

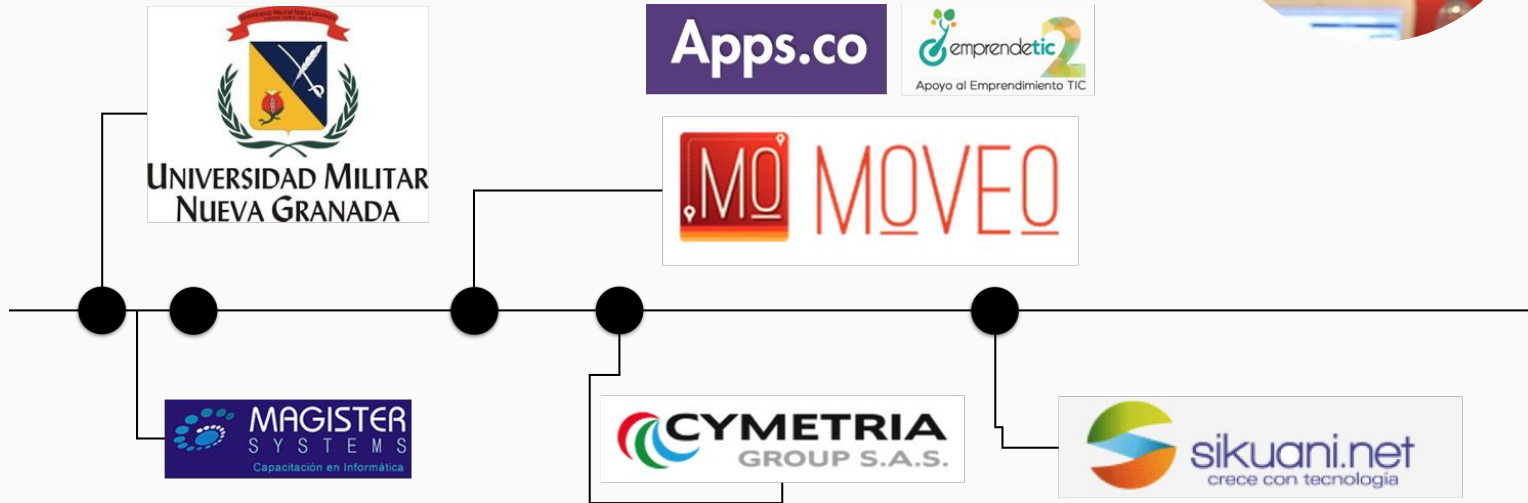
# Android Básico-Intermedio

BiccoFarms, Octubre 2015  
Instructor: Julián R Figueroa  
Cymetria Training



# Instructor: Julián R Figueroa

@unmultimedia



#Entrepreneurship, #LeanStartup, #Geek, #HCIDev,  
#TechDev, #Bycicle, #Sustainability.



# Agenda

- Historia
- Workflow
- “Hello World”
- Ciclo de vida
- Action Bar
- XML Layouts
- Menús
- Persistencia
- IDE/Debug
- Mapas y Geolocalización
- Fragmentos
- Interfaces
- Notificaciones
- Consumo APIs
- GCM
- Q&A

# Android: Historia

- Android Inc. (2005 Google)
- +1MM diarios
- Open source
- C | C++ | Java
- Construido y compilado sobre Linux (apps como usuarios)
- DB relacional (SQLite)
- OpenGL (v2.0, v3.0)
- Mercado móvil 2007 (iPhone)
- C2DM (Push)
- WebKit (Chromium)
- GPS, acelerómetro, giroscopio, proximidad, luz.
- Emulador sobre Eclipse\*
- Multitarea real desde GB



# Android: Versiones

CODENAME	ABR.	CODENAME	ABR.
Apple Pie?		Honeycomb	HC
Banana Bread?		Ice Cream Sandwich	ICS
Cupcake	C	Jelly Bean	JB
Donut	D	KitKat	KK
Eclair	E	Lollipop	LP
Froyo	F	Marshmallow	MM
Gingerbread	GB		

<http://developer.android.com/guide/topics/manifest/uses-sdk-element.html>

# Android: Versiones y APIs

API	V	CODENAME	ABR.	API	V	CODENAME	ABR.
1	1.0	Apple Pie?		11-13	3.x	Honeycomb	HC
2	1.1	Banana Bread?		14-15	4.0.x	Ice Cream Sandwich	ICS
3	1.5	Cupcake	C	16-18	4.1.x 4.2.x 4.3.x	Jelly Bean	JB
4	1.6	Donut	D	19-20*	4.4	KitKat	KK
5-7	2.0 2.0.1	Eclair	E	21-22	5.x	Lollipop	LP
8	2.2.x	Froyo	F	23	6.0	Marshmallow	MM
9-10	2.3.x	Gingerbread	GB				

# Flujo de trabajo (Workflow) DDD

- IDE
- Dispositivos, emuladores
- Test HelloApp
- Code!!!
- Desplegar
- Depuración
- Firmar, test a producción
- Alfa, beta, (privadas, públicas)
- Publicar, promocionar



Developers ▾

Design

Develop

Distribute

## Workflow ^

Setting Up Virtual Devices ▾

Using Hardware Devices ▾

Managing Projects ▾

Building and Running ▾

Testing ▾

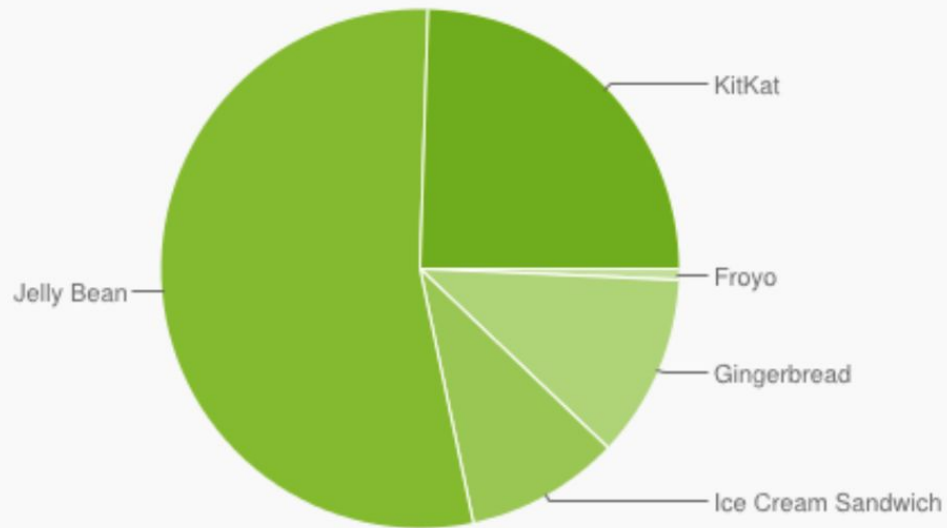
Debugging ▾

Publishing ▾



# Workflow: Soporte

Version	Codename	API	Distribution
2.2	Froyo	8	0.7%
2.3.3 - 2.3.7	Gingerbread	10	11.4%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	9.6%
4.1.x	Jelly Bean	16	25.1%
4.2.x		17	20.7%
4.3		18	8.0%
4.4	KitKat	19	24.5%

