantalla.

[View.DragShadowBuilder()](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html#View.DragShadowBuilder%28%29)

Si utiliza este constructor, no hay ningún objeto View está disponible en la [View.DragShadowBuilder](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html) objeto (el campo se establece en null ). Si utiliza este constructor, y no se extienden [View.DragShadowBuilder](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html) o invalidar sus métodos, usted recibirá una sombra invisible de arrastre. El sistema *no* da un error.

La [View.DragShadowBuilder](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html) clase tiene dos métodos:

[onProvideShadowMetrics()](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html#onProvideShadowMetrics%28android.graphics.Point,%20android.graphics.Point%29)

El sistema llama a este método inmediatamente después de llamar [startDrag()](http://developer.android.com/reference/android/view/View.html#startDrag%28android.content.ClipData,%20android.view.View.DragShadowBuilder,%20java.lang.Object,%20int%29) . Se utiliza para enviar al sistema de las dimensiones y el punto de contacto de la sombra de arrastre. El método tiene dos argumentos:

*Dimensiones*

Un [Point](http://developer.android.com/reference/android/graphics/Point.html) objeto. El ancho de la sombra de arrastre va en [x](http://developer.android.com/reference/android/graphics/Point.html#x) y su altura va en [y](http://developer.android.com/reference/android/graphics/Point.html#y) .

*- Puntos de toque*

Un [Point](http://developer.android.com/reference/android/graphics/Point.html) objeto. El punto de contacto es la ubicación dentro de la sombra de arrastre que debe estar bajo el dedo del usuario durante el arrastre. Su posición va en X [x](http://developer.android.com/reference/android/graphics/Point.html#x) Y y su posición va en [y](http://developer.android.com/reference/android/graphics/Point.html#y)

[onDrawShadow()](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html#onDrawShadow%28android.graphics.Canvas%29)

Inmediatamente después de la llamada a [onProvideShadowMetrics()](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html#onProvideShadowMetrics%28android.graphics.Point,%20android.graphics.Point%29) las llamadas al sistema [onDrawShadow()](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html#onDrawShadow%28android.graphics.Canvas%29) para obtener la resistencia propia sombra. El método tiene un solo argumento, un [Canvas](http://developer.android.com/reference/android/graphics/Canvas.html) objeto de que el sistema construye a partir de los parámetros que se proporcionan en [onProvideShadowMetrics()](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html#onProvideShadowMetrics%28android.graphics.Point,%20android.graphics.Point%29) Se usa para dibujar la sombra de arrastre en la siempre [Canvas](http://developer.android.com/reference/android/graphics/Canvas.html) objeto.

Para mejorar el rendimiento, se debe mantener el tamaño de la pequeña sombra de arrastre. Para un solo elemento, es posible que desee utilizar un icono. Para una selección múltiple, es posible que desee utilizar los iconos en una pila en lugar de las imágenes completas se extienden a lo largo de la pantalla.

**El diseño de una operación de arrastrar y soltar**

En esta sección se muestra paso a paso cómo iniciar un arrastre, cómo responder a eventos durante el arrastre, ¿cómo responder a un evento de colocación, y cómo poner fin a la operación de arrastrar y soltar.

**Inicio de una resistencia**

El usuario inicia una fricción con un gesto de arrastre, por lo general una pulsación larga, en un objeto View. En respuesta, usted debe hacer lo siguiente:

1. Si es necesario, crear un [ClipData](http://developer.android.com/reference/android/content/ClipData.html) y [ClipData.Item](http://developer.android.com/reference/android/content/ClipData.Item.html) de los datos que se mueven. Como parte de los metadatos de objeto ClipData la oferta, que se almacena en un [ClipDescription](http://developer.android.com/reference/android/content/ClipDescription.html) objeto dentro de la ClipData. Para una operación de arrastrar y soltar que no representa el movimiento de datos, es posible que desee utilizar null en lugar de un objeto real.

Por ejemplo, este fragmento de código muestra cómo responder a una pulsación larga en una ImageView mediante la creación de un objeto ClipData que contiene el rótulo o etiqueta de un ImageView. Después de este fragmento, el fragmento de código siguiente muestra cómo reemplazar los métodos de [View.DragShadowBuilder](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html) :

/ / Crear una cadena de la etiqueta ImageView

privada IMAGEVIEW\_TAG static final String = "icono de mapa de bits"

/ / Crea un nuevo ImageView

ImageView imageView = new ImageView (este);

/ / Establece el mapa de bits para el ImageView de un icono de mapa de bits (que se define en otra parte)

imageView.setImageBitmap (mIconBitmap);

/ / Establece la etiqueta

imageView.setTag (IMAGEVIEW\_TAG);

...

/ / Establece un oyente haga clic en largo para el ImageView con un objeto de detector anónima que

/ / Implementa la interfaz OnLongClickListener

imageView.setOnLongClickListener (Ver nueva.OnLongClickListener () {

/ / Define el método uno para la interfaz, que se llama cuando la vista es a largo clic

public boolean onLongClick (Ver v) {

/ / Crear un nuevo ClipData.

/ / Esto se hace en dos etapas para proporcionar claridad. El método de conveniencia

/ / ClipData.newPlainText () puede crear un ClipData texto sin formato en un solo paso.

/ / Crear un nuevo ClipData.Artículo de la etiqueta de objeto de ImageView

ClipData.Tema = new ClipData.Artículo (v.getTag ());

/ / Crear un nuevo ClipData usando la etiqueta como una etiqueta, el texto sin formato tipo MIME, y

/ / El tema ya creado. Esto creará un objeto nuevo en el ClipDescription

/ / ClipData, y establece su entrada tipo MIME a "text / plain"

ClipData dragData = new ClipData (v.getTag (), ClipData.MIMETYPE\_TEXT\_PLAIN, elemento);

/ / Crea una instancia del constructor de la sombra de arrastre.

Ver.DrawShadowBuilder myShadow = new MyDragShadowBuilder (imageView);

/ / Inicia el arrastre

v.startDrag (dragData, / / ​​los datos que se arrastran

myShadow, / / ​​el constructor de la sombra de arrastre

NULL, / / ​​no hay necesidad de utilizar los datos locales

0 / / banderas (que actualmente no se utiliza, poner a 0)

);

}

}

1. El siguiente fragmento de código define myDragShadowBuilder Se crea una sombra de arrastre para arrastrar un TextView como un pequeño rectángulo gris:
2. MyDragShadowBuilder privada clase estática se extiende Vista.DragShadowBuilder {
3. / / La imagen de arrastre la sombra, que se define como una cosa dibujable
4. Disponibles privada sombra estática;
5. / / Define el constructor de myDragShadowBuilder
6. pública MyDragShadowBuilder (Ver v) {
7. / / Almacena el parámetro pasado a Vista myDragShadowBuilder.
8. super (v);
9. / / Crea una imagen que se pueda arrastrar llenar el lienzo que proporciona el sistema.
10. la sombra = new ColorDrawable (color.LTGRAY);
11. }
12. / / Define una devolución de llamada que envía las dimensiones de la sombra de arrastre y punto de contacto de nuevo a la
13. Sistemas
14. @Override
15. public void onProvideShadowMetrics (Punto de tamaño, Punto de contacto)
16. / / Define las variables locales
17. ancho de private int, altura;
18. / / Establece la anchura de la sombra a la mitad de la anchura de la vista original
19. . width = getView () getWidth () / 2;
20. / / Establece la altura de la sombra a la mitad de la altura de la vista original
21. . height = getView () getHeight () / 2;
22. / / La sombra de arrastre es una ColorDrawable. Esto establece sus dimensiones para ser el mismo que el
23. / / Lienzo que el sistema va a dar. Como resultado, la sombra de arrastre llenará el
24. Canvas
25. shadow.setBounds (0, 0, ancho, alto);
26. / / Establece el ancho del parámetro de tamaño y los valores de altura. Se trata de volver al sistema de
27. / / A través del parámetro de tamaño.
28. size.set (ancho, alto);
29. / / Establece la posición del punto de contacto para estar en el medio de la sombra de arrastre
30. touch.set (ancho / 2, altura / 2);
31. }
32. / / Define una devolución de llamada que atrae a la sombra de arrastre en un lienzo que el sistema construye
33. / / De las dimensiones aprobadas en onProvideShadowMetrics ().
34. @Override
35. onDrawShadow public void (lienzo lienzo) {
36. / / Dibuja el ColorDrawable en el lienzo en el pasado del sistema.
37. shadow.draw (lienzo);
38. }
39. }

**Nota:** Recuerde que usted no tiene para ampliar [View.DragShadowBuilder](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html) . El constructor de [View.DragShadowBuilder(View)](http://developer.android.com/reference/android/view/View.DragShadowBuilder.html#View.DragShadowBuilder%28android.view.View%29) crea una sombra arrastrar por defecto que es del mismo tamaño que el argumento de Vista le ha pasado, con el punto de contacto centrada en la sombra de arrastre.

**En respuesta a un inicio de arrastre**

Durante la operación de arrastre, los despachos del sistema arrastrar eventos a los oyentes de eventos de arrastre de los objetos de vista de la presentación actual. Los oyentes deben reaccionar llamando [getAction()](http://developer.android.com/reference/android/view/DragEvent.html#getAction%28%29) para obtener el tipo de acción. Al comienzo de un arrastre, estos métodos rendimientos [ACTION\_DRAG\_STARTED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_STARTED) .

En respuesta a un evento con el tipo de acción [ACTION\_DRAG\_STARTED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_STARTED) , un oyente debe hacer lo siguiente:

1. Llame [getClipDescription()](http://developer.android.com/reference/android/view/DragEvent.html#getClipDescription%28%29) para obtener el [ClipDescription](http://developer.android.com/reference/android/content/ClipDescription.html) . Utilice los métodos de tipo MIME en [ClipDescription](http://developer.android.com/reference/android/content/ClipDescription.html) para ver si el oyente puede aceptar los datos que se arrastran.

Si la operación de arrastre no representa el movimiento de datos, esto puede no ser necesario.

1. Si el oyente puede aceptar una caída, debe devolver true . Esto le indica al sistema para seguir enviando los eventos de arrastre para el oyente. Si no se puede aceptar una caída, debe devolver false , y el sistema dejará de enviar los eventos de arrastre hasta que se envía [ACTION\_DRAG\_ENDED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENDED) .

Tenga en cuenta que para un [ACTION\_DRAG\_STARTED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_STARTED) caso, estas las siguientes [DragEvent](http://developer.android.com/reference/android/view/DragEvent.html) métodos no son válidos: [getClipData()](http://developer.android.com/reference/android/view/DragEvent.html#getClipData%28%29) , [getX()](http://developer.android.com/reference/android/view/DragEvent.html#getX%28%29) , [getY()](http://developer.android.com/reference/android/view/DragEvent.html#getY%28%29) , y [getResult()](http://developer.android.com/reference/android/view/DragEvent.html#getResult%28%29) .

**Gestión de eventos durante el arrastre**

Durante el arrastre, los oyentes que devolvieron true en respuesta a la [ACTION\_DRAG\_STARTED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_STARTED) evento de arrastre continuará recibiendo eventos de arrastre. Los tipos de eventos de arrastrar un oyente recibe durante el arrastre dependen de la ubicación de la sombra de la fricción y la visibilidad de vista del oyente.

Durante el arrastre, los oyentes utilizan principalmente los eventos de arrastre para decidir si debe cambiar la apariencia de su punto de vista.

Durante el arrastre, [getAction()](http://developer.android.com/reference/android/view/DragEvent.html#getAction%28%29) devuelve uno de los tres valores siguientes:

* [ACTION\_DRAG\_ENTERED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENTERED) : El oyente recibe esto cuando el punto de contacto (el punto en la pantalla por debajo de los dedos del usuario) ha entrado en el cuadro delimitador de vista del oyente.
* [ACTION\_DRAG\_LOCATION](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_LOCATION) : Una vez que el detector reciba un [ACTION\_DRAG\_ENTERED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENTERED) caso, y antes de recibir Un [ACTION\_DRAG\_EXITED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_EXITED) caso, recibe un nuevo [ACTION\_DRAG\_LOCATION](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_LOCATION) eventos cada vez que se mueve el punto de contacto. El [getX()](http://developer.android.com/reference/android/view/DragEvent.html#getX%28%29) y [getY()](http://developer.android.com/reference/android/view/DragEvent.html#getY%28%29) métodos devuelven las coordenadas X e Y del punto de contacto.
* [ACTION\_DRAG\_EXITED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_EXITED) : Este evento es enviado a un oyente que previamente haya recibido [ACTION\_DRAG\_ENTERED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENTERED) , después de arrastrar la sombra ya no está dentro del cuadro delimitador de vista del oyente.

El oyente no necesita para reaccionar a cualquiera de estos tipos de acción. Si el oyente devuelve un valor al sistema, se omite. Aquí hay algunas pautas para responder a cada uno de estos tipos de acciones:

* En respuesta a [ACTION\_DRAG\_ENTERED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENTERED) o [ACTION\_DRAG\_LOCATION](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_LOCATION) , el oyente puede cambiar la apariencia de la vista para indicar que está a punto de recibir una gota.
* Un evento con el tipo de acción [ACTION\_DRAG\_LOCATION](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_LOCATION) contiene datos válidos para [getX()](http://developer.android.com/reference/android/view/DragEvent.html#getX%28%29) y [getY()](http://developer.android.com/reference/android/view/DragEvent.html#getY%28%29) , correspondiente a la ubicación del punto de contacto. El oyente puede que quiera usar esta información para modificar el aspecto de esa parte de la vista que está en el punto de contacto. El oyente también puede utilizar esta información para determinar la posición exacta donde el usuario va a caer la sombra de arrastre.
* En respuesta a [ACTION\_DRAG\_EXITED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_EXITED) , el oyente debe restablecer los cambios de apariencia que se aplican en respuesta a [ACTION\_DRAG\_ENTERED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENTERED) o [ACTION\_DRAG\_LOCATION](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_LOCATION) . Esto indica al usuario que la vista ya no es un destino de colocación inminente.

**En respuesta a una caída**

Cuando el usuario suelta la sombra de arrastre en una vista en la solicitud, y que Vista ya se ha informado de que podría aceptar el contenido que está siendo arrastrado, los despachos del sistema de un evento de arrastre a la vista con el tipo de acción [ACTION\_DROP](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DROP) . El oyente debe hacer lo siguiente:

1. Llame [getClipData()](http://developer.android.com/reference/android/view/DragEvent.html#getClipData%28%29) para obtener el [ClipData](http://developer.android.com/reference/android/content/ClipData.html) objeto que fue suministrada originalmente en la llamada a [startDrag()](http://developer.android.com/reference/android/view/View.html#startDrag%28android.content.ClipData,%20android.view.View.DragShadowBuilder,%20java.lang.Object,%20int%29) y lo almacena. Si la operación de arrastre no representa el movimiento de datos, esto puede no ser necesario.
2. Volver booleano true para indicar que la caída se ha procesado correctamente, o boolean false si no lo era. El valor devuelto se convierte en el valor devuelto por [getResult()](http://developer.android.com/reference/android/view/DragEvent.html#getResult%28%29) para un [ACTION\_DRAG\_ENDED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENDED) evento.

Nótese que si el sistema no envía una [ACTION\_DROP](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DROP) caso, el valor de [getResult()](http://developer.android.com/reference/android/view/DragEvent.html#getResult%28%29) para un [ACTION\_DRAG\_ENDED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENDED) evento es false .

Para una [ACTION\_DROP](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DROP) caso, [getX()](http://developer.android.com/reference/android/view/DragEvent.html#getX%28%29) y [getY()](http://developer.android.com/reference/android/view/DragEvent.html#getY%28%29) devuelve el X e Y del punto de arrastre en el momento de la caída, utilizando el sistema de coordenadas de la vista de que recibió la gota.

El sistema permite que el usuario suelte el lastre para la sombra de una vista cuya escucha no recibe los eventos de arrastre. También permitirá que el usuario suelte el lastre para la sombra regiones vacías de la interfaz de usuario de la aplicación, o en áreas fuera de su aplicación. En todos estos casos, el sistema no envía un evento con una acción tipo [ACTION\_DROP](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DROP) , aunque se envíe un [ACTION\_DRAG\_ENDED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENDED) evento.

**En respuesta a un callejón sin arrastre**

Inmediatamente después de que el usuario suelte la sombra de arrastre, el sistema envía un evento de arrastre a todos los oyentes de eventos de arrastrar en su aplicación, con un tipo de acción de [ACTION\_DRAG\_ENDED](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DRAG_ENDED) . Esto indica que la operación de arrastre se ha terminado.

Cada oyente debe hacer lo siguiente:

1. Si escucha cambió la apariencia de su objeto Vista durante la operación, se debe restablecer la vista a su apariencia por defecto. Esto es una indicación visual al usuario de que la operación ha terminado.
2. El oyente lo desea, puede llamar a [getResult()](http://developer.android.com/reference/android/view/DragEvent.html#getResult%28%29) para obtener más información acerca de la operación. Si un oyente volvió true en respuesta a un evento de tipo de acción [ACTION\_DROP](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DROP) , entonces [getResult()](http://developer.android.com/reference/android/view/DragEvent.html#getResult%28%29) devolverá boolean true . En los demás casos, [getResult()](http://developer.android.com/reference/android/view/DragEvent.html#getResult%28%29) devuelve boolean false , incluyendo los casos en que el sistema no enviará un [ACTION\_DROP](http://developer.android.com/reference/android/view/DragEvent.html#ACTION_DROP) evento.
3. El oyente debe devolver boolean true en el sistema.