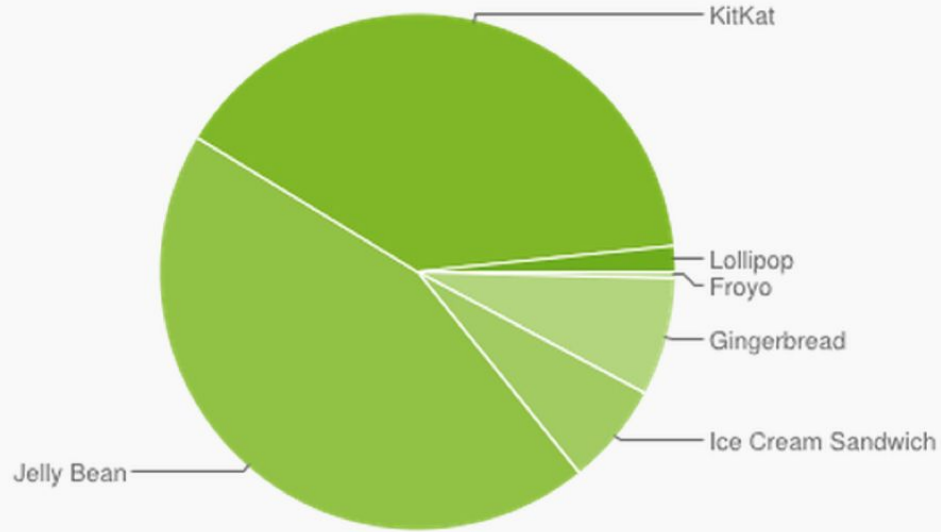


- Versiones
- Densidades y tamaños de pantalla
- Versión de OpenGL

2014-12

# Workflow: Soporte

Version	Codename	API	Distribution
2.2	Froyo	8	0.4%
2.3.3 - 2.3.7	Gingerbread	10	7.4%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	6.4%
4.1.x	Jelly Bean	16	18.4%
4.2.x		17	19.8%
4.3		18	6.3%
4.4	KitKat	19	39.7%
5.0	Lollipop	21	1.6%

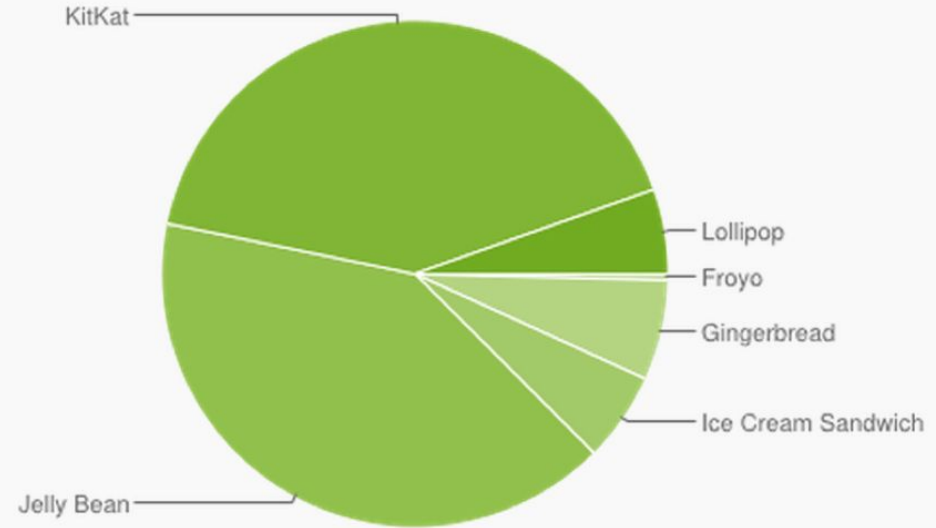


- Versiones
- Densidades y tamaños de pantalla
- Versión de OpenGL

2015-03

# Workflow: Soporte

Version	Codename	API	Distribution
2.2	Froyo	8	0.4%
2.3.3 - 2.3.7	Gingerbread	10	6.4%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	5.7%
4.1.x	Jelly Bean	16	16.5%
4.2.x		17	18.6%
4.3		18	5.6%
4.4	KitKat	19	41.4%
5.0	Lollipop	21	5.0%
5.1		22	0.4%

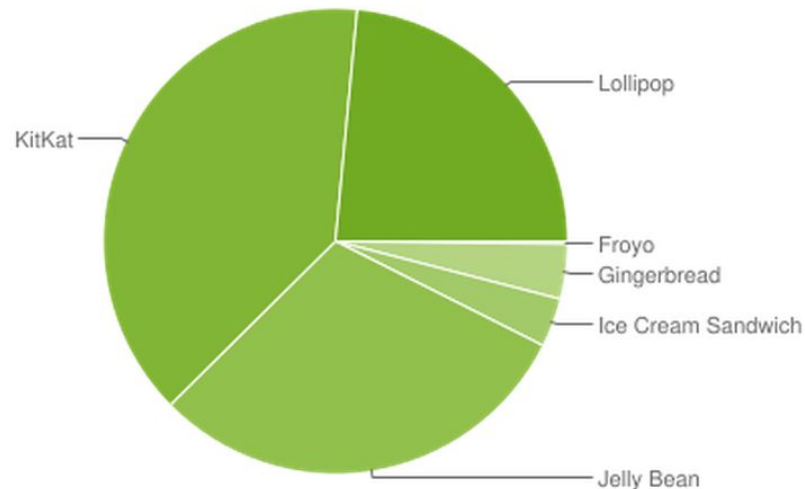


- Versiones
- Densidades y tamaños de pantalla
- Versión de OpenGL

2015-06

# Workflow: Soporte

Version	Codename	API	Distribution
2.2	Froyo	8	0.2%
2.3.3 - 2.3.7	Gingerbread	10	3.8%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	3.4%
4.1.x	Jelly Bean	16	11.4%
4.2.x		17	14.5%
4.3		18	4.3%
4.4	KitKat	19	38.9%
5.0	Lollipop	21	15.6%
5.1		22	7.9%



- Versiones
- Densidades y tamaños de pantalla
- Versión de OpenGL

2015-10

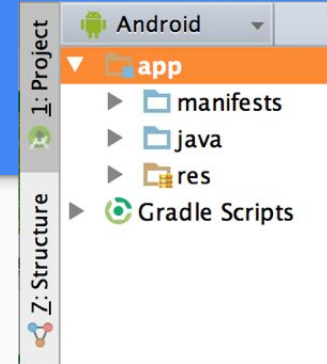
## Workflow: Requisitos

- Android Studio / Eclipse con ADT Oficial
- Acceso a la documentación de referencia ([developer.android.com](http://developer.android.com))
- Equipo Android de depuración
  - Windows OEM Driver
  - Linux `"/etc/udev/rules.d/51-android.rules"`
  - MacOSX :)
- Emulador con la imagen de OS correcta
  - Android Studio AVD
  - Genymotion
  - BlueStacks

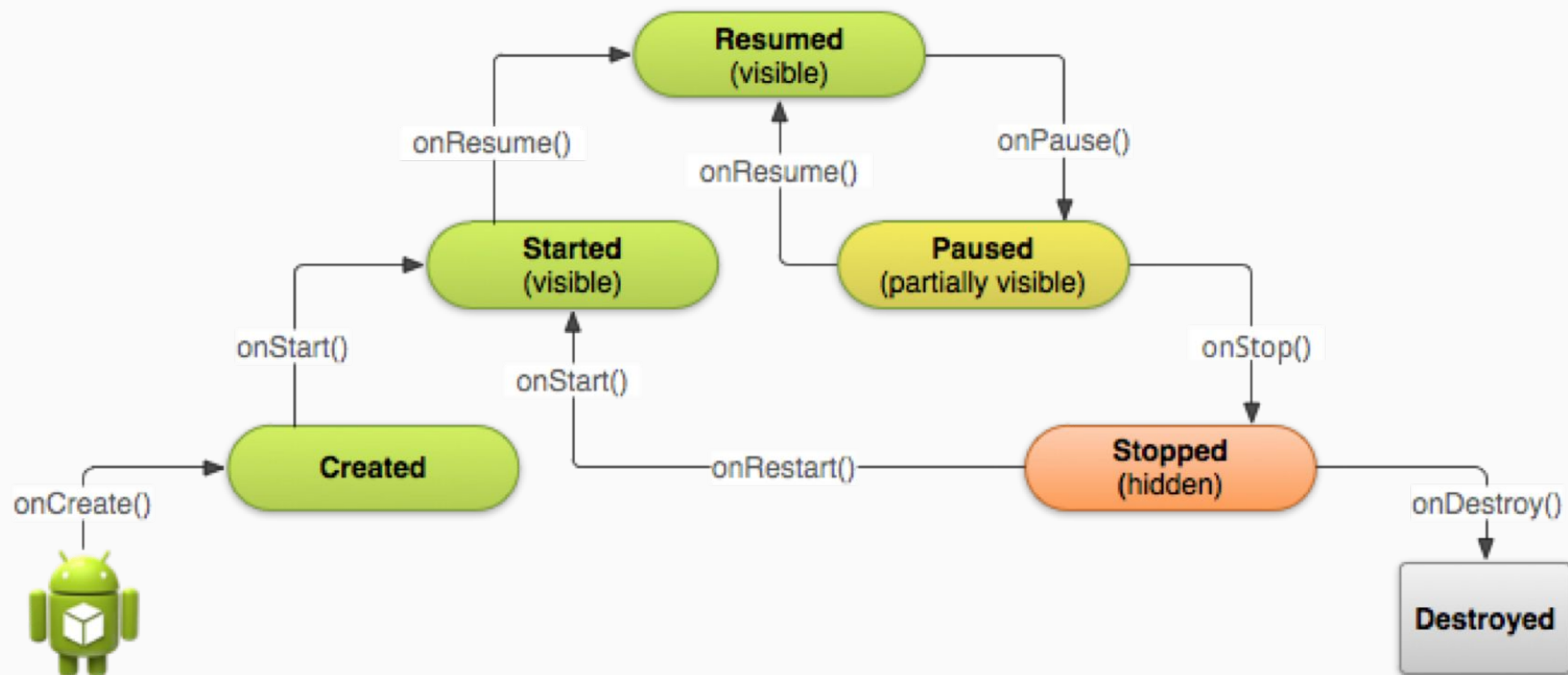
# “Hello World”

A tener en cuenta:

- Actividades
- Recursos
  - Layouts
  - Strings
- Toast
- Intent



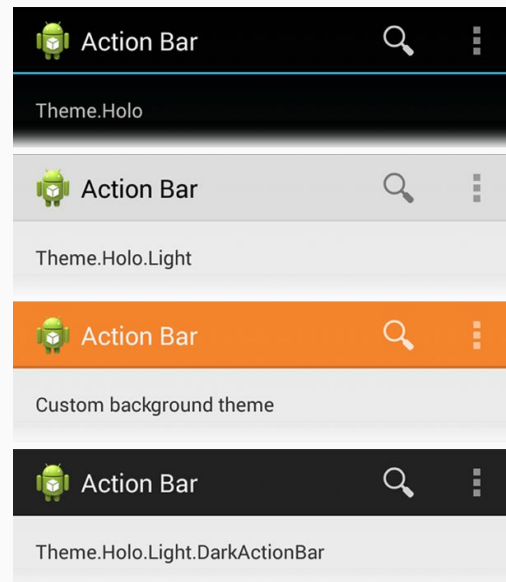
# Ciclo de vida (Lifecycle)



# ActionBar

Versión de soporte? → AppCompat

- Menús
- Íconos como acciones
- Estilos
- Recursos





# ActionBar: Material Design

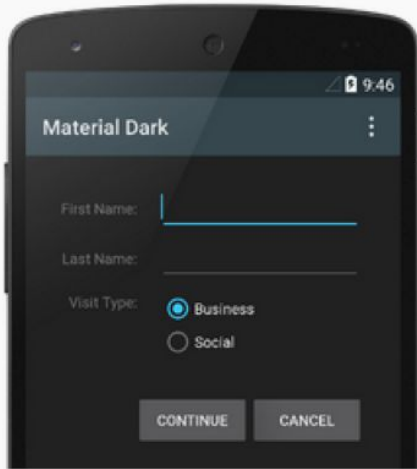


Figure 1. Dark material theme

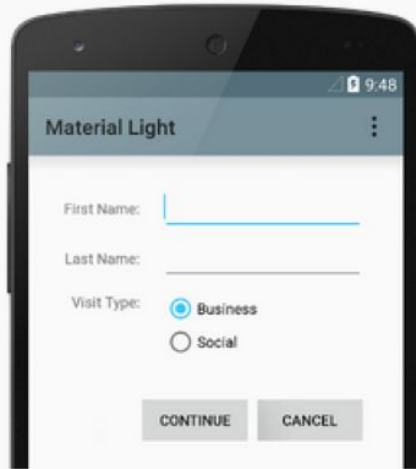
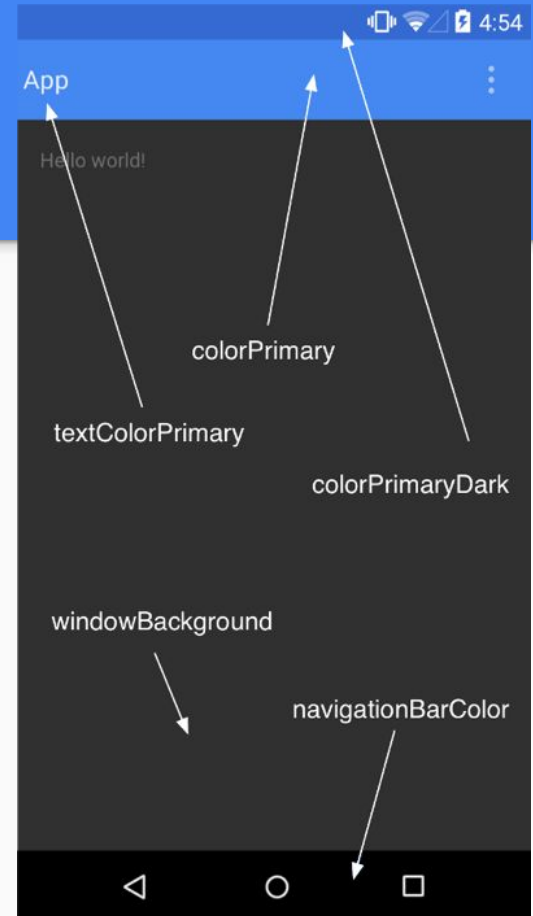
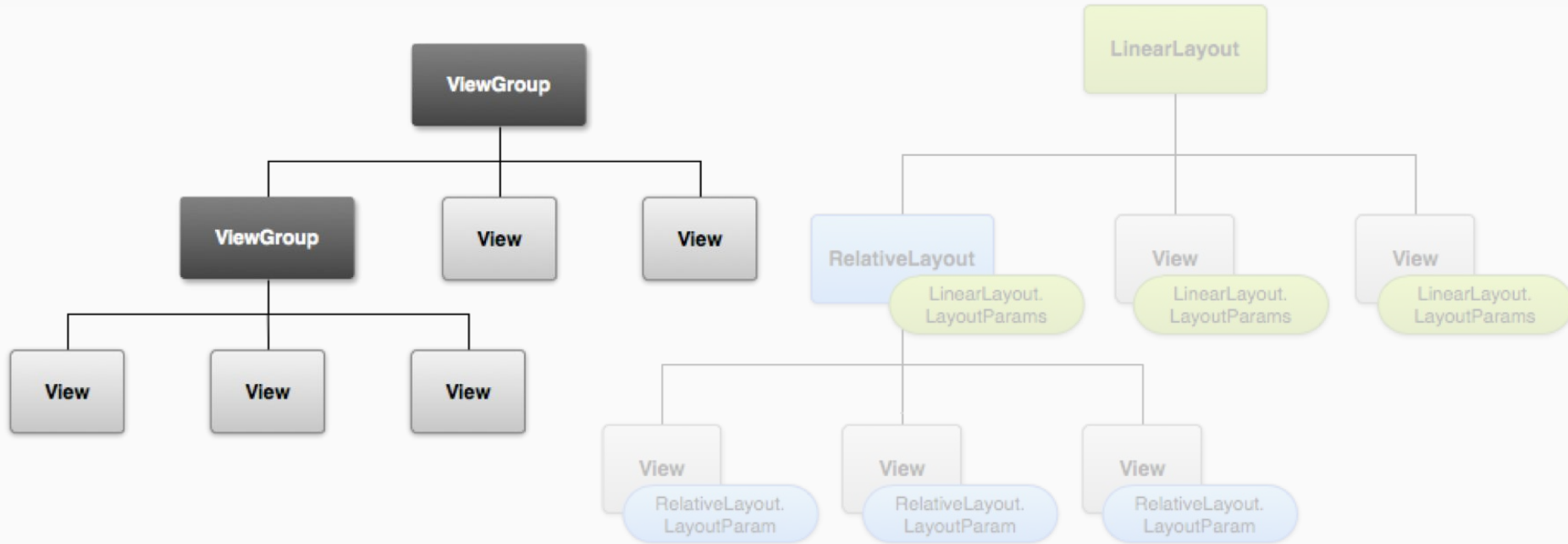