**En respuesta a arrastrar eventos: un ejemplo**

Todos los eventos de arrastre son inicialmente recibidos por el método de evento de arrastre o el oyente. El siguiente fragmento de código es un ejemplo sencillo de reaccionar para arrastrar los acontecimientos en un oyente:

/ / Se crea un detector de arrastre nuevo evento

mDragListen myDragEventListener = new ();

Ver imageView = new ImageView (este);

/ / Establece el detector de eventos de arrastre para la vista

imageView.setOnDragListener (mDragListen);

...

myDragEventListener protegidos clase implementa Vista.OnDragEventListener {

/ / Este es el método que el sistema llama cuando se distribuye un evento de arrastrar a la

r

public boolean ondrag (Ver v, DragEvent evento) {

/ / Define una variable para almacenar el tipo de acción para el evento de entrada

la acción final int = event.getAction ();

/ / Controla cada uno de los eventos esperados

Función de Interruptor

DragEvent caso.ACTION\_DRAG\_STARTED:

/ / Determina si esta opinión puede aceptar los datos arrastrados

if (event.getClipDescription (). hasMimeType (ClipDescription.MIMETYPE\_TEXT\_PLAIN)) {

/ / Como un ejemplo de lo que su aplicación podría hacer,

/ / Aplica un tinte de color azul a la vista para indicar que se puede aceptar

'DATA'

v.setColorFilter (color.(AZUL)

/ / Invalidar el fin de forzar un rediseño en el tinte nueva

v.invalidate ();

/ / Devuelve true para indicar que la vista puede aceptar los datos arrastrados.

return true;

} else {

.return false; Durante la operación actual de arrastrar y soltar, esta vista

/ / No recibir eventos de nuevo hasta ACTION\_DRAG\_ENDED se envía.

.return false;

}

[ break ; ]

DragEvent caso.ACTION\_DRAG\_ENTERED: {

/ / Aplica un tono verde a la vista. Return true; el valor de retorno se ignora.

v.setColorFilter (color.GREEN (VERDE)

/ / Invalidar el fin de forzar un rediseño en el tinte nueva

v.invalidate ();

return true;

[ break ; ]

DragEvent caso.ACTION\_DRAG\_LOCATION:

/ / Ignorar el evento

return true;

[ break ; ]

DragEvent caso.ACTION\_DRAG\_EXITED:

/ / Re-establece el tono de color a azul. Devuelve true; el valor de retorno se ignora.

v.setColorFilter (color.(AZUL)

/ / Invalidar el fin de forzar un rediseño en el tinte nueva

v.invalidate ();

return true;

[ break ; ]

DragEvent caso.ACTION\_DROP:

/ / Obtiene el elemento que contiene los datos arrastrados

ClipData.. Tema event.getClipData = () getItemAt (0);

/ / Obtiene los datos de texto del elemento.

dragData item.getText = ();

/ / Muestra un mensaje que contiene los datos arrastrados.

Toast.makeText (this, "Arrastrado de datos es" + dragData, pan tostado.LENGTH\_LONG);

/ / Desactiva los tonos de color

v.clearColorFilter ();

/ / Invalida el fin de forzar un redibujo

v.invalidate ();

return true; DragEvent.getResult () devolverá true.

return true;

[ break ; ]

DragEvent caso.ACTION\_DRAG\_ENDED:

/ / Apaga cualquier color tintado

v.clearColorFilter ();

/ / Invalida el fin de forzar un redibujo

v.invalidate ();

/ / ¿Un getResult (), y muestra lo que sucedió.

if (event.getResult ()) {

Toast.makeText (this, "La caída fue manejado.", Pan tostado.LENGTH\_LONG);

} else {

Toast.makeText (this, "La caída no funcionó.", Pan tostado.LENGTH\_LONG);

};

/ / Devuelve true, el valor se ignora.

return true;

[ break ; ]

/ / Un tipo de acción se desconoce fue recibido.

default:

Log.e ("Ejemplo DragDrop", "tipo de acción desconocido recibida por OnDragListener.");

[ break ; ]

};

};

};

**Estilos y Temas**

**En este documento**

1. [Definición de estilos](http://developer.android.com/guide/topics/ui/themes.html#DefiningStyles)
   1. [Herencia.](http://developer.android.com/guide/topics/ui/themes.html#Inheritance)
   2. [Propiedades de estilo de](http://developer.android.com/guide/topics/ui/themes.html#Properties)
2. [Aplicación de estilos y temas para la interfaz de usuario](http://developer.android.com/guide/topics/ui/themes.html#ApplyingStyles)
   1. [Aplicar un estilo a una vista](http://developer.android.com/guide/topics/ui/themes.html#ApplyAStyle)
   2. [Aplicar un tema a una actividad o de la aplicación](http://developer.android.com/guide/topics/ui/themes.html#ApplyATheme)
   3. [Seleccione un tema basado en versión de la plataforma](http://developer.android.com/guide/topics/ui/themes.html#SelectATheme)
3. [Uso de estilos y temas de la Plataforma](http://developer.android.com/guide/topics/ui/themes.html#PlatformStyles)

**Véase también**

1. [Estilo y el tema de Recursos](http://developer.android.com/guide/topics/resources/available-resources.html#stylesandthemes)
2. [R.style](http://developer.android.com/reference/android/R.style.html) de estilos y temas para Android
3. [R.attr](http://developer.android.com/reference/android/R.attr.html) para todos los atributos de estilo

Un **estilo** es una colección de propiedades que especifican el aspecto y el formato de una [View](http://developer.android.com/reference/android/view/View.html) o una ventana. Un estilo puede especificar las propiedades tales como la altura, el relleno, color de fuente, tamaño de fuente, color de fondo, y mucho más. Un estilo se define en un recurso XML que es independiente del código XML que especifica la disposición.

Estilos de la cuota de Android una filosofía similar a las hojas de estilo en cascada en diseño web que le permiten separar el diseño del contenido.

Por ejemplo, mediante el uso de un estilo, usted puede tomar este esquema XML:

<TextView

android: layout\_width = "fill\_parent"

android: layout\_height = "wrap\_content"

android: textColor = "# 00FF00"

android: tipo de letra = "monoespacio"

android: text = "@ cadena / hola" />

Y convertirlo en esto:

<TextView

style = "@ estilo / CodeFont"

android: text = "@ cadena / hola" />

Todos los atributos relacionados con el estilo se han eliminado de la disposición XML y poner en una definición de estilo llamado CodeFont , que luego se aplica con el style de atributos. Usted verá la definición de este estilo en la sección siguiente.

Un **tema** es un estilo que se aplica a toda una [Activity](http://developer.android.com/reference/android/app/Activity.html) o de la aplicación, en lugar de un individuo [View](http://developer.android.com/reference/android/view/View.html) (como en el ejemplo anterior). Cuando un estilo se aplica como un tema, todas las vistas en la actividad o la aplicación se aplicarán a cada propiedad de estilo que lo soporta. Por ejemplo, puede aplicar el mismo CodeFont estilo como un tema para realizar una actividad y luego todo el texto dentro de esa actividad habrá fuente de espacio sencillo verde.

**Definición de estilos**

Para crear un conjunto de estilos, guardar un archivo XML en la res/values/ de directorio de su proyecto. El nombre del archivo XML es arbitraria, sino que debe utilizar el .xml de extensión y se guardarán en la res/values/ la carpeta.

El nodo raíz del archivo XML debe ser <resources> .

Para cada estilo que desea crear, agregar un <style> elemento para el archivo con un name que identifica el estilo (este atributo es obligatorio). A continuación, agregue un <item> elemento para cada propiedad de ese estilo, con un name que declara la propiedad de estilo y un valor para ir con ella (este atributo es obligatorio). El valor de la <item> puede ser una cadena de palabras clave, un color hexadecimal, una referencia a otro tipo de recurso, u otro valor en función de la propiedad de estilo. He aquí un ejemplo de archivo con un estilo único:

<?xml version="1.0" encoding="utf-8"?>

<resources>

<Nombre del estilo = "CodeFont" padre = "@ android: estilo / TextAppearance.MEDIA

<item fill\_parent name="android:layout\_width"> </ item>

<item wrap\_content name="android:layout\_height"> </ item>

<item name="android:textColor"> # 00FF00 </ item>

<item name="android:typeface"> monoespacio </ item>

</style>

</ Recursos>

Cada niño de la <resources> elemento se convierte en un objeto de recursos de aplicación en tiempo de compilación, que se puede hacer referencia por el valor en el <style> elemento name atributo. Este estilo de ejemplo, se puede hacer referencia a un diseño de XML como @style/CodeFont (como se demuestra en la introducción).

El parent atributo en el <style> elemento es opcional y especifica el identificador de recursos de otro estilo de la que este estilo debe heredar propiedades. A continuación, puede modificar las propiedades de estilo heredada, si así lo desea.

Recuerde, un estilo que desea utilizar como una actividad o un tema de aplicación se define en XML exactamente lo mismo que un estilo para una vista. Un estilo como el que se ha definido anteriormente puede ser aplicado como un estilo de una vista única o como un tema de una actividad completa o aplicación. Cómo aplicar un estilo para una vista única o como un tema de aplicación se discute más adelante.

**Herencia.**

El parent atributo en el <style> elemento le permite especificar un estilo de la que su estilo debe heredar propiedades. Usted puede usar esto para heredar las propiedades de un estilo existente y luego definir sólo las propiedades que desea cambiar o agregar. Usted puede heredar de estilos que ha creado usted mismo o de estilos que están integradas en la plataforma. (Véase [Utilización de estilos y temas de la plataforma](http://developer.android.com/guide/topics/ui/themes.html#PlatformStyles) , a continuación, para obtener información sobre la herencia de estilos definidos por la plataforma Android.) Por ejemplo, se puede heredar la apariencia de la plataforma Android por defecto de texto y luego modificarlo:

<style name="GreenText" parent="@android:style/TextAppearance">

<item name="android:textColor"> # 00FF00 </ item>

</style>

Si desea heredar de estilos que ya se ha definido, *no* tiene que utilizar la parent de atributos. En cambio, sólo un prefijo al nombre del estilo que desea heredar el nombre de su nuevo estilo, separados por un punto. Por ejemplo, para crear un nuevo estilo que hereda la CodeFont estilo definido anteriormente, pero que el color rojo, que pueden crear el nuevo estilo de esta manera:

El nombre de <style = "CodeFont.ROJO

<item name="android:textColor"> # FF0000 </ item>

</style>

Tenga en cuenta que no hay parent atributo en el <style> etiqueta, sino porque el name atributo se inicia con la CodeFont nombre del estilo (que es un estilo que ha creado), este estilo hereda todas las propiedades de estilo de ese estilo. Este estilo anula la android:textColor la propiedad para que el texto rojo. Puede hacer referencia a este nuevo estilo como @style/CodeFont.ROJO

Usted puede seguir heredando como éste tantas veces como desee, por el encadenamiento de los nombres con los períodos. Por ejemplo, se puede extender CodeFont.Red para ser más grande, con:

El nombre de <style = "CodeFont.ROJOgr

<item name="android:textSize"> 30sp </ item>

</style>

Esto se hereda de ambos CodeFont y CodeFont.Red estilos, a continuación, añade el android:textSize propiedad.

**Nota:** Esta técnica de la herencia, mediante el encadenamiento sólo los nombres de las obras de estilos definidos por sus propios recursos. Usted no puede heredar Android incorporado en los estilos de esta manera. Para hacer referencia a un estilo incorporado, como [TextAppearance](http://developer.android.com/reference/android/R.style.html#TextAppearance) , debe utilizar la parent de atributos.

**Propiedades de estilo de**

Ahora que usted entiende cómo se define un estilo, usted necesita saber qué clase de propiedades de estilo definidas por el <item> elemento están disponibles. Probablemente usted está familiarizado con algunos de ellos ya, como [layout\_width](http://developer.android.com/reference/android/R.attr.html#layout_width) y [textColor](http://developer.android.com/reference/android/R.attr.html#textColor) . Por supuesto, hay muchas más propiedades de estilo que puede utilizar.

El mejor lugar para encontrar propiedades que se aplican a un determinado [View](http://developer.android.com/reference/android/view/View.html) es la referencia de clase correspondiente, el cual muestra todos los atributos XML compatibles. Por ejemplo, todos los atributos enumerados en la tabla de [TextView atributos XML](http://developer.android.com/reference/android/widget/TextView.html#lattrs) se puede utilizar en una definición de estilo para una [TextView](http://developer.android.com/reference/android/widget/TextView.html) elemento (o una de sus subclases). Uno de los atributos que figuran en la referencia es [android:inputType](http://developer.android.com/reference/android/widget/TextView.html#attr_android:inputType) , por lo que normalmente pueden hacer que el [android:inputType](http://developer.android.com/reference/android/widget/TextView.html#attr_android:inputType) atributo en un <EditText> elemento, de esta manera:

<EditarTexto

android: InputType = "número"

... />

En su lugar, puede crear un estilo para el [EditText](http://developer.android.com/reference/android/widget/EditText.html) elemento que incluye esta propiedad:

<style name="Numbers">

<item número name="android:inputType"> </ item>

...

</style>

Así que el XML para el diseño ahora puede implementar este estilo:

<EditarTexto

style = "@ estilo / Números"

... />

Este sencillo ejemplo puede parecer más trabajo, pero cuando se agrega más propiedades de estilo y el factor de la capacidad de volver a utilizar el estilo en varios lugares, el pago de compensación puede ser enorme.

Para una referencia de todas las propiedades de estilo disponibles, consulte la [R.attr](http://developer.android.com/reference/android/R.attr.html) de referencia. Tenga en cuenta que todos los objetos de Vista no acepta todos los atributos de estilo mismas, por lo que normalmente se debe hacer referencia específica a la [View](http://developer.android.com/reference/android/view/View.html) de clases para las propiedades de estilo admitidas. Sin embargo, si se aplica un estilo a una vista que no es compatible con todas las propiedades de estilo, la vista se aplicará sólo aquellas propiedades que son compatibles y simplemente ignorar a los demás.

Algunas propiedades de estilo, sin embargo, no son compatibles con cualquier elemento View y sólo puede ser aplicado como un tema. Estas propiedades de estilo se aplica a toda la ventana y no a cualquier tipo de Vista. Por ejemplo, las propiedades de estilo para un tema se puede ocultar el título de la aplicación, ocultar la barra de estado, o cambiar el fondo de la ventana. Este tipo de propiedades de estilo no pertenece a ningún objeto View. Para descubrir estas propiedades de estilo sólo el tema-, mira el [R.attr](http://developer.android.com/reference/android/R.attr.html) de referencia para los atributos que comienzan con window . Por ejemplo, windowNoTitle y windowBackground son propiedades de estilo que son efectivos sólo cuando se aplica el estilo como un tema a una actividad o de la aplicación. Consulte la sección siguiente para obtener información sobre la aplicación de un estilo como un tema.

**Nota:** No se olvide de prefijo de los nombres de las propiedades en cada <item> elemento con el android: espacio de nombres. Por ejemplo: <item name="android:inputType"> .

**Aplicación de estilos y temas para la interfaz de usuario**

Hay dos maneras de establecer un estilo:

* Para una vista individual, añadiendo el style atributo a un elemento de Vista en el XML para su diseño.
* O, para toda una actividad o de la aplicación, mediante la adición del android:theme atribuyen a los <activity> o <application> de elementos en el manifiesto de Android.

Cuando se aplica un estilo a una sola [View](http://developer.android.com/reference/android/view/View.html) en el diseño, las propiedades definidas por el estilo se aplica sólo a la [View](http://developer.android.com/reference/android/view/View.html) . Si un estilo se aplica a un [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html) , los niños [View](http://developer.android.com/reference/android/view/View.html) los elementos **no** heredarán el estilo de las propiedades de sólo el elemento al que se aplica directamente el estilo se aplicará a sus propiedades. Sin embargo, *puede* aplicar un estilo de modo que se aplique a todos [View](http://developer.android.com/reference/android/view/View.html) elementos, aplicando el estilo como un tema.

Para aplicar una definición de estilo como un tema, debe aplicar el estilo a una [Activity](http://developer.android.com/reference/android/app/Activity.html) o de la aplicación en el manifiesto de Android. Al hacerlo, cada [View](http://developer.android.com/reference/android/view/View.html) dentro de la actividad o la aplicación se aplicará cada propiedad que lo apoya. Por ejemplo, si se aplica el CodeFont estilo de los ejemplos anteriores para una actividad, entonces todos los elementos en que el fomento de las propiedades de estilo de texto va a aplicar. Cualquier opinión que no es compatible con las propiedades de las ignora. Si una vista sólo es compatible con algunas de las propiedades, entonces se aplica sólo esas propiedades.

**Aplicar un estilo a una vista**

Aquí está cómo configurar un estilo para una vista en el diseño de XML:

<TextView

style = "@ estilo / CodeFont"

android: text = "@ cadena / hola" />

Ahora bien, este TextView será de estilo definidas por el estilo con nombre CodeFont . (Véase el ejemplo anterior, en [Definición de estilos](http://developer.android.com/guide/topics/ui/themes.html#DefiningStyles) .)

**Nota:** El style atributo *no* utilizar el android: prefijo de espacio de nombres.

**Aplicar un tema a una actividad o de la aplicación**

Para establecer un tema para todas las actividades de su aplicación, abra el AndroidManifest.xml archivo y editar el <application> etiqueta para incluir el android:theme atributo con el nombre del estilo. Por ejemplo:

aplicación> android:theme="@style/CustomTheme">

Si usted desea un tema aplicado a una sola actividad en la aplicación, a continuación, agregue el android:theme atributo a la <activity> etiqueta en su lugar.