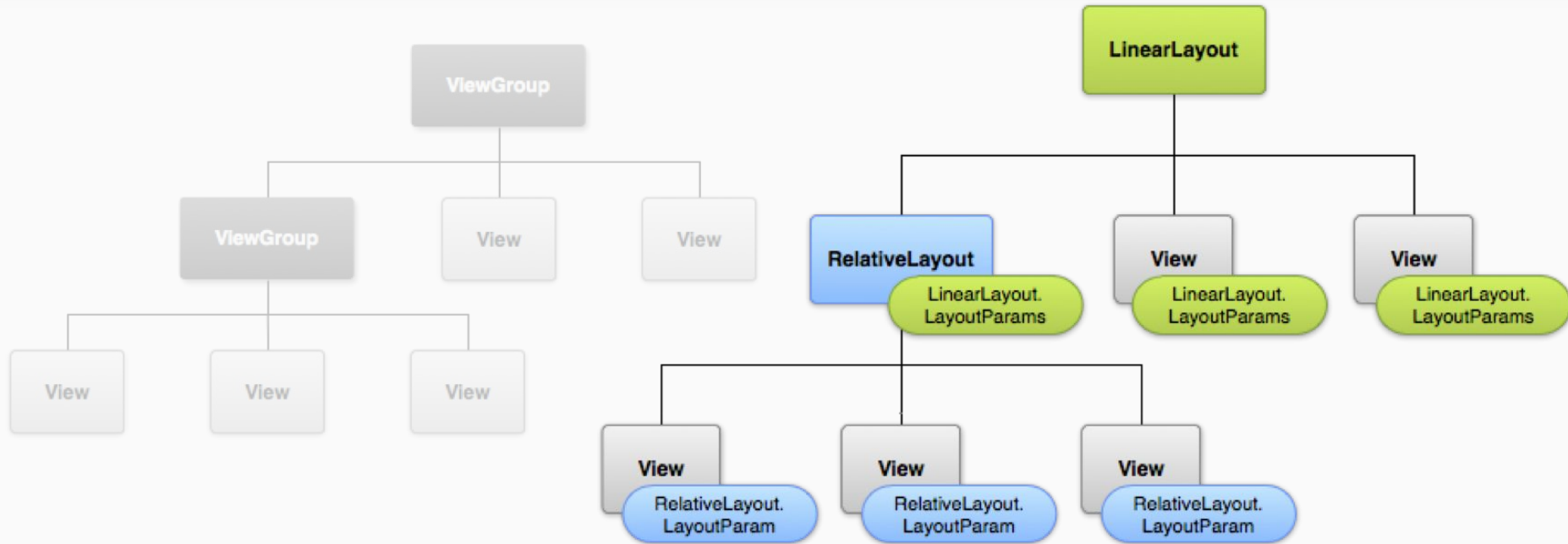
Figure 2. Light material theme



# Layouts: Vistas y Grupos de Vistas



# Layouts: Vistas y Grupos de Vistas



# Layouts: Organizaciones Comunes

- Linear
- Relative
- WebView
- ListView
- GridView

Linear Layout



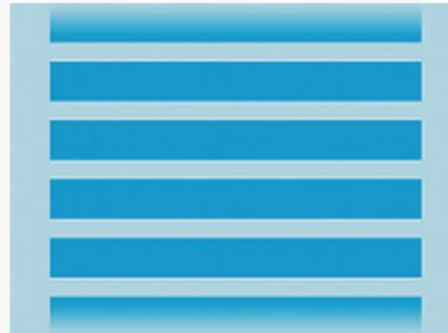
Relative Layout



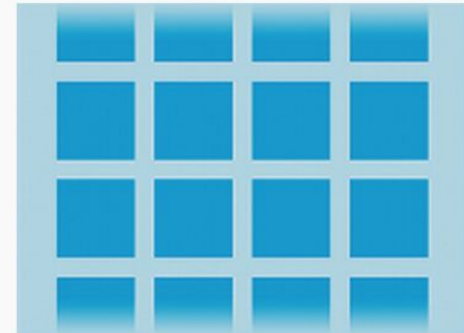
Web View



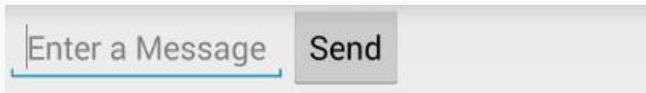
List View



Grid View



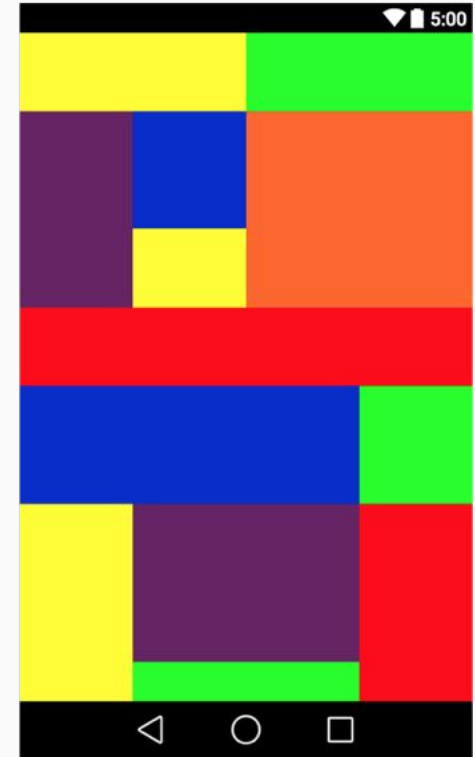
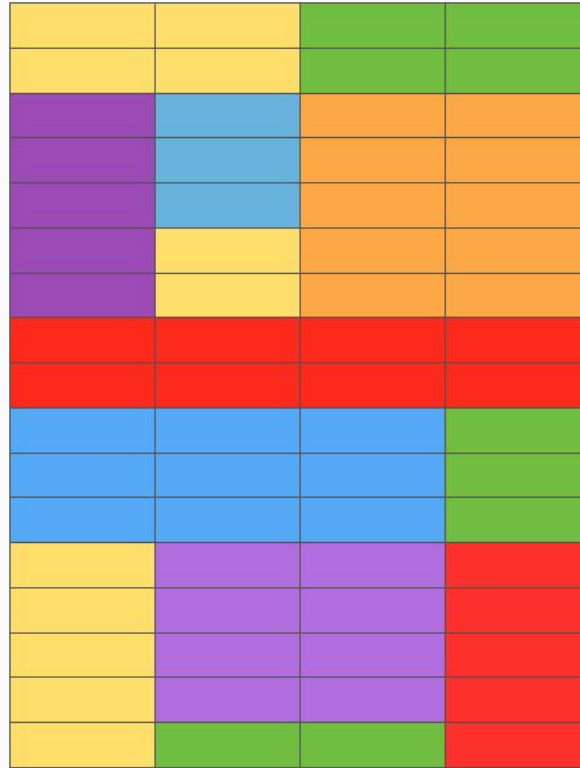
## ¿Cómo ajustar el porcentaje como se hace en web?



```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">
    <EditText android:id="@+id/edit_message"
        android:layout_weight="1"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:hint="@string/edit_message" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/button_send" />
</LinearLayout>
```

## Ejercicio Layout: Pesos

- LinearLayout
- No hay medidas estáticas
- Orientaciones y pesos
- ¿Organización?



# Extras: Parámetros en los Intent

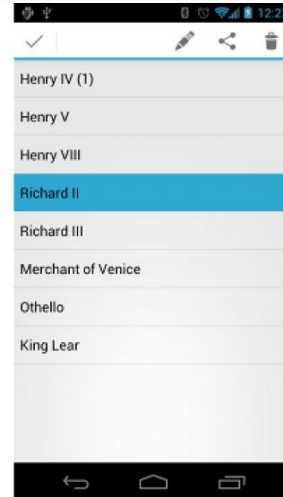
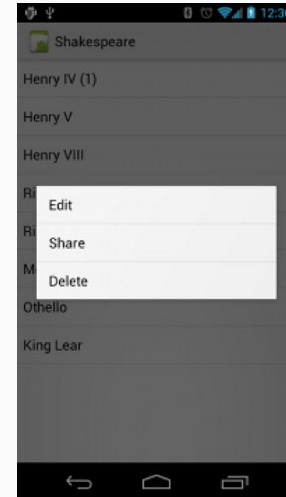
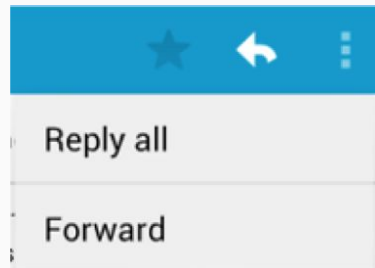
Los Intent, al igual que los llamados de un form en web, pueden llevar parámetros.

Tipos de datos primitivos + String

Enviar texto y color

# Menús

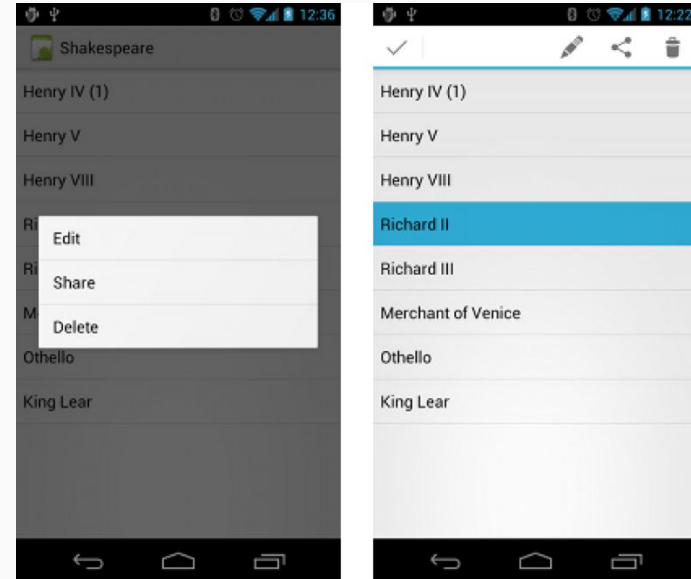
- Opciones (Acciones) del ActionBar
- Menús contextuales de acción
- Menú tipo Pop-up





# Menús Contextuales: Ejercicio

- Listas, adaptadores
- Actualización de lista desde campo de texto
- Menú contextual por ítem
- Actualización de listas



# Persistencia en Android

Almacenamiento persistente de información:

- Preferencias compartidas
- Archivos
- Base de datos SQLite

# Preferencias Compartidas

Diccionario/Mapa de primitivas

Almacenado en un archivo 'ruta-app/prefs'

Cargado en el ciclo de vida

getEditor(), put...(), commit()

MODE\_WORLD\_READABLE,  
MODE\_WORLD\_WRITEABLE, MODE\_PRIVATE



# Ejercicio: Preferencias Compartidas

Almacenar los siguientes valores:

- Nombre (String)
- Edad (int)
- Solter@ (boolean)

SharedPreferences

Ingresa los datos de las variables

Variable caracter  Guardar

Variable entera  Guardar

☒ Variable booleana Guardar

Limpiar

Variable caracter  
**Huila**

Variable entera  
**58**

Variable booleana  
**true**

# Base de datos: SQLite

Paquete contenedor: 'android.database.sqlite'

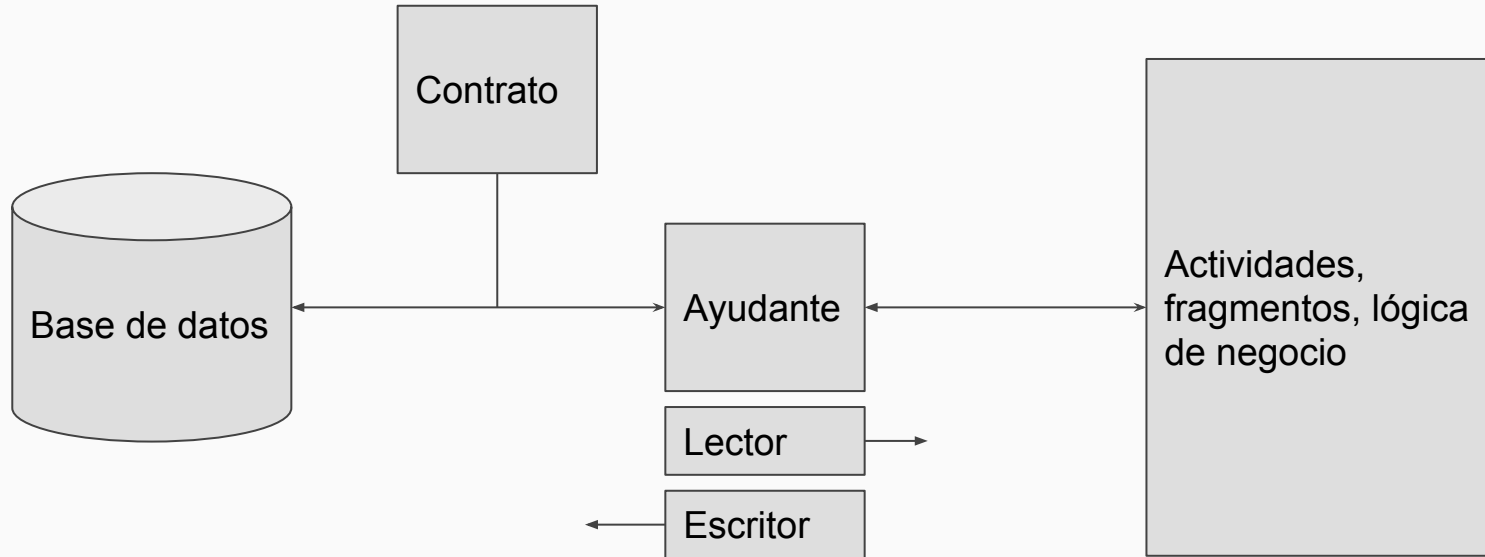
Clase contrato

Constante de nombres de tablas, vistas,  
columnas, tipos de datos

Contrato de replicación de tablas y secuencias



# Bases de datos: SQLite



```
public class DatabaseContract {  
    //Tabla Usuarios  
    public static class Users implements BaseColumns{  
        public static final String TABLE_NAME = "users";  
        public static final String COLUMN_NAME_NAME = "name";  
        public static final String COLUMN_NAME_DRINK = "drink";  
        public static final String COLUMN_NAME_SPORT = "sport";  
    }  
    // Otras tablas, vistas...  
}
```