Al igual que Android ofrece otros elementos de construcción en los recursos, hay muchos temas predefinidos que se pueden utilizar, para evitar escribir su propia cuenta. Por ejemplo, puede utilizar el Dialog el tema y hacer que su actividad aparece como un cuadro de diálogo:

<Actividad de Android: tema = "@ android: estilo / tema.Diálogo

O si desea que el fondo sea transparente, translúcido utilizar el tema:

<Actividad de Android: tema = "@ android: estilo / tema.Transl&#250;cido

Si te gusta un tema, pero quieren modificar, sólo tiene que añadir el tema como el parent de su tema personalizado. Por ejemplo, puede modificar el tema de la luz tradicional de usar su propio color de esta manera:

<Color name="custom\_theme\_color"> # b0b0ff </ color>

<Nombre del estilo = "CustomTheme" padre = "android: Tema.LUZ

<item name="android:windowBackground"> ​​@ color / custom\_theme\_color </ item>

<item name="android:colorBackground"> ​​@ color / custom\_theme\_color </ item>

</style>

(Tenga en cuenta que el color debe suministrarse como un recurso separado aquí, porque el android:windowBackground atributo sólo es compatible con una referencia a otro recurso, a diferencia de android:colorBackground ., no se puede dar un color literal)

Ahora usa CustomTheme vez de Theme.Light en el interior del Manifiesto de Android:

<activity android:theme="@style/CustomTheme">

**Seleccione un tema basado en versión de la plataforma**

Las nuevas versiones de Android tienen temas adicionales disponibles para las aplicaciones, y es posible que desee utilizar estos mientras se ejecuta en estas plataformas sin dejar de ser compatible con versiones anteriores. Usted puede lograr esto a través de un tema personalizado que utiliza la selección de recursos para cambiar entre los diferentes temas para padres, basados ​​en la versión de la plataforma.

Por ejemplo, aquí está la declaración de un tema personalizado, que es sencillamente el tema de las plataformas estándar de la luz. Esto iría en un archivo XML con res/values (por lo general res/values/styles.xml ):

<Nombre del estilo = "LightThemeSelector" padre = "android: Tema.LUZ

...

</style>

Para que este tema utilizar el tema más reciente holográfica cuando la aplicación se ejecuta en Android 3.0 (Nivel API 11) o superior, usted puede colocar una declaración alternativa para el tema en un archivo XML en res/values-v11 , pero que el tema de los padres el tema holográfica:

<Nombre del estilo = "LightThemeSelector" padre = "android: Tema.Holo!LUZ

...

</style>

Ahora usa este tema como lo haría con cualquier otro, y su solicitud pasará automáticamente a la temática holográfica si se ejecuta en Android 3.0 o superior.

Una lista de los atributos estándar que se pueden utilizar en los temas se puede encontrar en [R.styleable.](http://developer.android.com/reference/android/R.styleable.html#Theme)Tema

Para obtener más información acerca de la provisión de recursos alternativos, como los temas y diseños, basados ​​en la versión de la plataforma u otras configuraciones del dispositivo, consulte el [Abastecimiento de recursos](http://developer.android.com/guide/topics/resources/providing-resources.html) documento.

**Uso de estilos y temas de la Plataforma**

La plataforma Android ofrece una gran colección de estilos y temas que se pueden utilizar en sus aplicaciones. Usted puede encontrar una referencia de todos los estilos disponibles en el [R.style](http://developer.android.com/reference/android/R.style.html) clase. Para utilizar los estilos figuran en esta lista, vuelva a colocar todos los caracteres de subrayado en el nombre del estilo con un punto. Por ejemplo, se puede aplicar el [Theme\_NoTitleBar](http://developer.android.com/reference/android/R.style.html#Theme_NoTitleBar) tema con "@android:style/Theme.NoTitleBar" .

El [R.style](http://developer.android.com/reference/android/R.style.html) de referencia, sin embargo, no está bien documentada y no bien se describen los estilos, por lo que ver el código fuente real de estos estilos y temas le dará una mejor comprensión de lo que las propiedades de estilo de cada uno ofrece. Para una mejor referencia a los estilos de Android y los temas, vea el siguiente código fuente:

* [Android Estilos (styles.xml)](http://www.google.com/url?q=http%3A%2F%2Fandroid.git.kernel.org%2F%3Fp%3Dplatform%2Fframeworks%2Fbase.git%3Ba%3Dblob%3Bf%3Dcore%2Fres%2Fres%2Fvalues%2Fstyles.xml%3Bh%3Dd7b654e49809cb97a35682754b1394af5c8bc88b%3Bhb%3DHEAD&sa=D&sntz=1&usg=AFQjCNEnT-gKC2Jnuv1-GgNLg_IeDdcZvg)
* [Android Temas (themes.xml)](http://www.google.com/url?q=http%3A%2F%2Fandroid.git.kernel.org%2F%3Fp%3Dplatform%2Fframeworks%2Fbase.git%3Ba%3Dblob%3Bf%3Dcore%2Fres%2Fres%2Fvalues%2Fthemes.xml%3Bh%3D6b3d7407d1c895a3c297e60d5beac98e2d34c271%3Bhb%3DHEAD&sa=D&sntz=1&usg=AFQjCNFfibMG4L5HuiI9XJIt1RJTTPVahA)

Estos archivos le ayudará a aprender a través del ejemplo. Por ejemplo, en el código fuente de Android temas, encontrará una declaración de <style name="Theme.Diálogo En esta definición, podrá ver todas las propiedades que se utilizan para cuadros de diálogo de estilo que se utilizan en el marco de Android.

Para obtener más información acerca de la sintaxis utilizada para crear estilos de XML, vea [Tipos de recursos disponibles: Estilo y temas](http://developer.android.com/guide/topics/resources/available-resources.html#stylesandthemes) .

Para una referencia de los atributos de estilo disponibles que se pueden utilizar para definir un estilo o un tema (por ejemplo, "windowBackground" o "textAppearance"), ver [R.attr](http://developer.android.com/reference/android/R.attr.html) o de la clase View respectivo para el que está creando un estilo.

**Componentes personalizados**

**En este documento**

1. [El enfoque básico](http://developer.android.com/guide/topics/ui/custom-components.html#basic)
2. [Componentes totalmente personalizados](http://developer.android.com/guide/topics/ui/custom-components.html#custom)
3. [Los controles compuestos](http://developer.android.com/guide/topics/ui/custom-components.html#compound)
4. [Modificación de un tipo de vista existente](http://developer.android.com/guide/topics/ui/custom-components.html#modifying)

Android ofrece un modelo en componentes sofisticados y potentes para la construcción de la interfaz de usuario, basada en las clases de diseño fundamentales: [View](http://developer.android.com/reference/android/view/View.html) y [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html) . Para empezar, la plataforma incluye una variedad de Vista pre-compilados y subclases ViewGroup - llamadas widgets y diseños, respectivamente - que se puede utilizar para construir la interfaz de usuario.

Una lista parcial de widgets disponibles incluye [Button](http://developer.android.com/reference/android/widget/Button.html) , [TextView](http://developer.android.com/reference/android/widget/TextView.html) , [EditText](http://developer.android.com/reference/android/widget/EditText.html) , [ListView](http://developer.android.com/reference/android/widget/ListView.html) , [CheckBox](http://developer.android.com/reference/android/widget/CheckBox.html) , [RadioButton](http://developer.android.com/reference/android/widget/RadioButton.html) , [Gallery](http://developer.android.com/reference/android/widget/Gallery.html) , [Spinner](http://developer.android.com/reference/android/widget/Spinner.html) , y el más especial de uso [AutoCompleteTextView](http://developer.android.com/reference/android/widget/AutoCompleteTextView.html) , [ImageSwitcher](http://developer.android.com/reference/android/widget/ImageSwitcher.html) y [TextSwitcher](http://developer.android.com/reference/android/widget/TextSwitcher.html) .

Entre los diseños disponibles son [LinearLayout](http://developer.android.com/reference/android/widget/LinearLayout.html) , [FrameLayout](http://developer.android.com/reference/android/widget/FrameLayout.html) , [RelativeLayout](http://developer.android.com/reference/android/widget/RelativeLayout.html) , y otros. Para más ejemplos, vea [los objetos comunes de diseño](http://developer.android.com/guide/topics/ui/layout-objects.html) .

Si ninguno de los widgets creados previamente o diseños se adapte a sus necesidades, puede crear una subclase propia opinión. Si usted sólo tendrá que hacer pequeños ajustes a un widget o el diseño, simplemente puede subclase el widget o la disposición e invalidar sus métodos.

Creación de sus propias subclases Vista le ofrece un control preciso sobre el aspecto y la función de un elemento de la pantalla. Para dar una idea del control que se obtiene con vistas personalizadas, he aquí algunos ejemplos de lo que podía hacer con ellos:

* Se puede crear una completamente personalizado-renderizados tipo de vista, por ejemplo un "control de volumen" mando hizo uso de gráficos en 2D, y que se asemeja a un control electrónico analógico.
* Se puede combinar un grupo de componentes de Vista en un único componente nuevo, tal vez para hacer algo parecido a un cuadro combinado (una combinación de lista desplegable y el campo de entrada de texto libre), un control de doble panel de selección (un panel de la izquierda y la derecha con una lista en la cada uno donde se puede volver a asignar el elemento que es en el que la lista), y así sucesivamente.
* Se puede reemplazar la forma en que un componente EditarTexto se representa en la pantalla (el [Bloc de notas Tutorial](http://developer.android.com/resources/samples/NotePad/index.html) utiliza esto para un buen efecto, para crear una página forrada-bloc de notas).
* Usted puede capturar otros eventos como pulsaciones de teclas y controlar de alguna manera personalizada (por ejemplo, para un juego).

En las secciones siguientes se explica cómo crear vistas personalizadas y utilizarlas en su aplicación. Para obtener información de referencia detallada, consulte la [View](http://developer.android.com/reference/android/view/View.html) de clases.

**El enfoque básico**

Aquí está una descripción de alto nivel de lo que necesita saber para iniciarse en la creación de sus propios componentes de Vista:

1. Ampliar un [View](http://developer.android.com/reference/android/view/View.html) clase o subclase con su propia clase.
2. Reemplazar algunos de los métodos de la superclase. Los métodos de la superclase para anular comienza con ' on ', por ejemplo, [onDraw()](http://developer.android.com/reference/android/view/View.html#onDraw%28android.graphics.Canvas%29) , [onMeasure()](http://developer.android.com/reference/android/view/View.html#onMeasure%28int,%20int%29) , y [onKeyDown()](http://developer.android.com/reference/android/view/View.html#onKeyDown%28int,%20android.view.KeyEvent%29) . Esto es similar a los on... los acontecimientos en [Activity](http://developer.android.com/reference/android/app/Activity.html) o [ListActivity](http://developer.android.com/reference/android/app/ListActivity.html) que reemplazar la funcionalidad de los ganchos del ciclo de vida y de otra índole.
3. Utilice la clase nueva extensión. Una vez completado, su clase nueva extensión se puede utilizar en lugar de la vista en la que se basa.

**Sugerencia:** las clases de extensión se puede definir como las clases internas dentro de las actividades que los utilizan. Esto es útil, ya que controla el acceso a ellos, pero no es necesario (tal vez desea crear una nueva vista pública para un uso más amplio en su aplicación).

**Componentes totalmente personalizados**

Componentes totalmente personalizados se pueden utilizar para crear componentes gráficos que aparecen como desee. Tal vez un medidor de VU gráfica que se ve como un indicador analógico de edad, o una vista de texto sing-a-tiempo donde un rebote la bola se mueve a lo largo de las palabras para que pueda cantar junto con una máquina de karaoke. De cualquier manera, usted quiere algo que los componentes integrados simplemente no va a hacer, no importa cómo se combinan.

Afortunadamente, usted puede crear fácilmente los componentes que se ven y se comportan de la manera que desea, tal vez sólo limitado por tu imaginación, el tamaño de la pantalla, y el poder de procesamiento disponible (recordemos que en última instancia su aplicación puede tener para funcionar en algo con mucho menos poder que su estación de trabajo).

Para crear un componente totalmente personalizado:

1. El punto de vista más genérico, se puede ampliar es, como era de esperar, [View](http://developer.android.com/reference/android/view/View.html) , por lo que normalmente se iniciará mediante la ampliación de este para crear el componente nuevo súper.
2. Puede proporcionar un constructor que puede tomar los atributos y los parámetros del XML, y también se puede consumir sus propios atributos y parámetros de este tipo (tal vez el color y el rango del indicador de volumen, o la anchura y la amortiguación de la aguja, etc)
3. Es probable que desee crear sus propios detectores de eventos de acceso de propiedad y los modificadores de conducta, y posiblemente más sofisticados en su clase de componente, así.
4. Es casi seguro que desea reemplazar onMeasure() y también son propensos a necesitar para anular onDraw() si desea que el componente a mostrar algo. Mientras tanto tienen un comportamiento por defecto, el valor predeterminado onDraw() no hará nada, y el valor predeterminado onMeasure() siempre se establece un tamaño de 100x100 - que probablemente no es lo que quieres.
5. Otro on... métodos también pueden ser anulados si es necesario.

**Extender onDraw() y onMeasure()**

El onDraw() método que ofrece un [Canvas](http://developer.android.com/reference/android/graphics/Canvas.html) sobre el que se puede implementar todo lo que quieras: gráficos 2D, otros componentes estándar o personalizados, texto con estilo, o cualquier cosa que se pueda imaginar.

**Nota:** Esto no se aplica a los gráficos 3D. Si desea utilizar gráficos en 3D, debe extender [SurfaceView](http://developer.android.com/reference/android/view/SurfaceView.html) lugar de la vista, y sacar de un subproceso independiente. Vea el ejemplo GLSurfaceViewActivity para más detalles.

onMeasure() es un poco más complicado. onMeasure() es una pieza crítica del contrato de prestación de entre el componente y su contenedor. onMeasure() debe reemplazarse para informar con eficiencia y precisión de las mediciones de las partes contenidas. Esto se hace ligeramente más complejo por los requisitos de los límites de la matriz (que se pasan a la onMeasure() método) y por el requisito de llamar a la setMeasuredDimension() método con el ancho y la altura mide una vez que han sido calculados. Si usted no puede llamar a este método desde un anulada onMeasure() método, el resultado será una excepción en tiempo de medición.

A un alto nivel, la aplicación de onMeasure() se ve algo como esto:

1. La anulado onMeasure() se llama al método con las especificaciones medir el ancho y la altura ( widthMeasureSpec y heightMeasureSpec parámetros, ambos son códigos de números enteros que representan las dimensiones), que deben ser tratados como los requisitos para las restricciones de la anchura y la medición de la altura que debe producir. Una referencia completa a la clase de restricciones de estas especificaciones pueden requerir se puede encontrar en la documentación de referencia en [View.onMeasure(int, int)](http://developer.android.com/reference/android/view/View.html#onMeasure%28int,%20int%29) (esta documentación de referencia hace un buen trabajo de explicar la operación de medición de conjunto, así).
2. Su componente onMeasure() método debería calcular una medida del ancho y la altura que se requiere para hacer que el componente. Se debe tratar de mantenerse dentro de las especificaciones aprobadas en, aunque se puede optar por superar (en este caso, los padres pueden elegir qué hacer, incluyendo el recorte, el desplazamiento, una excepción, o pidiendo a la onMeasure() para volver a intentarlo, tal vez con las especificaciones de medición diferentes).
3. Una vez que la anchura y la altura se calcula, el setMeasuredDimension(int width, int height) método debe ser llamado con las medidas calculadas. De no hacerlo, dará lugar a una excepción que es lanzada.

He aquí un resumen de algunos de los otros métodos estándar que se llama el marco de puntos de vista:

| **Categoría** | **métod** | **Descripción** |
| --- | --- | --- |
| Creaci | Constructores | Hay una forma del constructor que se llama cuando se crea la vista de código y una forma que se llama cuando la vista se infla de un archivo de diseño. La segunda forma debe analizar y aplicar los atributos definidos en el archivo de diseño. |
| [onFinishInflate()](http://developer.android.com/reference/android/view/View.html#onFinishInflate%28%29) | Se llama después de un punto de vista y todos sus hijos se ha inflado a partir de XML. |
| Disposición | [onMeasure(int, int)](http://developer.android.com/reference/android/view/View.html#onMeasure%28int,%20int%29) | Llamado para determinar los requisitos de tamaño para este punto de vista y todos sus hijos. |
| [onLayout(boolean, int, int, int, int)](http://developer.android.com/reference/android/view/View.html#onLayout%28boolean,%20int,%20int,%20int,%20int%29) | Se le llama cuando este punto de vista debe asignar un tamaño y posición de todos sus hijos. |
| [onSizeChanged(int, int, int, int)](http://developer.android.com/reference/android/view/View.html#onSizeChanged%28int,%20int,%20int,%20int%29) | Se le llama cuando el tamaño de este punto de vista ha cambiado. |
| Dibujos | [onDraw(Canvas)](http://developer.android.com/reference/android/view/View.html#onDraw%28android.graphics.Canvas%29) | Se le llama cuando el punto de vista debe representar su contenido. |
| Procesamiento de eventos | [onKeyDown(int, KeyEvent)](http://developer.android.com/reference/android/view/View.html#onKeyDown%28int,%20android.view.KeyEvent%29) | Se le llama cuando un evento de la llave se produce. |
| [onKeyUp(int, KeyEvent)](http://developer.android.com/reference/android/view/View.html#onKeyUp%28int,%20android.view.KeyEvent%29) | Se le llama cuando un evento de tecla de arriba se produce. |
| [onTrackballEvent(MotionEvent)](http://developer.android.com/reference/android/view/View.html#onTrackballEvent%28android.view.MotionEvent%29) | Se le llama cuando se mueve el trackball se produce. |
| [onTouchEvent(MotionEvent)](http://developer.android.com/reference/android/view/View.html#onTouchEvent%28android.view.MotionEvent%29) | Se le llama cuando una pantalla táctil de eventos de movimiento se produce. |
| Enfocar | [onFocusChanged(boolean, int, Rect)](http://developer.android.com/reference/android/view/View.html#onFocusChanged%28boolean,%20int,%20android.graphics.Rect%29) | Se le llama cuando el punto de vista gana o pierde el foco. |
| [onWindowFocusChanged(boolean)](http://developer.android.com/reference/android/view/View.html#onWindowFocusChanged%28boolean%29) | Se le llama cuando la ventana que contiene los logros de vista o pierde el foco. |
| Instalación, fijación, sujeción, colocación | [onAttachedToWindow()](http://developer.android.com/reference/android/view/View.html#onAttachedToWindow%28%29) | Se llama cuando la vista está unido a una ventana. |
| [onDetachedFromWindow()](http://developer.android.com/reference/android/view/View.html#onDetachedFromWindow%28%29) | Se le llama cuando la vista se separa de su ventana. |
| [onWindowVisibilityChanged(int)](http://developer.android.com/reference/android/view/View.html#onWindowVisibilityChanged%28int%29) | Se le llama cuando la visibilidad de la ventana que contiene el punto de vista ha cambiado. |

**Un ejemplo de vistas personalizadas**

La muestra CustomView en el [API Demos](http://developer.android.com/resources/samples/ApiDemos/index.html) un ejemplo de una vista personalizada. La vista personalizada se define en el [LabelView](http://developer.android.com/resources/samples/ApiDemos/src/com/example/android/apis/view/LabelView.html) clase.