# SQLite: Ayudante

```
public class Ayudante extends SQLiteOpenHelper {

    public static final String DATABASE_NAME = "nombrebase.db";
    public static final int DATABASE_VERSION = 1;
    public static final String SQL_CREATE_USERS =
        "CREATE TABLE " + Users.TABLE_NAME
        + " (" + Users._ID + " INTEGER PRIMARY KEY, "
        + Users.COLUMN_NAME_NAME + " TEXT, "
        + Users.COLUMN_NAME_DRINK + " TEXT, "
        + Users.COLUMN_NAME_SPORT + " TEXT)";
    public static final String SQL_DELETE_USERS =
        "DROP TABLE IF EXISTS " + Users.TABLE_NAME;

    // Otras Sentencias ...
    // Las bases de datos, una para leer, una para escribir datos
    SQLiteDatabase escritor;
    SQLiteDatabase lector;
    public Ayudante(Context context){
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }
}
```



## SQLite: ¿Cómo insertar un usuario?

```
public boolean insertarUsuario(Usuario nuevo) {  
    ContentValues values = new ContentValues();  
  
    values.put(Users.COLUMN_NAME_NAME, nuevo.name);  
    values.put(Users.COLUMN_NAME_DRINK, nuevo.drink);  
    values.put(Users.COLUMN_NAME_SPORT, nuevo.sport);  
  
    long inserted = escritor.insert(  
        Users.TABLE_NAME,  
        Users.COLUMN_NAME_NAME,  
        values);  
  
    if(inserted == -1) return false;  
    return true;  
}
```

## SQLite: ¿Cómo consultar un usuario?

```
public List<Usuario> consultarUsuarios(){

    String[] columns = {Users._ID, Users.COLUMN_NAME_NAME,
        Users.COLUMN_NAME_DRINK, Users.COLUMN_NAME_SPORT};

    String selection = null; //Users.COLUMN_NAME_NAME + " Like ?";
    String selectionArgs[] = null; //{ "%a%"};
    String groupBy = null; //Users.COLUMN_NAME_SPORT;
    String having = null; //condición aritmética
    String orderBy = null; //Users._ID;
    String limit = null; //"10";

    Cursor results = lector.query(Users.TABLE_NAME, columns,
        selection, selectionArgs, groupBy, having, orderBy, limit);
    // results ya es un cursor con los datos de regreso
}
```

# SQLite: ejercicio

Hacer un formulario, para ingreso de usuarios

- Nombre
- Bebida
- Deporte

Agregar usuarios a la base

Mostrarlos en una lista

Eliminarlos de la tabla, actualizar

# Persistencia: Archivos

Directorios tipo File, sin importar si son archivos o rutas

Imágenes, archivos susceptibles de ser compartidos por red, borradores de composiciones (correos), borradores de archivos de configuración



```
<uses-permission
```

```
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

```
<uses-permission
```

```
android:name="android.permission.READ_EXTERNAL_STORAGE" />
```

# Archivos: Memoria Interna

Siempre está disponible por defecto. Se puede cambiar dirección de instalación con android:  
`installLocation`

Por defecto accede solo el app

Cuando el usuario des-instala el app, los archivos se eliminan

La mejor opción para información privada o de sentido solo para el app

# Archivos: Memoria Externa

No siempre está disponible

No tiene recursos, suele ser `WORLD_READABLE` y estar en los directorios compartidos

Sólo se eliminan los archivos al des-instalar cuando se guardan en el directorio privado del app `getExternalDir()`

Mejor opción para archivos de almacenamiento compartido como archivos estándar multimedia, ó documentos de ofimática

# Archivos: ejercicio

Guardar un título y contenido en local interna

Capturar la información de un formulario

Almacenar en el dispositivo

Verificar su contenido

# IDE: Depuración

Escoger punto de depuración

Iniciar la ejecución siempre con el teléfono conectado o con emulador

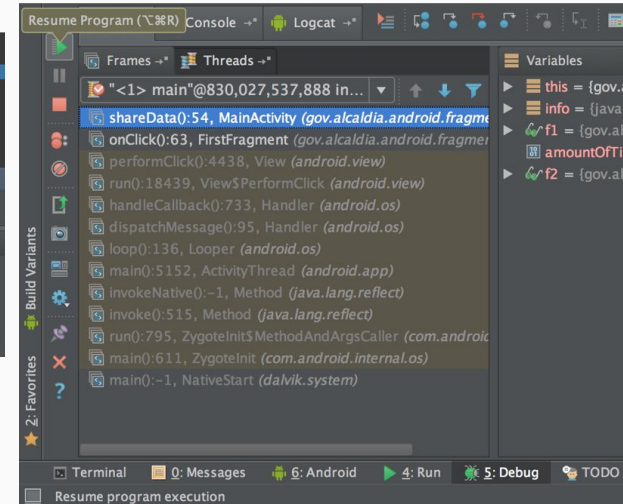
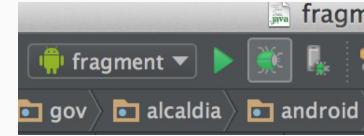
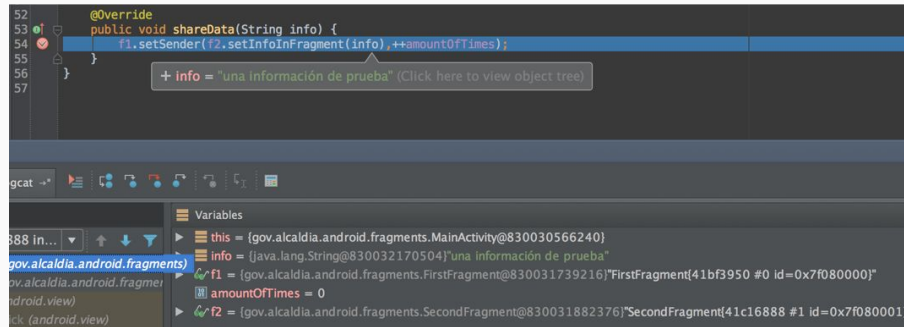
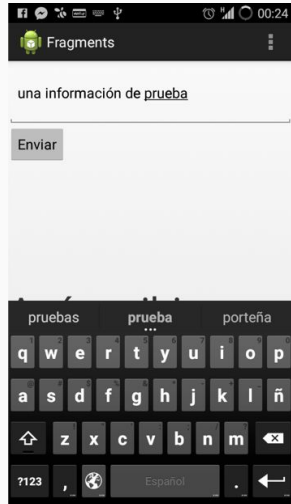
Ejecutar con la instrucción de 'Depurar'

El depurador se ejecuta de manera normal como otros IDEs



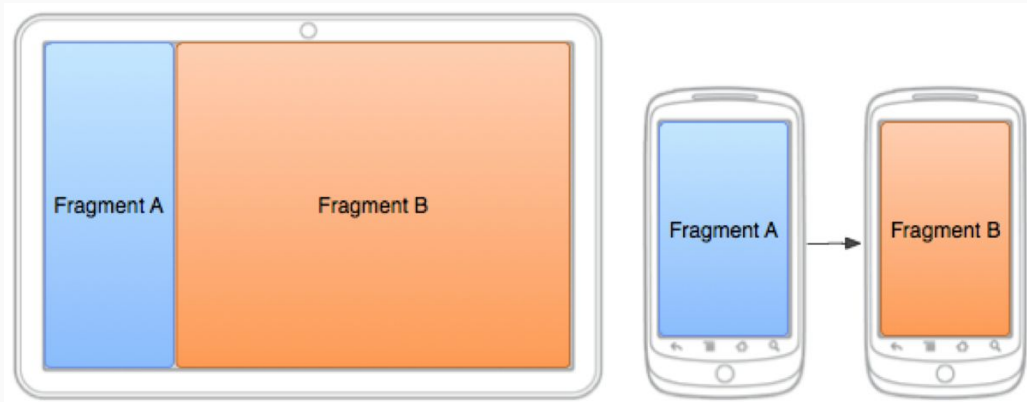
# IDE: Depuración

```
52 @Override
53 public void shareData(String info) {
54     f1.setSender(f2.setInfoInFragment(info), ++amountOfTimes);
55 }
56 }
```



# Fragmentos

- Nuevo estándar
- Reutilización de código
- Ciclo de vida propio
- Transacciones
- Diseño multi-screen
  - Teléfonos
  - Tabletas
  - Wear



Tablet

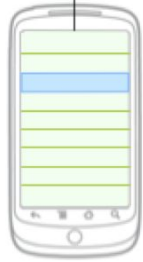
Selecting an item  
updates Fragment B



Activity A contains  
Fragment A and Fragment B

Handset

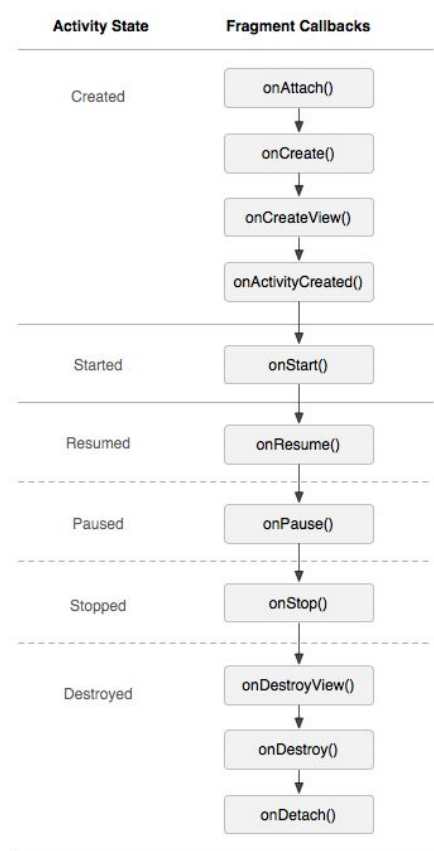
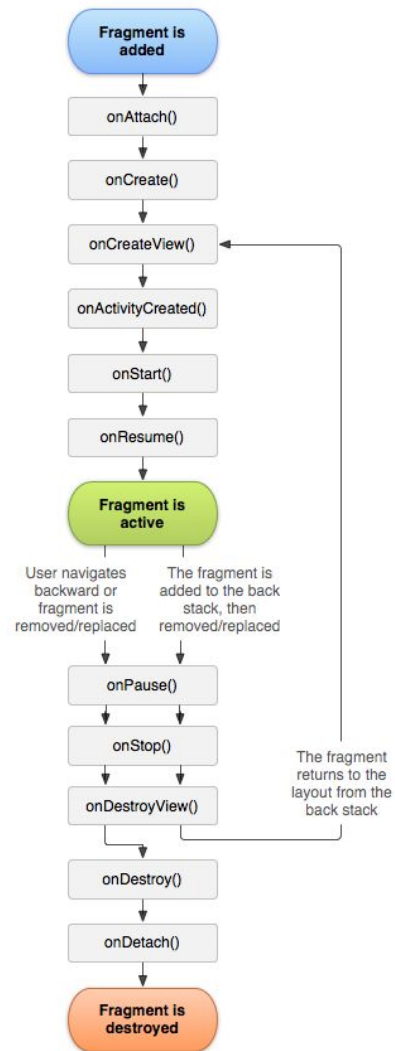
Selecting an item  
starts Activity B



Activity A contains  
Fragment A



Activity B contains  
Fragment B

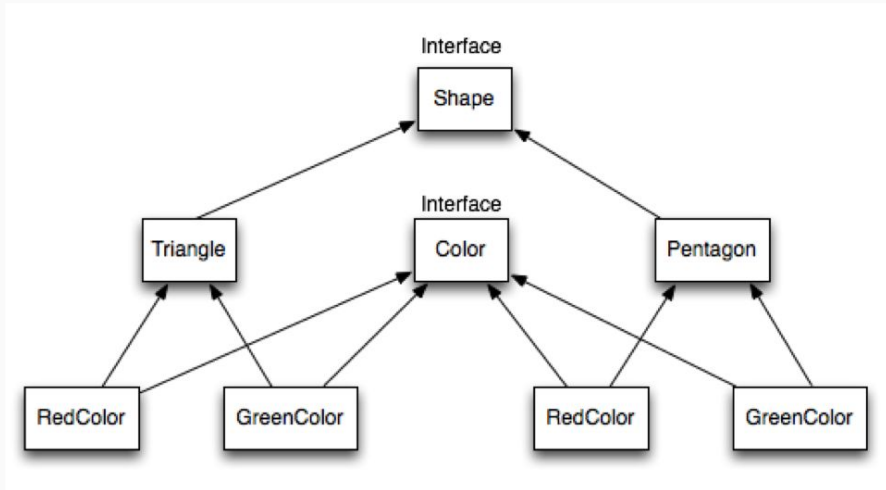


# Fragmentos: Interfaces

Manera segura de compartir información

getActivity() pasa usos estándar

La interfaz exige un estándar para procesos propios, facilita colaboración



# Ejercicio Fragmentos

Las mismas listas del ejercicio de listas

El input va en un fragmento, la lista en otro

La comunicación a través de interfaces

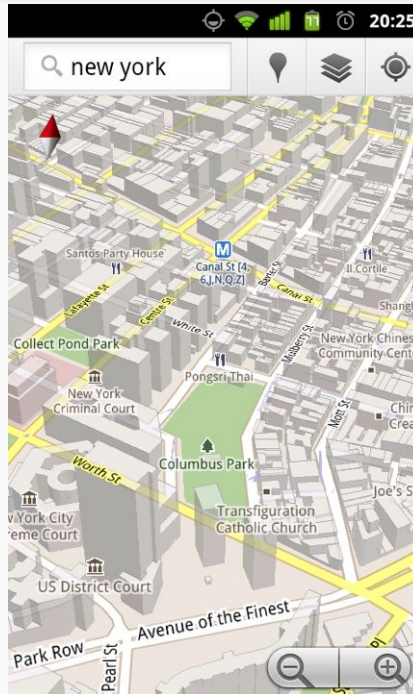
La impresión de la lista es personalizada

Debe haber comprobación de duplicados

# Mapas y Geolocalización

## Mapas:

- Fragmentos
- Conexión con cuenta de desarrollador
- Firma SHA1



## Geolocalización:

- Independiente de mapas
- Permisos en manifiesto
  - Google Play Services
- Diferencia entre clases Location y LatLng

# Geolocalización: Manifiesto