La muestra LabelView demuestra una serie de aspectos diferentes de componentes personalizados:

* Ampliación de la clase Vista de un componente completamente personalizado.
* Parametrizada constructor que toma los parámetros de la inflación (ver los parámetros definidos en el XML). Algunas de ellas pasan a través de la superclase Vista, pero lo más importante, hay algunos atributos personalizados definidos y utilizados para LABELVIEW.
* Standard métodos públicos del tipo que se puede esperar para ver a un componente de la etiqueta, por ejemplo setText() , setTextSize() , setTextColor() y así sucesivamente.
* Un anulado onMeasure método para determinar y establecer el tamaño de representación del componente. (Tenga en cuenta que en LABELVIEW, el trabajo real es realizado por una empresa privada measureWidth() método.)
* Un anulado onDraw() método para dibujar la etiqueta sobre el lienzo siempre.

Puede ver algunos ejemplos de usos de la vista personalizada en LabelView [custom\_view\_1.xml](http://developer.android.com/resources/samples/ApiDemos/res/layout/custom_view_1.html) de las muestras. En particular, se puede ver una mezcla de ambos android: los parámetros de espacio de nombres y la costumbre app: los parámetros de espacio de nombres. Estas app: los parámetros son los personalizados que el LabelView reconoce y trabaja con, y se definen en una clase styleable interior dentro de las muestras de los recursos de I definición de la clase.

**Los controles compuestos**

Si no desea crear un componente totalmente personalizado, sino que se busca armar un componente reutilizable que consiste en un grupo de los controles existentes, a continuación, la creación de un componente compuesto (o Control Compuesto) podría encajar el proyecto de ley. En pocas palabras, este reúne a una serie de controles más atómico (o visitas) en un grupo lógico de los elementos que pueden ser tratadas como una sola cosa. Por ejemplo, un cuadro combinado se puede considerar como una combinación de un campo de línea EditarTexto único y un botón adyacente con una PopupList adjunta. Si pulsa el botón y seleccionar algo de la lista, rellena el campo EditarTexto, pero el usuario también puede escribir algo directamente en el EditarTexto si así lo prefieren.

En Android, en realidad hay dos otros puntos de vista fácilmente disponibles para hacer esto: [Spinner](http://developer.android.com/reference/android/widget/Spinner.html) y [AutoCompleteTextView](http://developer.android.com/reference/android/widget/AutoCompleteTextView.html) , pero sin tener en cuenta, el concepto de un cuadro combinado hace una fácil de entender el ejemplo.

Para crear un componente compuesto:

1. El punto de partida habitual es un diseño de algún tipo, por lo que crear una clase que se extiende a un diseño. Quizás en el caso de un cuadro combinado que podría utilizar un LinearLayout con orientación horizontal. Recuerde que los otros diseños se pueden anidar en el interior, por lo que el componente de compuesto puede ser arbitrariamente compleja y estructurada. Tenga en cuenta que al igual que con una actividad, puede utilizar la memoria declarativa (basado en XML) aproximación a la creación de los componentes que contiene, o puede anidar ellos mediante programación desde el código.
2. En el constructor de la nueva clase, tomar todas los parámetros de la superclase espera, y pasar a través del constructor de la superclase en primer lugar. A continuación, puede configurar los puntos de vista de otros para usar dentro de su nuevo componente, que es donde se crearía el campo de la EditarTexto y PopupList. Tenga en cuenta que también puede introducir sus propios atributos y parámetros en el XML que se puede sacar y utilizada por el constructor.
3. También se pueden crear detectores de eventos que sus puntos de vista contenidos pueden generar, por ejemplo, un método de escucha para el elemento de la lista Haga clic en escucha para actualizar el contenido de la EditarTexto si una lista de selección que se haga.
4. También puede crear sus propias propiedades y modificadores de acceso a, por ejemplo, permitir que el valor EditarTexto que se estableció inicialmente en el componente y la consulta de su contenido cuando sea necesario.
5. En el caso de la ampliación de un diseño, no es necesario para anular el onDraw() y onMeasure() ya que el diseño de métodos que tienen un comportamiento por defecto que es probable que funcione muy bien. Sin embargo, todavía se pueden reemplazar si es necesario.
6. Usted puede reemplazar otros on... métodos, como onKeyDown() , tal vez para elegir ciertos valores predeterminados de la lista desplegable de un cuadro combinado en una determinada tecla que se pulsa.

Para resumir, el uso de una disposición como la base para un control personalizado tiene un número de ventajas, incluyendo:

* Usted puede especificar el diseño utilizando los archivos XML declarativos al igual que con una pantalla de la actividad, o se pueden crear vistas mediante programación y anidar en el diseño de su código.
* El onDraw() y onMeasure() métodos (además de la mayor parte de la otra on... métodos) probablemente tendrá un comportamiento adecuado por lo que no tiene que reemplazar.
* Al final, puede muy rápidamente la construcción de puntos de vista de compuestos arbitrariamente complejas y re-usarlas como si fueran un solo componente.

**Ejemplos de controles compuestos**

En el proyecto de la API demos que viene con el SDK, hay dos ejemplos de lista: Ejemplo 4 y Ejemplo 6 en los dictámenes y listas de demostrar una SpeechView que se extiende LinearLayout para hacer un componente para la visualización de las cotizaciones del habla. Las clases correspondientes en el código de ejemplo son List4.java y List6.java .

**Modificación de un tipo de vista existente**

Hay una opción aún más fácil para crear una vista personalizada que es útil en determinadas circunstancias. Si hay un componente que es ya muy similar a lo que usted desea, usted simplemente puede ampliar ese componente y simplemente reemplazar el comportamiento que desea cambiar. Usted puede hacer todas las cosas que harías con un componente totalmente personalizado, pero comenzando con una clase más especializada en la jerarquía de Vista, también puede obtener una gran cantidad de conductas de forma gratuita que, probablemente, hace exactamente lo que quieres.

Por ejemplo, el SDK incluye una [aplicación Bloc de notas](http://developer.android.com/resources/samples/NotePad/index.html) en las muestras. Esto demuestra muchos aspectos de la utilización de la plataforma Android, entre ellos se extiende una vista EditarTexto para hacer una libreta forrada. Esto no es un ejemplo perfecto, y la API para hacer esto podría cambiar a partir de esta primera vista previa, pero sí demostrar los principios.

Si no lo ha hecho, la importación de la muestra el Bloc de notas en Eclipse (o tan sólo mirar en la fuente a través del enlace incluido). En particular, hacia la definición de MyEditText en el [NoteEditor.java](http://developer.android.com/resources/samples/NotePad/src/com/example/android/notepad/NoteEditor.html) archivo.

Algunos puntos a tener en cuenta aquí

1. **Definición**

La clase se define con la siguiente línea:  
public static class MyEditText extends EditText

* + Se define como una clase interna dentro de la NoteEditor actividad, pero es público, por lo que se puede acceder como NoteEditor.MyEditText desde fuera de la NoteEditor la clase si así lo desea.
  + Es static , lo que significa que no genera los llamados "métodos de síntesis" que le permiten acceder a datos de la clase padre, que a su vez significa que lo que realmente se comporta como una clase separada en lugar de algo muy relacionado con NoteEditor . Esta es una forma más limpia para crear clases internas si no es necesario el acceso al estado de la clase externa, mantiene la clase generada pequeño, y le permite ser utilizado con facilidad de otras clases.
  + Se extiende EditText , que es la vista que hemos decidido personalizar en este caso. Cuando hayamos terminado, la nueva clase será capaz de sustituir a una normal EditText vista.

1. **Clase de inicialización**

Como siempre, el super se llama en primer lugar. Además, este no es un constructor predeterminado, pero un uno con parámetros. El EditarTexto se crea con estos parámetros cuando se infla de un archivo de formato XML, por lo tanto, nuestro constructor necesita tanto tomarlos y pasarlos al constructor de la superclase también.

1. **Métodos sobrescritos**

En este ejemplo, sólo hay un método para reemplazar: onDraw() - pero fácilmente podría haber otros necesarios al crear sus propios componentes personalizados.

Para la muestra el Bloc de notas, haciendo caso omiso de la onDraw() método nos permite pintar las líneas azules en la EditText vista de lienzo (el lienzo se ha pasado al anular onDraw() método). El método super.onDraw () es llamado antes de que el método termina. El método de la superclase debe ser invocado, pero en este caso, lo hacemos al final después de haber pintado las líneas que desee incluir.

1. **Utilizar el componente personalizado**

Ahora tenemos nuestro componente personalizado, pero ¿cómo podemos usarlo? En el ejemplo el Bloc de Notas, el componente personalizado se utiliza directamente de la disposición declarativa, por lo que echar un vistazo a note\_editor.xml en la res/layout carpeta.

<Punto de vista

class = "com.android.notepad.NoteEditor $ MyEditText "

id = "+ id @ / nota"

android: layout\_width = "fill\_parent"

android: layout\_height = "fill\_parent"

android: background = "@ android: dibujable / vacío"

Android: padding = "10dip"

android: barras de desplazamiento = "vertical"

android: fadingEdge = "vertical" />

* + El componente personalizado se crea como un punto de vista genérico en el XML, y la clase se especifica mediante el paquete completo. Tenga en cuenta también que la clase interna que se hace referencia se define utilizando el NoteEditor$MyEditText notación que es una forma estándar para referirse a las clases internas en el lenguaje de programación Java.

Si el componente de vista personalizada no se define como una clase interna, entonces usted puede, alternativamente, declarar el componente de Vista con el nombre de elemento XML, y excluye la class atributo. Por ejemplo:

<Com.android.notepad.MyEditText

id = "+ id @ / nota"

... />

Observe que el MyEditText clase es ahora un archivo de clase independiente. Cuando la clase se anida en el NoteEditor clase, esta técnica no funcionará.

* + Los otros atributos y parámetros en la definición son los que pasan en el constructor de componentes personalizados, y luego pasa al constructor de EditarTexto, por lo que son los mismos parámetros que se utilizan para obtener una vista EditarTexto. Tenga en cuenta que es posible añadir sus propios parámetros, así, y vamos a tocar en esto de nuevo a continuación.

Y eso es todo lo que hay que hacer! Es cierto que este es un caso simple, pero ese es el punto - la creación de componentes personalizados es tan complicado como usted necesita que sea.

Uno de los componentes más sofisticados pueden anular aún más on... métodos e introducir algunos de sus propios métodos de ayuda, sustancialmente la personalización de sus propiedades y el comportamiento. El único límite es su imaginación y lo que necesita el componente que hacer.

**Accesibilidad**

**Temas**

1. [Hacer que las aplicaciones sean accesibles](http://developer.android.com/guide/topics/ui/accessibility/apps.html)
2. [Servicios de construcción de Accesibilidad](http://developer.android.com/guide/topics/ui/accessibility/services.html)

**Clases de clave**

1. [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html)
2. [AccessibilityService](http://developer.android.com/reference/android/accessibilityservice/AccessibilityService.html)

**Véase también**

1. [Implementación de Accesibilidad](http://developer.android.com/training/accessibility/index.html)

Muchos usuarios de Android tienen alguna discapacidad que les obligan a interactuar con sus dispositivos Android de diferentes maneras. Estos incluyen los usuarios que tienen discapacidades visuales, físicas o relacionadas con la edad que les impiden disfrutar plenamente de la vista o con una pantalla táctil.

Android proporciona las características de accesibilidad y servicios para ayudar a estos usuarios a navegar más fácilmente sus dispositivos, incluyendo comentarios de texto a voz, táctil, trackball y el D-pad de navegación que aumentan su experiencia. Los desarrolladores de Android aplicaciones pueden tomar ventaja de estos servicios para hacer que sus aplicaciones sean más accesibles y también crear sus propios servicios de accesibilidad.

Los temas siguientes muestran cómo utilizar el marco de Android para que las aplicaciones sean más accesibles.

[**Hacer que las aplicaciones sean accesibles**](http://developer.android.com/guide/topics/ui/accessibility/apps.html)

Prácticas de desarrollo y características de la API para asegurar que su aplicación es accesible a usuarios con discapacidad.

[**Servicios de construcción de Accesibilidad**](http://developer.android.com/guide/topics/ui/accessibility/service.html)

Cómo utilizar las funciones de la API para crear servicios que hacen otras aplicaciones más accesibles para los usuarios.

**Hacer que las aplicaciones sean accesibles**

**En este documento**

1. [Etiquetado de los elementos de interfaz de usuario](http://developer.android.com/guide/topics/ui/accessibility/apps.html#label-ui)
2. [Habilitación de enfoque de navegación](http://developer.android.com/guide/topics/ui/accessibility/apps.html#focus-nav)
   1. [Habilitación de vista el enfoque](http://developer.android.com/guide/topics/ui/accessibility/apps.html#focus-enable)
   2. [Control del orden de enfoque](http://developer.android.com/guide/topics/ui/accessibility/apps.html#focus-order)
3. [Edificio accesible Vistas personalizadas](http://developer.android.com/guide/topics/ui/accessibility/apps.html#custom-views)
   1. [Manejo de clics de dirección del controlador](http://developer.android.com/guide/topics/ui/accessibility/apps.html#directional-control)
   2. [Implementar métodos de accesibilidad API](http://developer.android.com/guide/topics/ui/accessibility/apps.html#accessibility-methods)
   3. [El envío de los eventos de accesibilidad](http://developer.android.com/guide/topics/ui/accessibility/apps.html#send-events)
   4. [Rellenar los eventos de accesibilidad](http://developer.android.com/guide/topics/ui/accessibility/apps.html#populate-events)
4. [Revisión de la Accesibilidad](http://developer.android.com/guide/topics/ui/accessibility/apps.html#test)
   1. [Prueba de respuesta audible](http://developer.android.com/guide/topics/ui/accessibility/apps.html#test-audibles)
   2. [Prueba de enfoque de navegación](http://developer.android.com/guide/topics/ui/accessibility/apps.html#test-navigation)

**Clases de clave**

1. [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html)
2. [AccessibilityNodeInfo](http://developer.android.com/reference/android/view/accessibility/AccessibilityNodeInfo.html)
3. [AccessibilityNodeInfoCompat](http://developer.android.com/reference/android/support/v4/view/accessibility/AccessibilityNodeInfoCompat.html)
4. [View.AccessibilityDelegate](http://developer.android.com/reference/android/view/View.AccessibilityDelegate.html)
5. [AccessibilityDelegateCompat](http://developer.android.com/reference/android/support/v4/view/AccessibilityDelegateCompat.html)

**Véase también**

1. [Implementación de Accesibilidad](http://developer.android.com/training/accessibility/index.html)
2. [Diseño de navegación efectiva](http://developer.android.com/training/design-navigation/index.html)
3. [Android Diseño](http://developer.android.com/design/index.html)

Las aplicaciones creadas para Android son accesibles para usuarios con discapacidades visuales, físicas o relacionadas con la edad cuando se activan las funciones de accesibilidad y servicios en un dispositivo. De forma predeterminada, estos servicios que su aplicación sea más accesible. Sin embargo, hay otras medidas que debe tomar para optimizar la accesibilidad de su aplicación y garantizar una experiencia agradable para todos los usuarios.

Asegurarse de que su aplicación es accesible a todos los usuarios es relativamente fácil, sobre todo cuando se utiliza el marco proporcionados por los componentes de la interfaz de usuario. Si sólo vas a usar estos componentes estándar para su aplicación, sólo hay unos pocos pasos necesarios para asegurarse de que su aplicación es accesible:

1. Etiqueta tu [ImageButton](http://developer.android.com/reference/android/widget/ImageButton.html) , [ImageView](http://developer.android.com/reference/android/widget/ImageView.html) , [EditText](http://developer.android.com/reference/android/widget/EditText.html) , [CheckBox](http://developer.android.com/reference/android/widget/CheckBox.html) y otros controles de interfaz de usuario con el [android:contentDescription](http://developer.android.com/reference/android/view/View.html#attr_android:contentDescription) atributo.
2. Haga todos los elementos de la interfaz de usuario accesible con un controlador de dirección, como una rueda de desplazamiento o D-Pad.
3. Pruebe su aplicación mediante la activación de los servicios de accesibilidad, como TalkBack y explorar mediante el tacto, e intente utilizar la aplicación utilizando sólo los controles de dirección.

Los desarrolladores que crean controles personalizados que se extienden desde la [View](http://developer.android.com/reference/android/view/View.html) de clases tiene algunas responsabilidades adicionales para asegurarse de que sus componentes son accesibles para los usuarios. Este documento también se explica cómo hacer la vista personalizada controla compatible con los servicios de accesibilidad.

**Etiquetado de los elementos de interfaz de usuario**

Muchos controles de interfaz de usuario se basan en señales visuales para informar a los usuarios de su significado. Por ejemplo, una aplicación para tomar notas puede usar un [ImageButton](http://developer.android.com/reference/android/widget/ImageButton.html) con una foto de un signo más para indicar que el usuario puede añadir una nueva nota. O bien, una [EditText](http://developer.android.com/reference/android/widget/EditText.html) componente puede tener una etiqueta que indica que cerca de su objetivo. Cuando un usuario con problemas de visión tiene acceso a su aplicación, estas claves visuales son a menudo inútiles.

Para proporcionar la información textual acerca de los controles de la interfaz (como alternativa a las señales visuales), utilice el [android:contentDescription](http://developer.android.com/reference/android/view/View.html#attr_android:contentDescription) atributo. El texto que usted proporciona en este atributo no es visible en la pantalla, pero si un usuario ha permitido a los servicios de accesibilidad que proporcionan avisos audibles, a continuación, la descripción de este atributo se leen en voz alta al usuario.

Ajuste el [android:contentDescription](http://developer.android.com/reference/android/view/View.html#attr_android:contentDescription) atributo para todos los [ImageButton](http://developer.android.com/reference/android/widget/ImageButton.html) , [ImageView](http://developer.android.com/reference/android/widget/ImageView.html) , [EditText](http://developer.android.com/reference/android/widget/EditText.html) , [CheckBox](http://developer.android.com/reference/android/widget/CheckBox.html) en la interfaz de usuario de su aplicación, y en los controles de entrada de otros que puedan requerir información adicional para los usuarios que no son capaces de verlo.

Por ejemplo, el siguiente [ImageButton](http://developer.android.com/reference/android/widget/ImageButton.html) establece la descripción del contenido para el botón más a la add\_note recurso de cadena, lo que podría ser definido como "Añadir Nota" para una interfaz de idioma Inglés:

<ImageButton

android: id = "+ id @ / add\_note\_button"

android: src = "@ dibujable / add\_note"

android: contentDescription = "@ cadena / add\_note" />

Con la inclusión de la descripción, los servicios basados ​​en la voz de accesibilidad puede anunciar "Añadir nota" cuando un usuario mueve el foco a este botón, o se cierne sobre él.

**Nota:** Para [EditText](http://developer.android.com/reference/android/widget/EditText.html) campos, proporcionar un [androide: pista](http://developer.android.com/reference/android/widget/TextView.html#attr_android:hint) de atributo para ayudar a los usuarios a comprender el contenido de lo que se espera.

**Habilitación de enfoque de navegación**

Enfoque de navegación permite a los usuarios con discapacidad para desplazarse por la interfaz de usuario controla el uso de un controlador direccional. Los controladores de dirección puede ser física, como una rueda de desplazamiento se puede hacer clic, pad direccional (D-pad) o las teclas de flecha, tecla de tabulación de navegación con un teclado conectado o una aplicación de software, tales como el [teclado Eyes-Free](https://play.google.com/store/apps/details?id=com.googlecode.eyesfree.inputmethod.latin) , que ofrece una dirección en la pantalla controlar.

Un controlador de dirección es un medio primario de navegación para muchos usuarios. Verifique que todas las interfaces de usuario (UI) de controles de la aplicación son accesibles sin necesidad de utilizar la pantalla táctil y que al hacer clic con el botón central (o el botón OK) de un controlador de dirección tiene el mismo efecto que tocar los controles en la pantalla táctil. Para obtener más información en las pruebas de controles de dirección, consulte [la navegación Pruebas de enfoque](http://developer.android.com/guide/topics/ui/accessibility/apps.html#test-navigation) .

**Habilitación de vista el enfoque**

Un elemento de la interfaz de usuario es accesible a través de controles de dirección cuando su [android:focusable](http://developer.android.com/reference/android/view/View.html#attr_android:focusable) atributo se establece en true . Esta configuración permite a los usuarios centrarse en el elemento utilizando los controles de dirección y luego interactuar con él. La interfaz de usuario controla el marco proporcionado por la plataforma Android a recibir el foco de forma predeterminada y visualmente indicar el enfoque al cambiar la apariencia del control.

Android ofrece varias APIs que permiten controlar si un control de interfaz de usuario puede tener el foco e incluso solicitar que un control se da el foco:

* [setFocusable()](http://developer.android.com/reference/android/view/View.html#setFocusable%28boolean%29)
* [isFocusable()](http://developer.android.com/reference/android/view/View.html#isFocusable%28%29)
* [requestFocus()](http://developer.android.com/reference/android/view/View.html#requestFocus%28%29)

Cuando se trabaja con una vista que no puede tener el foco de forma predeterminada, puede hacer que pueden recibir el foco desde el archivo XML mediante el establecimiento de diseño del [android:focusable](http://developer.android.com/reference/android/view/View.html#attr_android:focusable) atributo true o mediante el [setFocusable()](http://developer.android.com/reference/android/view/View.html#setFocusable%28boolean%29) método.

**Control del orden de enfoque**

Cuando los usuarios navegar en cualquier dirección utilizando los controles de dirección, el enfoque se pasa de un elemento de la interfaz de usuario (Vista) a otro, según lo determinado por el enfoque de ordenar. El orden del movimiento de enfoque se basa en un algoritmo que encuentra el vecino más cercano en una dirección dada. En casos raros, el algoritmo por defecto no puede coincidir con el orden que usted piensa para su interfaz de usuario. En estas situaciones, puede proporcionar reemplazos explícitas a la orden usando los siguientes atributos XML en el archivo de diseño:

[android:nextFocusDown](http://developer.android.com/reference/android/view/View.html#attr_android:nextFocusDown)

Define el punto de vista junto al recibir el foco cuando el usuario navega hacia abajo.

[android:nextFocusLeft](http://developer.android.com/reference/android/view/View.html#attr_android:nextFocusLeft)

Define el punto de vista junto al recibir el foco cuando el usuario navega izquierda.

[android:nextFocusRight](http://developer.android.com/reference/android/view/View.html#attr_android:nextFocusRight)

Define el punto de vista junto al recibir el foco cuando el usuario navega bien.

[android:nextFocusUp](http://developer.android.com/reference/android/view/View.html#attr_android:nextFocusUp)

Define el punto de vista junto al recibir el foco cuando el usuario navega hacia arriba.

En el ejemplo siguiente esquema XML muestra dos elementos de la interfaz de la interfaz de usuario donde el [android:nextFocusDown](http://developer.android.com/reference/android/view/View.html#attr_android:nextFocusDown) y [android:nextFocusUp](http://developer.android.com/reference/android/view/View.html#attr_android:nextFocusUp) atributos han sido establecidos explícitamente. La [TextView](http://developer.android.com/reference/android/widget/TextView.html) se encuentra a la derecha de la [EditText](http://developer.android.com/reference/android/widget/EditText.html) . Sin embargo, dado que estas propiedades se han establecido, el [TextView](http://developer.android.com/reference/android/widget/TextView.html) elemento ahora se puede llegar pulsando la flecha hacia abajo cuando el foco está en la [EditText](http://developer.android.com/reference/android/widget/EditText.html) elemento:

<LinearLayout android: orientation = "horizontal"

... >

<EditarTexto android: id = "+ @ id / edit"

android: nextFocusDown = "@ + id / texto"

... />

<TextView android: id = "+ id @ / texto"

android: enfocable = "true"

android: text = "Hola, soy un TextView enfocable"

android: nextFocusUp = "@ id / edit"

... />

</ LinearLayout>

Cuando se modifica la Orden del enfoque, asegúrese de que la navegación funciona como se espera en todas las direcciones de cada control de la interfaz de usuario y cuando se navega en sentido contrario (para volver al punto de que viene).

**Nota:** Puede modificar el orden del foco de los componentes de la interfaz de usuario en tiempo de ejecución, utilizando métodos como el [setNextFocusDownId()](http://developer.android.com/reference/android/view/View.html#setNextFocusDownId%28int%29) y [setNextFocusRightId()](http://developer.android.com/reference/android/view/View.html#setNextFocusRightId%28int%29) .

**Edificio accesible Vistas personalizadas**

Si su aplicación requiere un [componente de la vista personalizada](http://developer.android.com/guide/topics/ui/custom-components.html) , debe hacer un trabajo adicional para asegurar que la vista personalizada es accesible. Estas son las principales tareas para garantizar la accesibilidad de su punto de vista:

* Administrar los clics de dirección del controlador
* Implementar métodos de accesibilidad API
* Enviar [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) objetos específicos para su vista personalizada
* Llenar [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) y [AccessibilityNodeInfo](http://developer.android.com/reference/android/view/accessibility/AccessibilityNodeInfo.html) para la vista

**Manejo de clics de dirección del controlador**

En la mayoría de los dispositivos, hacer clic en una vista con un controlador de dirección envía un [KeyEvent](http://developer.android.com/reference/android/view/KeyEvent.html) con [KEYCODE\_DPAD\_CENTER](http://developer.android.com/reference/android/view/KeyEvent.html#KEYCODE_DPAD_CENTER) a la vista en la actualidad en el foco. Todo Android estándar de opiniones ya manejar [KEYCODE\_DPAD\_CENTER](http://developer.android.com/reference/android/view/KeyEvent.html#KEYCODE_DPAD_CENTER) adecuadamente. Cuando se construye una costumbre [View](http://developer.android.com/reference/android/view/View.html) de control, asegúrese de que este evento tiene el mismo efecto que tocar el punto de vista sobre la pantalla táctil.

Su control personalizado también debe tratar la [KEYCODE\_ENTER](http://developer.android.com/reference/android/view/KeyEvent.html#KEYCODE_ENTER) caso de que el mismo [KEYCODE\_DPAD\_CENTER](http://developer.android.com/reference/android/view/KeyEvent.html#KEYCODE_DPAD_CENTER) . Este enfoque hace que la interacción de un teclado completo mucho más fácil para los usuarios.

**Implementar métodos de accesibilidad API**

Eventos de accesibilidad son los mensajes de los usuarios acerca de la interacción con los componentes de la interfaz visual de la aplicación. Estos mensajes son manejados por [los servicios de accesibilidad](http://developer.android.com/guide/topics/ui/accessibility/services.html) , que utilizan la información de estos eventos para producir información complementaria y le pide que los usuarios han permitido a los servicios de accesibilidad. A partir de Android 4.0 (Nivel API 14) y superior, los métodos para la generación de eventos de accesibilidad se han ampliado para proporcionar información más detallada más allá de la [AccessibilityEventSource](http://developer.android.com/reference/android/view/accessibility/AccessibilityEventSource.html) introducido en la interfaz de Android 1.6 (Nivel API 4). Los métodos Expandir la accesibilidad son parte de la [View](http://developer.android.com/reference/android/view/View.html) de clases, así como el [View.AccessibilityDelegate](http://developer.android.com/reference/android/view/View.AccessibilityDelegate.html) clase. Los métodos son como sigue:

[sendAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#sendAccessibilityEvent%28int%29)

(Nivel API 4) Este método se llama cuando un usuario realiza una acción sobre un punto de vista. El evento se clasifica con un tipo de acción del usuario como [TYPE\_VIEW\_CLICKED](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html#TYPE_VIEW_CLICKED) . Por lo general, no es necesario aplicar este método a menos que se crea una vista personalizada.

[sendAccessibilityEventUnchecked()](http://developer.android.com/reference/android/view/View.html#sendAccessibilityEventUnchecked%28android.view.accessibility.AccessibilityEvent%29)

(Nivel API 4) Este método se utiliza cuando el código de llamada debe controlar directamente el control de la accesibilidad está activada en el dispositivo ( [AccessibilityManager.isEnabled()](http://developer.android.com/reference/android/view/accessibility/AccessibilityManager.html#isEnabled%28%29) ). Si lo hace poner en práctica este método, usted debe asumir que el método de llamada ya se ha comprobado que el acceso está habilitado y el resultado es true . Por lo general, no es necesario poner en práctica este método para una vista personalizada.

[dispatchPopulateAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#dispatchPopulateAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29)

(Nivel API 4) El sistema llama a este método cuando su vista personalizada genera un evento de la accesibilidad. A partir del nivel 14 de la API, la aplicación por defecto de este método llama a [onPopulateAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#onPopulateAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29) de este punto de vista y luego el [dispatchPopulateAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#dispatchPopulateAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29) método para cada niño de este punto de vista. Con el fin de apoyar los servicios de accesibilidad en las revisiones de Android *anteriores* a 4.0 (nivel API 14) que *debe* reemplazar este método y rellenar [getText()](http://developer.android.com/reference/android/view/accessibility/AccessibilityRecord.html#getText%28%29) con un texto descriptivo para la vista personalizada.

[onPopulateAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#onPopulateAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29)

(Nivel API 14) Este método establece el texto resultante de un [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) para la vista. Este método también se conoce como si la vista es un niño de un punto de vista que genera un evento de la accesibilidad.

**Nota:** Modificación de atributos adicionales más allá del texto dentro de este método potencialmente sobrescribe propiedades establecidas por otros métodos. Así, mientras que son los atributos capaces de modificar de la accesibilidad con el caso de este método, se debe limitar estos cambios en el contenido de sólo texto y utilizar el [onInitializeAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#onInitializeAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29) método para modificar otras propiedades del evento.

**Nota:** Si la ejecución de este evento requiere por completo overiding el texto de salida sin permitir que otras partes de su disposición a modificar su contenido, entonces no llamar a la implementación de este método de súper en el código.

[onInitializeAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#onInitializeAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29)

(Nivel API 14) El sistema llama a este método para obtener información adicional sobre el estado de la vista, más allá del contenido del texto. Si su vista personalizada proporciona control interactivo más allá de un simple [TextView](http://developer.android.com/reference/android/widget/TextView.html) o [Button](http://developer.android.com/reference/android/widget/Button.html) , debe reemplazar este método y establecer la información adicional acerca de su opinión en el caso de usar este método, como el campo de contraseña de tipo casilla de verificación tipo, o los estados que proporcionan la interacción del usuario o comentarios . Si lo haces reemplazar este método, usted debe llamar a su aplicación súper y sólo modificar las propiedades que no han sido establecidas por la superclase.

[onInitializeAccessibilityNodeInfo()](http://developer.android.com/reference/android/view/View.html#onInitializeAccessibilityNodeInfo%28android.view.accessibility.AccessibilityNodeInfo%29)

(Nivel de API 14) Este método proporciona servicios de accesibilidad con información sobre el estado de la vista. El valor por defecto [View](http://developer.android.com/reference/android/view/View.html) la aplicación establece un conjunto estándar de propiedades de la vista, pero si su vista personalizada proporciona control interactivo más allá de un simple [TextView](http://developer.android.com/reference/android/widget/TextView.html) o [Button](http://developer.android.com/reference/android/widget/Button.html) , debe reemplazar este método y establecer la información adicional acerca de su punto de vista en el [AccessibilityNodeInfo](http://developer.android.com/reference/android/view/accessibility/AccessibilityNodeInfo.html) objeto manipulado con este método.

[onRequestSendAccessibilityEvent()](http://developer.android.com/reference/android/view/ViewGroup.html#onRequestSendAccessibilityEvent%28android.view.View,%20android.view.accessibility.AccessibilityEvent%29)

(Nivel API 14) El sistema llama a este método cuando un niño de su punto de vista ha generado una [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) . Este paso permite a la vista el padre de enmendar el caso de la accesibilidad con información adicional. Se debe implementar este método sólo si su vista personalizada puede tener puntos de vista del niño y si la vista los padres pueden proporcionar información de contexto para el caso de la accesibilidad que podrían ser útiles a los servicios de accesibilidad.

Con el fin de apoyar a estos métodos de accesibilidad para una vista personalizada, usted debe tomar uno de los siguientes métodos:

* Si su aplicación está destinada para Android 4.0 (nivel de la API 14) y más alto, sustituir e implementar los métodos de accesibilidad antes mencionados directamente en la clase de vista personalizado.
* Si su vista personalizada está diseñada para ser compatible con Android 1.6 (Nivel API 4) y por encima, añadir el Android [Biblioteca de compatibilidad](http://developer.android.com/sdk/compatibility-library.html) , la revisión 5 o superior, a su proyecto. Entonces, dentro de la clase de vista personalizado, llame a la [ViewCompat.setAccessibilityDelegate()](http://developer.android.com/reference/android/support/v4/view/ViewCompat.html#setAccessibilityDelegate%28android.view.View,%20android.support.v4.view.AccessibilityDelegateCompat%29) método para implementar los métodos de acceso antes. Para ver un ejemplo de este enfoque, consulte la Biblioteca de compatibilidad de Android (versión 5 o superior) muestra AccessibilityDelegateSupportActivity en ( <sdk>/extras/android/support/v4/samples/Support4Demos/ )

En cualquier caso, se deben implementar los métodos siguientes de accesibilidad para la clase de vista personalizada:

* [dispatchPopulateAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#dispatchPopulateAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29)
* [onPopulateAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#onPopulateAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29)
* [onInitializeAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#onInitializeAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29)
* [onInitializeAccessibilityNodeInfo()](http://developer.android.com/reference/android/view/View.html#onInitializeAccessibilityNodeInfo%28android.view.accessibility.AccessibilityNodeInfo%29)

Para obtener más información sobre la implementación de estos métodos, vea [Llenar Eventos de accesibilidad](http://developer.android.com/guide/topics/ui/accessibility/apps.html#populate-events) .

**El envío de los eventos de accesibilidad**

Dependiendo de los detalles de su vista personalizada, puede ser necesario para enviar [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) objetos en un par de veces diferentes o para eventos que no controla la aplicación por defecto. La [View](http://developer.android.com/reference/android/view/View.html) de clases proporciona una implementación predeterminada para estos tipos de eventos:

* Empezando por el nivel de la API 4:
  + [TYPE\_VIEW\_CLICKED](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html#TYPE_VIEW_CLICKED)
  + [TYPE\_VIEW\_LONG\_CLICKED](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html#TYPE_VIEW_LONG_CLICKED)
  + [TYPE\_VIEW\_FOCUSED](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html#TYPE_VIEW_FOCUSED)
* A partir de nivel 14 de la API:
  + [TYPE\_VIEW\_SCROLLED](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html#TYPE_VIEW_SCROLLED)
  + [TYPE\_VIEW\_HOVER\_ENTER](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html#TYPE_VIEW_HOVER_ENTER)
  + [TYPE\_VIEW\_HOVER\_EXIT](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html#TYPE_VIEW_HOVER_EXIT)

**Nota: Los** eventos Hover se asocian con la función de Búsqueda por Touch, que utiliza estos eventos como disparadores para proporcionar avisos audibles para los elementos de la interfaz de usuario.

En general, usted debe enviar una [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) siempre que el contenido de sus cambios de la vista personalizada. Por ejemplo, si va a implementar una barra de desplazamiento personalizado que permite al usuario seleccionar un valor numérico pulsando las flechas izquierda o derecha, la vista personalizada debe emitir un evento del tipo [TYPE\_VIEW\_TEXT\_CHANGED](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html#TYPE_VIEW_TEXT_CHANGED) siempre que cambie el valor del control deslizante. El código de ejemplo siguiente muestra el uso de la [sendAccessibilityEvent()](http://developer.android.com/reference/android/view/accessibility/AccessibilityEventSource.html#sendAccessibilityEvent%28int%29) el método para informar de este evento.

@Override

public boolean onKeyUp (int keyCode, evento KeyEvent) {

if (keyCode == KeyEvent.KEYCODE\_DPAD\_LEFT) {

mCurrentValue -;

sendAccessibilityEvent (AccessibilityEvent.TYPE\_VIEW\_TEXT\_CHANGED);

return true;

}

...

}

**Rellenar los eventos de accesibilidad**

Cada [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) tiene un conjunto de propiedades requeridas que describen el estado actual de la vista. Estas propiedades incluyen cosas tales como el nombre de la clase de la vista, descripción del contenido y estado de activación. Las propiedades específicas requeridas para cada tipo de evento se describen en la [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) documentación de referencia. El [View](http://developer.android.com/reference/android/view/View.html) la aplicación proporciona los valores predeterminados para estas propiedades. Muchos de estos valores, incluyendo el nombre de la clase y la hora del evento, se proporcionan de forma automática. Si va a crear un componente de vista personalizada, debe proporcionar cierta información sobre el contenido y las características de la vista. Esta información puede ser tan simple como una etiqueta de botón, pero también puede incluir la información de estado adicional que desee agregar al evento.

El requisito mínimo para el suministro de información a los servicios de accesibilidad, con una vista personalizada es implementar [dispatchPopulateAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#dispatchPopulateAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29) . Este método es llamado por el sistema para solicitar información para un [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) y hace que su vista personalizada compatible con los servicios de accesibilidad de Android 1.6 (Nivel API 4) y superior. El código de ejemplo siguiente muestra una implementación básica de este método.

@Override

public void dispatchPopulateAccessibilityEvent (AccessibilityEvent evento) {

super.dispatchPopulateAccessibilityEvent (evento);

/ / Llamar a la aplicación estupenda para llenar su texto para el evento, que

/ / Llama a onPopulateAccessibilityEvent () en el nivel API de 14 en adelante.

/ / En caso de que se está ejecutando en una revisión de la API antes que 14, consulte

/ / El contenido del texto del evento y añadir un texto apropiado

/ / Descripción de este punto de vista personalizada:

CharSequence text = getText ();

Si los númerosTextUtils.isEmpty (texto)) {

. event.getText () add (texto);

}

}

En Android 4.0 (Nivel API 14) y más alto, el [onPopulateAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#onPopulateAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29) y [onInitializeAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#onInitializeAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29) métodos son el método recomendado para poblar o modificar la información en un [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) . Utilice el [onPopulateAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#onPopulateAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29) método específicamente para añadir o modificar el contenido de texto del evento, que se convierte en audibles por los servicios de accesibilidad, como Talkback. Utilice el [onInitializeAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#onInitializeAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29) método para llenar la información adicional sobre el evento, tales como el estado de selección de la vista.

Además, también se debe aplicar el [onInitializeAccessibilityNodeInfo()](http://developer.android.com/reference/android/view/View.html#onInitializeAccessibilityNodeInfo%28android.view.accessibility.AccessibilityNodeInfo%29) método. [AccessibilityNodeInfo](http://developer.android.com/reference/android/view/accessibility/AccessibilityNodeInfo.html) objetos pobladas por este método son utilizados por los servicios de accesibilidad para investigar la jerarquía de la vista que genera un evento accesibilidad después de recibir ese caso, para obtener una información más detallada contexto y proporcionar información adecuada a los usuarios.

El código de ejemplo siguiente muestra cómo reemplazar estos tres métodos mediante el uso de [ViewCompat.setAccessibilityDelegate()](http://developer.android.com/reference/android/support/v4/view/ViewCompat.html#setAccessibilityDelegate%28android.view.View,%20android.support.v4.view.AccessibilityDelegateCompat%29) . Tenga en cuenta que este código de ejemplo requiere que el Android [Biblioteca de Apoyo](http://developer.android.com/sdk/compatibility-library.html) para el Nivel 4 de la API (revisión 5 o superior) se agrega a su proyecto.

ViewCompat.setAccessibilityDelegate (nuevo AccessibilityDelegateCompat () {

@Override

public void onPopulateAccessibilityEvent (host Vista, AccessibilityEvent evento) {

super.onPopulateAccessibilityEvent (anfitrión, evento);

/ / Llamamos a la aplicación estupenda para llenar su texto para el

EVENT: A continuación, añadimos nuestro texto no está presente en un super clase.

/ / Muy a menudo, sólo tiene que añadir el texto de la vista personalizada.

CharSequence text = getText ();

Si los númerosTextUtils.isEmpty (texto)) {

. event.getText () add (texto);

}

}

@Override

public void onInitializeAccessibilityEvent (host Vista, AccessibilityEvent evento) {

super.onInitializeAccessibilityEvent (anfitrión, evento);

/ / Llamamos a la aplicación para permitir que las clases súper súper

/ / Establecer las propiedades correspondientes del evento. A continuación, añadimos la nueva propiedad

/ / (Comprobado) que no es compatible con una super clase.

event.setChecked (isChecked ());

}

@Override

onInitializeAccessibilityNodeInfo public void (host Vista,

AccessibilityNodeInfoCompat info) {

super.onInitializeAccessibilityNodeInfo (acogida, información);

/ / Llamamos a la aplicación para permitir que las clases súper súper establece

/ / Propiedades apropiada información. A continuación, añadimos nuestras propiedades

/ / (Comprobable y comprobado) que no están soportadas por un super clase.

info.setCheckable (true);

info.setChecked (isChecked ());

/ / Muy a menudo sólo es necesario añadir el texto de la vista personalizada.

CharSequence text = getText ();

Si los númerosTextUtils.isEmpty (texto)) {

info.setText (texto);

}

}

}

En las aplicaciones dirigidas a Android 4.0 (Nivel API 14) y superior, estos métodos pueden aplicarse directamente en la clase de vista personalizado. Para otro ejemplo de este enfoque, ver el Android [Support Library](http://developer.android.com/sdk/compatibility-library.html) (revisión 5 o superior) muestra AccessibilityDelegateSupportActivity en ( <sdk>/extras/android/support/v4/samples/Support4Demos/ ).

**Nota:** Usted puede encontrar información sobre la aplicación de la accesibilidad para las vistas personalizadas escritas antes de Android 4.0, que describe el uso de la [dispatchPopulateAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#dispatchPopulateAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29) método para poblar AccessibilityEvents. A partir de la versión Android 4.0, sin embargo, el enfoque recomendado es utilizar el [onPopulateAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#onPopulateAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29) y [onInitializeAccessibilityEvent()](http://developer.android.com/reference/android/view/View.html#onInitializeAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29) métodos.

**Revisión de la Accesibilidad**

Comprobación de la accesibilidad de su aplicación es una parte importante de asegurar que los usuarios tienen una gran experiencia. Usted puede probar las partes más importantes de la accesibilidad de probar la aplicación con el habilitado una respuesta audible y la navegación dentro de la aplicación de pruebas utilizando los controles de dirección.

**Prueba de respuesta audible**

Se puede simular la experiencia de muchos usuarios al permitir que un servicio de accesibilidad que habla como usted se mueve alrededor de la pantalla. El Explora por el servicio de acceso táctil, que está disponible en dispositivos con Android 4.0 y versiones posteriores. El [TalkBack](https://play.google.com/store/apps/details?id=com.google.android.marvin.talkback) servicio de la accesibilidad, por el [Proyecto Eyes-Free](http://code.google.com/p/eyes-free/) viene preinstalado en muchos dispositivos Android.

Para habilitar TalkBack en las revisiones anteriores de Android a Android 4.0:

1. Inicie la aplicación de configuración.
2. Vaya a la categoría de **Accesibilidad** y seleccionarlo.
3. Seleccione **Accesibilidad** para activarlo.
4. Seleccione **TalkBack** para activarlo.

**Nota:** Si el servicio de acceso TalkBack no está disponible, se puede instalar de forma gratuita en [juego Google](http://play.google.com) .

Para activar la Búsqueda por táctil con Android 4.0 y versiones posteriores:

1. Inicie la aplicación de configuración.
2. Vaya a la categoría de **Accesibilidad** y seleccionarlo.
3. Seleccione la **TalkBack** para activarlo.
4. Volver a la categoría de **Accesibilidad** y seleccione **Explorar por Touch** para activarlo.

**Nota:** Debe activar la TalkBack *primero,* de lo contrario esta opción no está disponible.

**Prueba de enfoque de navegación**

Como parte de su test de accesibilidad, puede probar la navegación de la aplicación utilizando el enfoque, incluso si los dispositivos de prueba no tiene un controlador direccional. El [emulador de Android](http://developer.android.com/guide/developing/tools/emulator.html) proporciona un controlador de simulación de dirección que usted pueda usar para probar la navegación. También puede utilizar un controlador de dirección basado en software, como por ejemplo la proporcionada por el [teclado Eyes-Free](https://play.google.com/store/apps/details?id=com.googlecode.eyesfree.inputmethod.latin) para simular el uso de un D-pad.

**Servicios de construcción de Accesibilidad**

**Temas**

1. [Las declaraciones de manifiestos y licencias](http://developer.android.com/guide/topics/ui/accessibility/services.html#manifest)
   1. [Declaración de accesibilidad de servicios](http://developer.android.com/guide/topics/ui/accessibility/service-declaration)
   2. [Accesibilidad de configuración del servicio](http://developer.android.com/guide/topics/ui/accessibility/services.html#service-config)
2. [Métodos AccessibilityService](http://developer.android.com/guide/topics/ui/accessibility/services.html#methods)
3. [Obtención de los detalles del evento](http://developer.android.com/guide/topics/ui/accessibility/services.html#event-details)
4. [Ejemplo de código](http://developer.android.com/guide/topics/ui/accessibility/services.html#examples)

**Clases de clave**

1. [AccessibilityService](http://developer.android.com/reference/android/accessibilityservice/AccessibilityService.html)
2. [AccessibilityServiceInfo](http://developer.android.com/reference/android/accessibilityservice/AccessibilityServiceInfo.html)
3. [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html)
4. [AccessibilityRecord](http://developer.android.com/reference/android/view/accessibility/AccessibilityRecord.html)
5. [AccessibilityNodeInfo](http://developer.android.com/reference/android/view/accessibility/AccessibilityNodeInfo.html)

**Véase también**

1. [Implementación de Accesibilidad](http://developer.android.com/training/accessibility/index.html)

Un servicio de la accesibilidad es una aplicación que proporciona mejoras en la interfaz de usuario para ayudar a los usuarios con discapacidad, o que temporalmente puede ser capaz de interactuar completamente con un dispositivo. Por ejemplo, los usuarios que están impulsando, el cuidado de un niño pequeño o en una fiesta muy ruidosa puede ser que necesite información adicional o alternativa de interfaz.

Android proporciona los servicios estándar de accesibilidad, incluyendo TalkBack, y los desarrolladores pueden crear y distribuir sus propios servicios. Este documento explica los fundamentos de la creación de un servicio de accesibilidad.

La capacidad para que usted pueda construir y desplegar servicios de accesibilidad fue introducida con Android 1.6 (Nivel API 4) y ha recibido importantes mejoras con Android 4.0 (Nivel API 14). La Biblioteca de compatibilidad de Android se ha actualizado también con el lanzamiento de Android 4.0 para ofrecer soporte para estas características de accesibilidad mejoradas de nuevo a Android 1.6. Desarrolladores destinados para los servicios de accesibilidad ampliamente compatibles se les anima a utilizar la [Biblioteca de compatibilidad](http://developer.android.com/sdk/compatibility-library.html) y el desarrollo de las características de accesibilidad más avanzados introducidos en Android 4.0.

**Las declaraciones de manifiestos y licencias**

Las aplicaciones que proporcionan servicios de accesibilidad debe incluir declaraciones específicas en los manifiestos de sus aplicaciones con el fin de ser tratado como un servicio de acceso mediante un sistema de Android. Esta sección explica los ajustes necesarios y opcionales para los servicios de accesibilidad.

**Declaración de accesibilidad de servicios**

Con el fin de ser tratado como un servicio de accesibilidad, su solicitud debe incluir el service elemento (en lugar de la activity los elementos) dentro de la application los elementos de su manifiesto. Además, en el service los elementos, también debe incluir un servicio de filtro de acceso a la intención, como se muestra en el siguiente ejemplo:

<application>

<Servicio de android: name = ".MyAccessibilityService "

android: label = "@ cadena / accessibility\_service\_label">

<intent-filter>

<Action: name = "android.accessibilityservice.AccessibilityService "/>

</ Intención de filtro>

Salud.

</ Application>

Estas declaraciones son necesarios para todos los servicios de accesibilidad implementadas en Android 1.6 (Nivel API 4) o superior.

**Accesibilidad de configuración del servicio**

Los servicios de accesibilidad también debe proporcionar una configuración que especifica los tipos de eventos de accesibilidad que maneja el servicio y la información adicional sobre el servicio. La configuración de un servicio de accesibilidad está contenida en el [AccessibilityServiceInfo](http://developer.android.com/reference/android/accessibilityservice/AccessibilityServiceInfo.html) clase. Su servicio se puede construir y establecer una configuración utilizando una instancia de esta clase y [setServiceInfo()](http://developer.android.com/reference/android/accessibilityservice/AccessibilityService.html#setServiceInfo%28android.accessibilityservice.AccessibilityServiceInfo%29) en tiempo de ejecución. Sin embargo, no todas las opciones de configuración están disponibles con este método.

A partir de Android 4.0, que puede incluir una <meta-data> elemento en su manifiesto con una referencia a un archivo de configuración, que le permite ajustar la gama de opciones para el servicio de accesibilidad, como se muestra en el siguiente ejemplo:

<Servicio de android: name = ".MyAccessibilityService ">

...

Meta Datos

android: name = "android.accessibilityservice"

android: resource = "@ xml / accessibility\_service\_config" />

Salud.

Este elemento de metadatos se refiere a un archivo XML que se crea en el directorio de recursos de la aplicación ( <project\_dir>/res/xml/accessibility\_service\_config.xml ). El siguiente código muestra el contenido de ejemplo para el archivo de configuración del servicio:

<Servicio accesibilidad xmlns: android = "http://schemas.android.com/apk/res/android~~HEAD=NNS"

android: description = "@ cadena / accessibility\_service\_description"

android: packageNames = "com.example.android.apis"

android: accessibilityEventTypes = "typeAllMask"

android: accessibilityFlags = "flagDefault"

android: accessibilityFeedbackType = "feedbackSpoken"

android: notificationTimeout = "100"

android: canRetrieveWindowContent = "true"

android: settingsActivity = "com.example.android.accessibility.ServiceSettingsActivity "

/>

Una de las funciones más importantes de los parámetros de configuración de accesibilidad de servicios es para que usted pueda especificar qué tipos de eventos de accesibilidad de su servicio puede manejar. Ser capaz de especificar esta información permite a los servicios de accesibilidad para cooperar entre sí, y le permite a un desarrollador de la flexibilidad de manejar sólo los tipos de eventos específicos de aplicaciones específicas. El filtrado de eventos pueden incluir los siguientes criterios:

* **Nombres de los paquetes** - Especificar los nombres de los paquetes de aplicaciones cuya accesibilidad eventos usted quiere que su servicio para manejar. Si se omite este parámetro, el servicio de acceso se considera disponible para eventos de accesibilidad de servicios para cualquier aplicación. Este parámetro se puede establecer en los archivos de configuración de accesibilidad de servicios con el android:packageNames atributo como una lista separada por comas, o un conjunto utilizando la [AccessibilityServiceInfo.packageNames](http://developer.android.com/reference/android/accessibilityservice/AccessibilityServiceInfo.html#packageNames) miembro.
* **Tipos de eventos:** especifique los tipos de eventos de accesibilidad que desee su servicio para manejar. Este parámetro se puede establecer en los archivos de configuración de accesibilidad de servicios con el android:accessibilityEventTypes atributo como una lista separada por comas, o un conjunto utilizando la [AccessibilityServiceInfo.eventTypes](http://developer.android.com/reference/android/accessibilityservice/AccessibilityServiceInfo.html#eventTypes) miembro.

Para obtener más información acerca de los atributos XML que pueden ser utilizados en el archivo de configuración de accesibilidad de servicio, siga estos enlaces a la documentación de referencia:

* [android:description](http://developer.android.com/reference/android/R.styleable.html#AccessibilityService_description)
* [android:packageNames](http://developer.android.com/reference/android/R.styleable.html#AccessibilityService_packageNames)
* [android:accessibilityEventTypes](http://developer.android.com/reference/android/R.styleable.html#AccessibilityService_accessibilityEventTypes)
* [android:accessibilityFlags](http://developer.android.com/reference/android/R.styleable.html#AccessibilityService_accessibilityFlags)
* [android:accessibilityFeedbackType](http://developer.android.com/reference/android/R.styleable.html#AccessibilityService_accessibilityFeedbackType)
* [android:notificationTimeout](http://developer.android.com/reference/android/R.styleable.html#AccessibilityService_notificationTimeout)
* [android:canRetrieveWindowContent](http://developer.android.com/reference/android/R.styleable.html#AccessibilityService_canRetrieveWindowContent)
* [android:settingsActivity](http://developer.android.com/reference/android/R.styleable.html#AccessibilityService_settingsActivity)

Para obtener más información acerca de los ajustes de configuración se puede establecer dinámicamente en tiempo de ejecución, consulte la [AccessibilityServiceInfo](http://developer.android.com/reference/android/accessibilityservice/AccessibilityServiceInfo.html) documentación de referencia.

**Métodos AccessibilityService**

Una aplicación que proporciona servicio de la accesibilidad debe extender el [AccessibilityService](http://developer.android.com/reference/android/accessibilityservice/AccessibilityService.html) clase y redefinir los siguientes métodos de esa clase. Estos métodos se presentan en el orden en que sean llamados por el sistema Android, desde que se inicia el servicio ( [onServiceConnected()](http://developer.android.com/reference/android/accessibilityservice/AccessibilityService.html#onServiceConnected%28%29) ), mientras se está ejecutando ( [onAccessibilityEvent()](http://developer.android.com/reference/android/accessibilityservice/AccessibilityService.html#onAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29) , [onInterrupt()](http://developer.android.com/reference/android/accessibilityservice/AccessibilityService.html#onInterrupt%28%29) ) que cuando se apaga ( [onUnbind()](http://developer.android.com/reference/android/app/Service.html#onUnbind%28android.content.Intent%29) ).

* [onServiceConnected()](http://developer.android.com/reference/android/accessibilityservice/AccessibilityService.html#onServiceConnected%28%29) - (opcional) Este sistema llama a este método cuando se conecta con éxito a su servicio de accesibilidad. Utilice este método para hacer los pasos de configuración de una sola vez para su servicio, incluyendo la conexión a los servicios de usuario del sistema de retroalimentación, tales como el gestor de audio o vibración del dispositivo. Si desea ajustar la configuración de su servicio en tiempo de ejecución o hacer ajustes de una sola vez, se trata de una buena ubicación desde la que llaman [setServiceInfo()](http://developer.android.com/reference/android/accessibilityservice/AccessibilityService.html#setServiceInfo%28android.accessibilityservice.AccessibilityServiceInfo%29) .
* [onAccessibilityEvent()](http://developer.android.com/reference/android/accessibilityservice/AccessibilityService.html#onAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29) - (obligatorio) Este método es llamado por el sistema cuando se detecta un [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) que coincide con el evento de filtrado de los parámetros especificados por el servicio de accesibilidad. Por ejemplo, cuando el usuario hace clic en un botón o se centra en un control de interfaz de usuario en una aplicación para la que el servicio de la accesibilidad está ofreciendo información. Cuando esto sucede, el sistema llama a este método de su servicio con la correspondiente [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) , que luego se puede interpretar y proporcionar información al usuario. Este método puede ser llamado varias veces el ciclo de vida de su servicio.
* [onInterrupt()](http://developer.android.com/reference/android/accessibilityservice/AccessibilityService.html#onInterrupt%28%29) - (obligatorio) Este método se llama cuando el sistema quiere interrumpir la retroalimentación de su servicio se proporciona, por lo general en respuesta a un usuario la adopción de medidas, como se mueve el foco a un control de la interfaz de usuario diferente a aquel para el cual usted está Actualmente, la retroalimentación. Este método puede ser llamado varias veces el ciclo de vida de su servicio.
* [onUnbind()](http://developer.android.com/reference/android/app/Service.html#onUnbind%28android.content.Intent%29) - (opcional) Este método se llama cuando el sistema está a punto de apagar el servicio de accesibilidad. Utilice este método para realizar cualquier procedimiento de cierre de un tiempo, incluida la eliminación de la asignación de los servicios de usuario del sistema de retroalimentación, tales como el gestor de audio o vibración del dispositivo.

Estos métodos de devolución de llamada proporciona la estructura básica para el servicio de accesibilidad. Es hasta usted para decidir sobre la forma de procesar los datos proporcionados por el sistema Android en forma de [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) objetos y proporcionar información al usuario.

**Obtención de los detalles del evento**

El sistema Android proporciona información a los servicios de accesibilidad de la interacción a través de la interfaz de usuario [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) objetos. Antes de Android 4,0, la información disponible en un evento accesibilidad, mientras que proporciona una cantidad significativa de detalles acerca de un control de interfaz de usuario seleccionado por el usuario, típicamente proporcionan información contextual limitada. En muchos casos, esta información de contexto que falta puede ser fundamental para comprender el significado del control seleccionado.

Un ejemplo típico de una interfaz donde el contexto es de suma importancia es un calendario o agenda. Si un usuario selecciona una ranura para las 4:00 PM tiempo en una lista de días de lunes a viernes y el servicio de acceso anuncia "16:00", pero no indica se trata de un viernes, un lunes, el mes o el día, esto no es ideal para la retroalimentación el usuario. En este caso, el contexto de un control de interfaz de usuario es de vital importancia para un usuario que quiere programar una reunión.

Android 4.0 se extiende de manera significativa la cantidad de información que un servicio de acceso puede obtener de una interacción de la interfaz de usuario mediante la composición de los eventos de accesibilidad basadas en la jerarquía de vistas. Una jerarquía de la vista es el conjunto de componentes de la interfaz de usuario que contienen el componente (sus padres) y los elementos de la interfaz de usuario que pueda formar parte de ese componente (sus hijos). De esta manera, el sistema Android puede proporcionar detalles mucho más rica sobre los eventos de accesibilidad, permitiendo servicios de accesibilidad para proporcionar información más útil para los usuarios.

Un servicio de la accesibilidad obtiene información acerca de un evento a través de una interfaz de usuario [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) pasado por el sistema para el servicio de [onAccessibilityEvent()](http://developer.android.com/reference/android/accessibilityservice/AccessibilityService.html#onAccessibilityEvent%28android.view.accessibility.AccessibilityEvent%29) el método de devolución de llamada. Este objeto proporciona detalles sobre el evento, incluyendo el tipo de objeto que se actúe sobre ellos, el texto descriptivo y otros detalles. A partir de Android 4.0 (y en versiones anteriores a través de la [AccessibilityEventCompat](http://developer.android.com/reference/android/support/v4/view/accessibility/AccessibilityEventCompat.html) objeto en la biblioteca de compatibilidad), se puede obtener información adicional sobre el evento con estas llamadas:

* [AccessibilityEvent.getRecordCount()](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html#getRecordCount%28%29) y [getRecord(int)](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html#getRecord%28int%29) - Estos métodos le permiten recuperar el conjunto de [AccessibilityRecord](http://developer.android.com/reference/android/view/accessibility/AccessibilityRecord.html) objetos que contribuyeron a la [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) pasó a usted por el sistema, que puede dar más contexto para el servicio de accesibilidad.
* [AccessibilityEvent.getSource()](http://developer.android.com/reference/android/view/accessibility/AccessibilityRecord.html#getSource%28%29) - Este método devuelve un [AccessibilityNodeInfo](http://developer.android.com/reference/android/view/accessibility/AccessibilityNodeInfo.html) objeto. Este objeto le permite solicitar a los padres y los hijos de los componentes que dieron origen al caso e investigar la accesibilidad de sus contenidos y estatales a fin de proporcionar

**Importante:** La capacidad para investigar la jerarquía de la vista de un [AccessibilityEvent](http://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html) puede exponer información privada del usuario de su servicio de accesibilidad. Por esta razón, el servicio tiene que pedir este nivel de acceso a través de la accesibilidad [de servicios de configuración XML](http://developer.android.com/guide/topics/ui/accessibility/services.html#service-config) de archivos, incluyendo el canRetrieveWindowContent atributo y establecerla en true . Si usted no incluye esta opción en la configuración de su servicio de archivos xml, las llamadas a [getSource()](http://developer.android.com/reference/android/view/accessibility/AccessibilityRecord.html#getSource%28%29) fallará.

**Ejemplo de código**

El proyecto de demostración de la API contiene dos ejemplos que pueden ser utilizados como punto de partida para la generación de servicios de accesibilidad ( <sdk>/samples/<platform>/ApiDemos/src/com/example/android/apis/accessibility ):

* [ClockBackService](http://developer.android.com/resources/samples/ApiDemos/src/com/example/android/apis/accessibility/ClockBackService.html) - Este servicio se basa en la aplicación original de [AccessibilityService](http://developer.android.com/reference/android/accessibilityservice/AccessibilityService.html) y se puede utilizar como una base para el desarrollo de los servicios básicos de accesibilidad que sean compatibles con Android 1,6 (Nivel API 4) y superior.
* [TaskBackService](http://developer.android.com/resources/samples/ApiDemos/src/com/example/android/apis/accessibility/TaskBackService.html) - Este servicio se basa en la API de mejorar la accesibilidad introducido en Android 4.0 (Nivel API 14). Sin embargo, usted puede utilizar el Android [Libary Apoyo](http://developer.android.com/sdk/compatibility-library.html) a sustituir las clases introducidas en los niveles posteriores API (por ejemplo, [AccessibilityRecord](http://developer.android.com/reference/android/view/accessibility/AccessibilityRecord.html) , [AccessibilityNodeInfo](http://developer.android.com/reference/android/view/accessibility/AccessibilityNodeInfo.html) ) con equivalentes clases de apoyo de paquetes (por ejemplo, [AccessibilityRecordCompat](http://developer.android.com/reference/android/support/v4/view/accessibility/AccessibilityRecordCompat.html) , [AccessibilityNodeInfoCompat](http://developer.android.com/reference/android/support/v4/view/accessibility/AccessibilityNodeInfoCompat.html) ) para que este ejemplo funcione con las versiones de la API de nuevo a Android 1.6 (API Nivel 4).

**Enlazar a los datos con AdapterView**

**En este documento**

1. [Llenar el diseño con datos](http://developer.android.com/guide/topics/ui/binding.html#FillingTheLayout)
2. [Manejo de Selección de usuario](http://developer.android.com/guide/topics/ui/binding.html#HandlingUserSelections)

**Tutoriales relacionados**

1. [Hilandero](http://developer.android.com/resources/tutorials/views/hello-spinner.html)
2. [Ver la lista de](http://developer.android.com/resources/tutorials/views/hello-listview.html)
3. [Cuadricula](http://developer.android.com/resources/tutorials/views/hello-gridview.html)

El [AdapterView](http://developer.android.com/reference/android/widget/AdapterView.html) es una subclase ViewGroup cuyo hijo Vistas están determinadas por un [Adapter](http://developer.android.com/reference/android/widget/Adapter.html) que se une a los datos de algún tipo. AdapterView es útil cuando se necesita mostrar los datos almacenados (en lugar de cadenas de recursos o dibujables) en su diseño.

[Gallery](http://developer.android.com/reference/android/widget/Gallery.html) , [ListView](http://developer.android.com/reference/android/widget/ListView.html) , y [Spinner](http://developer.android.com/reference/android/widget/Spinner.html) son ejemplos de las subclases AdapterView que se pueden utilizar para unirse a un tipo específico de datos y mostrarlos de una manera determinada.

AdapterView objetos tienen dos responsabilidades principales:

* Llenar el diseño con los datos
* Manejo de las selecciones del usuario

**Llenar el diseño con datos**

Inserción de datos en el diseño se realiza normalmente mediante la unión de la clase AdapterView a un [Adapter](http://developer.android.com/reference/android/widget/Adapter.html) , que recupera los datos de una fuente externa (tal vez una lista de códigos que los suministros o los resultados de la consulta de la base de datos del dispositivo).

El siguiente ejemplo de código hace lo siguiente:

1. Crea un [Spinner](http://developer.android.com/reference/android/widget/Spinner.html) , con una vista existente y lo enlaza a un ArrayAdapter nuevos que lee una gran variedad de colores de los recursos locales.
2. Crea otro objeto Spinner de una vista y lo enlaza a un SimpleCursorAdapter nuevas que van a leer los nombres de las personas de los contactos del dispositivo (véase [Contacts.People](http://developer.android.com/reference/android/provider/Contacts.People.html) ).

/ / Obtener un Spinner y asociarla a un ArrayAdapter que

/ / Hace referencia a una matriz de cadenas.

Spinner s1 = (Spinner) findViewById (R.id.spinner1);

ArrayAdapter adaptador de ArrayAdapter.createFromResource = (

este R.array.colors y, Android.R.layout.simple\_spinner\_item);

adapter.setDropDownViewResource (android.R.layout.simple\_spinner\_dropdown\_item);

s1.setAdapter (adaptador);

/ / Carga un Spinner y enlazarlo a una consulta de datos.

static String privada [] PROYECCIÓN = new String [] {

Las personas.\_ID, Gente.NOMBRE

};

Spinner s2 = (Spinner) findViewById (R.id.spinner2);

Act = Cursor managedQuery (People.CONTENT\_URI, PROYECCIÓN, null, null);

SimpleCursorAdapter adapter2 = new SimpleCursorAdapter (esto,

AndroidR.layout.simple\_spinner\_item, / / ​​Utilizar una plantilla

/ / Que muestra una

Vista de texto

act, / / ​​Dar el cursor hasta el adaptador de la lista

new String [] {personas.NOMBRE}, / / ​​Mapa del nombre de columna del

/ / Base de datos para la gente ...

new int [] {androide.R.id.text1}); / / La "texto1" se define en el punto de vista

/ / El XML de la plantilla

adapter2.setDropDownViewResource (android.R.layout.simple\_spinner\_dropdown\_item);

s2.setAdapter (adapter2);

Tenga en cuenta que es necesario contar con las personas.Columna de \_ID en la proyección se utiliza con CursorAdapter o de lo contrario obtendrá una excepción.

Si durante el curso de la vida de la aplicación, se cambian los datos subyacentes que son leídos por el adaptador, usted debe llamar a [notifyDataSetChanged()](http://developer.android.com/reference/android/widget/ArrayAdapter.html#notifyDataSetChanged%28%29) . Esto se notificará a la vista de adjunto que los datos se han cambiado y que debería volver a cargar.

**Manejo de Selección de usuario**

Usted maneja la selección del usuario mediante la creación de la clase [AdapterView.OnItemClickListener](http://developer.android.com/reference/android/widget/AdapterView.OnItemClickListener.html) miembro a un oyente y la captura de los cambios de selección.

/ / Crear un mensaje de manipulación de objetos, como una clase anónima.

mMessageClickedHandler OnItemClickListener privada = new OnItemClickListener () {

pública onItemClick vacío (padre AdapterView, Vista v, la posición int, long id)

{

/ / Muestra un cuadro de mensaje.

Toast.makeText (mContext, "Tienes un evento", pan tostado.. LENGTH\_SHORT) show ();

}

};

/ / Ahora conectar a nuestro objeto y establecer su miembro onItemClickListener

/ / En nuestro objeto de controlador de clase.

mHistoryView = (ListView) findViewById (R.id.history);

mHistoryView.setOnItemClickListener (mMessageClickedHandler);

Para más discusión sobre cómo crear AdapterViews diferentes, lea los siguientes tutoriales: [Hola Spinner](http://developer.android.com/resources/tutorials/views/hello-spinner.html) , [Hola ListView](http://developer.android.com/resources/tutorials/views/hello-listview.html) y [GridView Hola](http://developer.android.com/resources/tutorials/views/hello-gridview.html) .

**Objetos comunes de diseño**

**En este documento**

1. [FrameLayout](http://developer.android.com/guide/topics/ui/layout-objects.html#framelayout)
2. [LinearLayout](http://developer.android.com/guide/topics/ui/layout-objects.html#linearlayout)
3. [TableLayout](http://developer.android.com/guide/topics/ui/layout-objects.html#tablelayout)
4. [RelativeLayout](http://developer.android.com/guide/topics/ui/layout-objects.html#relativelayout)
5. [Resumen de los Grupos de juicio, es importante](http://developer.android.com/guide/topics/ui/layout-objects.html#viewgroupsummary)

Esta sección describe algunos de los tipos más comunes de los objetos de diseño para utilizar en sus aplicaciones. Al igual que todos los diseños, que son subclases de [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html) .

Véanse también las [visitas Hello](http://developer.android.com/resources/tutorials/views/index.html) tutoriales para alguna orientación sobre el uso de más diseños de vista para Android.

**FrameLayout**

[FrameLayout](http://developer.android.com/reference/android/widget/FrameLayout.html) es el tipo más simple de objetos de diseño. Se trata básicamente de un espacio en blanco en la pantalla que más tarde puede llenar con un solo objeto - por ejemplo, una imagen que va a cambiar dentro y fuera. Todos los elementos secundarios de la FrameLayout se fija en la esquina superior izquierda de la pantalla, no se puede especificar una ubicación diferente para un punto de vista infantil. Visitas posteriores del niño simplemente se dibujan sobre las anteriores, total o parcialmente a oscurecer (a menos que el objeto más reciente es transparente).

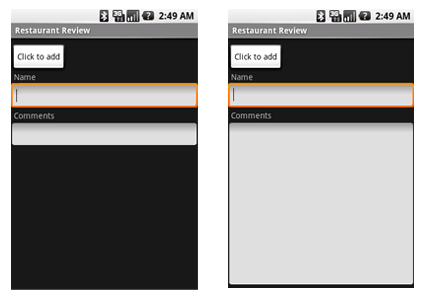
**LinearLayout**

[LinearLayout](http://developer.android.com/reference/android/widget/LinearLayout.html) alinea todos los niños en una sola dirección - vertical u horizontal, dependiendo de cómo se define la orientation atributo. Todos los niños están apilados uno tras otro, por lo que una lista vertical sólo tienen un hijo en cada fila, sin importar su ancho, y una lista horizontal sólo será una fila de alto (la altura de la más alta de niño, además del relleno). Un [LinearLayout](http://developer.android.com/reference/android/widget/LinearLayout.html) s *margen de* aspectos entre los niños y la *gravedad* (centro-derecha, o alineación a la izquierda) de cada niño.

[LinearLayout](http://developer.android.com/reference/android/widget/LinearLayout.html) también es compatible con la asignación de un *peso* a cada niño. Este atributo asigna una "importancia" de valor a un punto de vista, y permite que se expanda para llenar cualquier espacio que quede en la vista de los padres. Puntos de vista del niño puede especificar un valor de peso entero, y entonces cualquier espacio que queda en el grupo de opinión se asigna a los niños en la proporción de su peso declarado. El peso no está en cero. Por ejemplo, si hay tres cuadros de texto y dos de ellos declaran un peso de 1, mientras que el otro se da ningún peso (0), el tercer cuadro de texto sin peso no crecerá y sólo ocupará el área requerida por su contenido. Los otros dos se expandirá igualmente para llenar el espacio que queda después de las tres cajas se miden. Si la tercera casilla se le da un peso de 2 (en lugar de 0), entonces ahora se declara "más importante" que los otros dos, por lo que se pone la mitad del espacio total restante, mientras que los dos primeros comparten el resto igual.

**Sugerencia:** Para crear un diseño de tamaño proporcional en la pantalla, cree una vista contenedor de objeto de grupo con los layout\_width y layout\_height atributos establecidos para *fill\_parent;* asignar a los niños height o la width de 0 (cero), a continuación, asignar relación weight a los valores de cada niño, dependiendo de sobre la proporción de la pantalla de cada uno debe tener.

Los siguientes dos formas representan un [LinearLayout](http://developer.android.com/reference/android/widget/LinearLayout.html) con un conjunto de elementos: un botón, las etiquetas de algunos y cuadros de texto. Los cuadros de texto tienen su anchura a *fill\_parent;* otros elementos se establecen en *wrap\_content.* La gravedad, por defecto, está a la izquierda. La diferencia entre las dos versiones de la forma es que el buscador de la izquierda tiene un peso los valores unset (0 por defecto), mientras que la forma de la derecha tiene el texto de los comentarios de peso de la caja pone a 1. Si el cuadro de texto Nombre también había sido establecido en 1, los cuadros de texto Nombre y comentarios sería la misma altura.



Dentro de un eje horizontal [LinearLayout](http://developer.android.com/reference/android/widget/LinearLayout.html) , los elementos están alineados con la posición de su línea base de texto (la primera línea del elemento de la primera lista - superior o más a la izquierda - se considera la línea de referencia). Esto es para que la gente los elementos de análisis en una forma no debería tener que saltar hacia arriba y hacia abajo para leer el texto en el elemento de los elementos vecinos. Esto se puede desactivar mediante el establecimiento de android:baselineAligned="false" en el diseño de XML.

Para ver otro código de ejemplo, vea el [LinearLayout Hola](http://developer.android.com/resources/tutorials/views/hello-linearlayout.html) tutorial.

**TableLayout**

[TableLayout](http://developer.android.com/reference/android/widget/TableLayout.html) posiciona a sus hijos en filas y columnas. Contenedores TableLayout no se muestran las líneas fronterizas de sus filas, columnas o celdas. La tabla tendrá tantas columnas como la fila con la mayoría de las células. Una tabla puede salir de las celdas vacías, pero las células no pueden abarcar varias columnas, como pueden hacerlo en HTML.

[TableRow](http://developer.android.com/reference/android/widget/TableRow.html) objetos son los puntos de vista de un niño TableLayout (cada TableRow define una única fila en la tabla). Cada fila tiene cero o más células, cada una de las cuales está definida por cualquier tipo de Ver otro. Así, las celdas de una fila puede estar compuesto de una variedad de objetos Ver, como objetos ImageView o TextView. Una célula puede ser también un objeto ViewGroup (por ejemplo, puede anidar otra TableLayout como una célula).

El diseño de la muestra siguiente tiene dos filas y dos celdas en cada uno. La captura de pantalla adjunta muestra el resultado, con bordes de las celdas que se muestran como líneas de puntos (agregado para el efecto visual).

|  |  |
| --- | --- |
| <?xml version="1.0" encoding="utf-8"?>  <TableLayout xmlns: android = "http://schemas.android.com/apk/res/android"  android: layout\_width = "fill\_parent"  android: layout\_height = "fill\_parent"  android: stretchColumns = "1">  <TableRow>  <TextView  android: text = "@ string/table\_layout\_4\_open"  Android: padding = "3DIP" />  <TextView  android: text = "@ string/table\_layout\_4\_open\_shortcut"  Android: la gravedad = "right"  Android: padding = "3DIP" />  </ TableRow>  <TableRow>  <TextView  android: text = "@ string/table\_layout\_4\_save"  Android: padding = "3DIP" />  <TextView  android: text = "@ string/table\_layout\_4\_save\_shortcut"  Android: la gravedad = "right"  Android: padding = "3DIP" />  </ TableRow>  </ TableLayout> | http://developer.android.com/images/table_layout.png |

Las columnas pueden estar escondidos, marcado para estirar y llenar el espacio disponible de la pantalla, o puede ser marcado como contraíble para forzar la columna para reducir el tamaño hasta que la tabla se ajusta a la pantalla. Ver la [TableLayout reference](http://developer.android.com/reference/android/widget/TableLayout.html) documentación para más detalles.

Para ver el código de ejemplo, vea el [TableLayout Hola](http://developer.android.com/resources/tutorials/views/hello-tablelayout.html) tutorial.

**RelativeLayout**

[RelativeLayout](http://developer.android.com/reference/android/widget/RelativeLayout.html) permite puntos de vista del niño especificar su posición en relación con la opinión de los padres o el uno al otro (especificado por ID). Así que usted puede alinear dos elementos por el borde derecho, o hacer uno debajo de otro, centrado en la pantalla, centrado a la izquierda, y así sucesivamente. Los elementos se representan en el orden dado, por lo que si el primer elemento se centra en la pantalla, otros elementos que se alinean a ese elemento se alinea respecto a centro de la pantalla. Además, debido a este ordenamiento, si el uso de XML para especificar esta disposición, el elemento que hará referencia a (a fin de colocar otros objetos vista) debe estar incluido en el archivo XML antes de referirse a él de los otros puntos de vista a través de su ID de referencia .

El siguiente ejemplo muestra un archivo XML y la pantalla que aparece en la interfaz de usuario. Tenga en cuenta que los atributos que hacen referencia a los elementos relativos (por ejemplo, *layout\_toLeft)* se refieren a la identificación con la sintaxis de un recurso relativo *(@ id / id).*

|  |  |
| --- | --- |
| <?xml version="1.0" encoding="utf-8"?>  <RelativeLayout xmlns: android = "http://schemas.android.com/apk/res/android"  android: layout\_width = "fill\_parent"  android: layout\_height = "wrap\_content"  android: background = "@ dibujable / azul"  Android: padding = "10px">  <TextView android: id = "+ id @ / etiqueta"  android: layout\_width = "fill\_parent"  android: layout\_height = "wrap\_content"  android: text = "Escriba aquí" />  <EditarTexto android: id = "+ id @ / entrada"  android: layout\_width = "fill\_parent"  android: layout\_height = "wrap\_content"  android: background = "@ android: dibujable / editbox\_background"  android: layout\_below = "@ id / etiqueta" />    <Botón de android: id = "+ id @ / OK"  android: layout\_width = "wrap\_content"  android: layout\_height = "wrap\_content"  android: layout\_below = "@ id / entrada"  android: layout\_alignParentRight = "true"  android: layout\_marginLeft = "10px"  android: text = "Aceptar" />  <Botón de Android: layout\_width = "wrap\_content"  android: layout\_height = "wrap\_content"  android: layout\_toLeftOf = "@ id / ok"  android: layout\_alignTop = "@ id / ok"  android: text = "Cancelar" />  </ RelativeLayout> | http://developer.android.com/images/designing_ui_layout_example.png |

Algunas de estas propiedades están soportados directamente por el elemento, y algunos son apoyados por sus Estados miembros LayoutParams (RelativeLayout subclase de todos los elementos de esta pantalla, porque todos los elementos son hijos de un objeto primario RelativeLayout). Los parámetros RelativeLayout definidos son: width , height , below , alignTop , toLeft , padding[Bottom|Left|Right|Top] , y margin[Bottom|Left|Right|Top] . Tenga en cuenta que algunos de estos parámetros que apoyan específicamente la posición relativa de diseño - sus valores deben ser el identificador del elemento al que le gustaría que este punto de vista establecido relativa. Por ejemplo, asignar el parámetro toLeft="my\_button" a una TextView pondría a la Vista de Texto a la izquierda de la vista con la *my\_button* ID (que debe estar escrito en el XML *antes de* la Vista de Texto).

Para ver este código de ejemplo, vea el [RelativeLayout Hola](http://developer.android.com/resources/tutorials/views/hello-relativelayout.html) tutorial.

**Resumen de los Grupos de juicio, es importante**

Estos objetos tienen todos los elementos secundarios de interfaz de usuario. Algunas de ellas ofrecen su propia forma de una interfaz de usuario visible, mientras que otros son estructuras invisibles que sólo gestionan el diseño de puntos de vista a su hijo.

|  |  |
| --- | --- |
| **Exact.** | **Descripción** |
| [FrameLayout](http://developer.android.com/reference/android/widget/FrameLayout.html) | Disposición que actúa como marco de la vista para mostrar un único objeto. |
| [Gallery](http://developer.android.com/reference/android/widget/Gallery.html) | Una pantalla de desplazamiento horizontal de las imágenes, de una lista enlazada. |
| [GridView](http://developer.android.com/reference/android/widget/GridView.html) | Muestra una cuadrícula de desplazamiento de m columnas y filas n. |
| [LinearLayout](http://developer.android.com/reference/android/widget/LinearLayout.html) | Un diseño que organiza sus hijos en una sola fila horizontal o vertical. Se crea una barra de desplazamiento si la longitud de la ventana supera la longitud de la pantalla. |
| [ListView](http://developer.android.com/reference/android/widget/ListView.html) | Muestra una lista de desplazamiento de una sola columna. |
| [RelativeLayout](http://developer.android.com/reference/android/widget/RelativeLayout.html) | Le permite especificar la ubicación de los objetos secundarios respecto a la otra (el niño de A a la izquierda de B infantil). O el padre (alineado a la parte superior de la matriz) |
| [ScrollView](http://developer.android.com/reference/android/widget/ScrollView.html) | Una columna de desplazamiento vertical de los elementos. |
| [Spinner](http://developer.android.com/reference/android/widget/Spinner.html) | Muestra un solo elemento a la vez de una lista unida, dentro de un cuadro de texto de una fila. Algo así como un cuadro de lista de una sola fila que puede desplazarse horizontal o verticalmente. |
| [SurfaceView](http://developer.android.com/reference/android/view/SurfaceView.html) | Proporciona acceso directo a una superficie de dibujo dedicado. Puede llevar a cabo visitas de niños colocados en capas en la parte superior de la superficie, pero está diseñado para aplicaciones que necesitan dibujar píxeles, en lugar de usar widgets. |
| [TabHost](http://developer.android.com/reference/android/widget/TabHost.html) | Proporciona una lista de selección de ficha que supervisa los clics y permite la aplicación para cambiar la pantalla cada vez que una ficha se hace clic. |
| [TableLayout](http://developer.android.com/reference/android/widget/TableLayout.html) | Un diseño de tablas con un número arbitrario de filas y columnas, cada celda el widget de su elección. Las filas de tamaño para ajustarse a la más grande de la columna. Los bordes de las celdas no son visibles. |
| [ViewFlipper](http://developer.android.com/reference/android/widget/ViewFlipper.html) | Una lista que muestra un elemento a la vez, dentro de un cuadro de texto de una fila. Se puede configurar para cambiar los productos a intervalos de tiempo, como una presentación de diapositivas. |
| [ViewSwitcher](http://developer.android.com/reference/android/widget/ViewSwitcher.html) | Igual que ViewFlipper. |

**¿Cómo Android Dibuja Vistas**

Cuando una actividad recibe el foco, se solicita a elaborar su diseño. El marco de Android se encargará del procedimiento para el dibujo, pero la actividad debe proporcionar el nodo raíz de su jerarquía de diseño.

Dibujo comienza con el nodo raíz de la presentación. Se solicita para medir y dibujar el árbol de diseño. Dibujo se maneja a pie del árbol y la prestación de cada vista que cruza la región no válido. A su vez, cada grupo de Vista es responsable de solicitar cada uno de sus hijos a ser dibujado (con el [draw()](http://developer.android.com/reference/android/view/View.html#draw%28android.graphics.Canvas%29) método) y cada vista es responsable de la elaboración propia. Debido a que el árbol se recorre en orden, esto significa que los padres se elaborará antes (es decir, detrás de) sus hijos, con los hermanos establecidos en el orden en que aparecen en el árbol.

El marco no va a llamar la vistas que no son válidos en la región, y también se encargará de dibujar el fondo Vistas para usted.

Puede forzar a una vista para dibujar, llamando a [invalidate()](http://developer.android.com/reference/android/view/View.html#invalidate%28%29) .

Dibujar el diseño es un proceso de dos pasos: un paso de la medida y un paso de diseño. El pase de la medición se lleva a cabo en la [measure(int, int)](http://developer.android.com/reference/android/view/View.html#measure%28int,%20int%29) y es un recorrido de arriba hacia abajo de la vista de árbol. Cada uno empuja Ver especificaciones de dimensión por el árbol durante la recursividad. Al final de la pasada de medida, cada vista ha almacenado sus mediciones. El segundo paso que sucede en [layout(int, int, int, int)](http://developer.android.com/reference/android/view/View.html#layout%28int,%20int,%20int,%20int%29) y también es de arriba hacia abajo. Durante esta fase de cada padre es responsable de posicionar a todos sus hijos con los tamaños calculados en el paso de la medida.

Cuando una vista de la measure() devuelve el método, su [getMeasuredWidth()](http://developer.android.com/reference/android/view/View.html#getMeasuredWidth%28%29) y [getMeasuredHeight()](http://developer.android.com/reference/android/view/View.html#getMeasuredHeight%28%29) los valores se debe establecer, junto con las de todos los descendientes de esa vista. Ancho, medidos Un punto de vista y valores de medición de altura deben respetar las limitaciones impuestas por los padres de la vista. Esto garantiza que, al final del paso de la medida, todos los padres aceptar todos mediciones de sus hijos. Un punto de vista los padres pueden llamar a measure() más de una vez en sus hijos. Por ejemplo, el padre puede medir a cada niño una vez con las dimensiones especificadas para averiguar qué tan grande quieren ser, entonces llame a measure() sobre ellos de nuevo con los números reales, si la suma de todos los tamaños de los niños sin restricciones es demasiado grande o demasiado pequeño ( es decir, si los niños no se ponen de acuerdo entre sí en cuanto a la cantidad de espacio que cada uno recibe, el padre va a intervenir y establecer las normas relativas a la segunda pasada).

Para iniciar una presentación, llame [requestLayout()](http://developer.android.com/reference/android/view/View.html#requestLayout%28%29) . Este método se suele llamar por una visión sobre sí mismo cuando se cree que es posible ya no encajan dentro de sus límites actuales.

El paso de la medida se utilizan dos clases de comunicar dimensiones. La [View.MeasureSpec](http://developer.android.com/reference/android/view/View.MeasureSpec.html) clase se utiliza vistas para decirle a sus padres cómo quieren ser medido y colocado. La clase base LayoutParams sólo describe el tamaño del Vista quiere ser para la anchura y la altura. Para cada dimensión, se puede especificar uno de:

* un número exacto
* *FILL\_PARENT,* lo que significa la Vista quiere ser tan grande como su padre (menos el relleno)
* *WRAP\_CONTENT,* lo que significa que la Vista quiere ser lo suficientemente grande para encerrar su contenido (además de relleno).

Hay subclases de LayoutParams de las diferentes subclases de ViewGroup. Por ejemplo, RelativeLayout tiene su propia subclase de LayoutParams, que incluye la capacidad de las opiniones del niño del centro de forma horizontal y vertical.

MeasureSpecs se utilizan para empujar las necesidades de todo el árbol de padres a hijos. Un MeasureSpec puede estar en uno de tres modos:

* *Sin especificar:* Este es utilizado por uno de los padres para determinar la dimensión deseada de una vista de niño. Por ejemplo, un LinearLayout puede llamar a measure() en su hijo con el conjunto de altura *a* Unspecified y una anchura de *exactamente* 240 para averiguar la altura de la vista niño quiere dar un ancho de 240 pixeles.
* *EXACTAMENTE:* Esto se utiliza por los padres para imponer una medida exacta en el niño. El niño debe usar este tamaño, y garantizar que todos sus descendientes se ajusta dentro de este tamaño.
* *AT\_MOST:* Esta se utiliza por los padres para imponer un tamaño máximo en el niño. El niño debe garantizar que él y todos sus descendientes se ajusta dentro de este tamaño.