```
<uses-permission
android:name="android.permission.INTERNET"/>
<uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission
android:name=
"com.google.android.providers.gsf.permission.READ_GSERVICES"/>

<!-- The following two permissions are not required to use
Google Maps Android API v2, but are recommended. -->
<uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION"/>
```

```
import com.google.android.gms.maps.MapFragment;
/*...*/

MapFragment elMapaQueVamosAInyectar = new MapFragment();

FragmentManager fm = getActivity().getFragmentManager();

fm.beginTransaction().add(
    R.id.contenedor_mapa + mapaId,
    elMapaQueVamosAInyectar,
    "mapa")
.commit();
```



```
<fragment  
  android:id="@+id/map"  
  android:layout_width="match_parent"  
  android:layout_height="match_parent"  
  class="com.google.android.gms.maps.MapFragment" />  
  
<FrameLayout android:id="@+id/contenedor_mapa"  
  android:layout_width="match_parent"  
  android:layout_height="match_parent"></FrameLayout>
```

# Ejercicio mapas

Agregar un mapa a un fragmento

Agregar varios Markers

Controlar eventos de Click y LongClick

Cambiar de modo el mapa

Diferencia entre Location de mapa y de Geolocalización

# Ejercicio GeoCoder

Controlar un evento OnClick

Arrastrar un marcador buscando una dirección

Capturar la dirección con la cantidad de líneas

Actualizar las direcciones

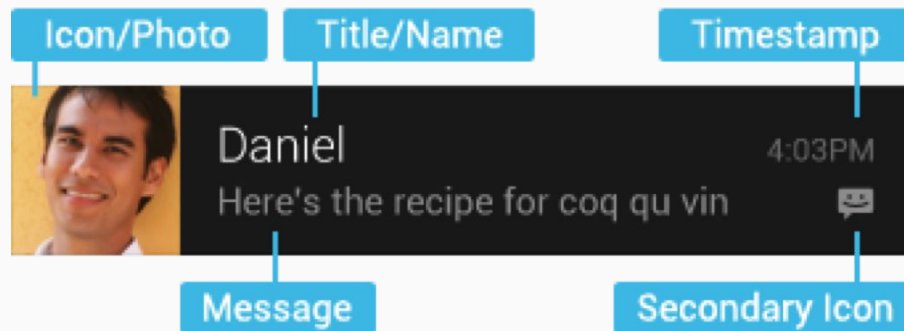
# Notificaciones

Componentes obligatorios:

- `smallIcon`
- `title`
- `detailedText`

Clase `NotificationCompat` y `Builder`

El `Builder` genera las notificaciones, se notifica por servicio del sistema



# Notificaciones

Tienen por defecto un PendingIntent (que contiene un Intent que conduce generalmente a un Activity)

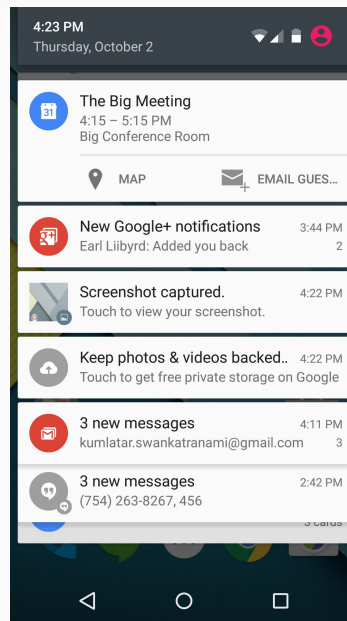
Pueden tener hasta 3 Action, con distintos Intent

Tienen prioridades que varían  $-2 \leq x \leq 2$

Pueden tener varios BigStyle

Son actualizables, y deben actualizarse siempre en la medida de lo posible

Son “cancelables”, y pueden mostrar progreso `setProgress()`



# Ejercicio Notificaciones

Notificar dado un formulario

Agregar datos obligatorios

Adicionar Ticker, ContentInfo, Patrón de Vibración, Acciones, Intent, Segunda Actividad

Agregar acciones

Modificar vista tipo lista

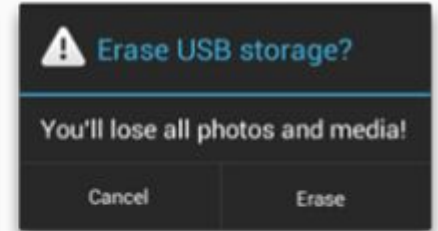
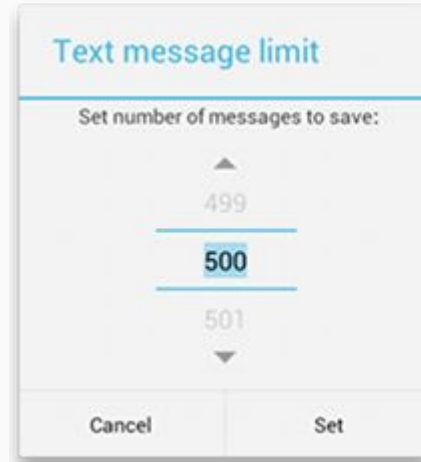
# Diálogos: Fragmentos

Se basan en fragmentos

Tienen 3 componentes:

- Título
- Mensaje / Contenido
- Botones / Acciones (Máximo 3)

Se comunican a través de interfaces



# Ejercicio Diálogo 1

Crear un diálogo con contenidos personalizados

Recibir a cuál botón se le dió click



# Ejercicio Diálogo 2

Volver a la lista de items

Agregar la opción de editar ítem al menú contextual

Poder cambiar el nombre al ítem, y actualizarlo en la lista

# Mapas y Geolocalización

## Mapas:

- Fragmentos
- Conexión con cuenta de desarrollador
- Firma SHA1



## Geolocalización:

- Independiente de mapas
- Permisos en manifiesto
  - Google Play Services
- Diferencia entre clases Location y LatLng

# Geolocalización

Ubicación desde un LocationManager

Múltiples fuentes de información de las ubicaciones

Actualizable con el movimiento del usuario

Precisión variable

# Geolocalización: Usando un LocationManager

```
// Acquire a reference to the system Location Manager
LocationManager locationManager = (LocationManager) this.getSystemService(Context.
LOCATION_SERVICE);

// Define a listener that responds to location updates
LocationListener locationListener = new LocationListener() {
    public void onLocationChanged(Location location) {
        // Called when a new location is found by the network location provider.
        makeUseOfNewLocation(location);
    }
    public void onStatusChanged(String provider, int status, Bundle extras) {}
    public void onProviderEnabled(String provider) {}
    public void onProviderDisabled(String provider) {}
};

// Register the listener with the Location Manager to receive location updates
locationManager.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, 0, 0,
locationListener);
```

# Ejercicio Geolocalización

Obtener información de ubicación desde las dos (tres) fuentes


Corroborar con un mapa y un marcador que se actualice con la posición

Comparar la precisión del método de obtener información desde el mapa, y desde los proveedores del LocationManager

# APIs: Conexión, lectura y consumo

Las API (Application Programming Interface) son vitales en todos los sistemas modernos web

Exponen información al recibir (o no) parámetros en una dirección web, en un formato estándar (JSON/XML)



The screenshot shows a web browser window with the address bar displaying the URL: `api.openweathermap.org/data/2.5/weather?q=bogota&mode=json&appid=261db4f32`. The main content area displays a JSON object representing weather data for Bogotá. The JSON is formatted with syntax highlighting, and a light blue selection box highlights the 'main' object. The 'main' object contains the following data: temperature (287.167), pressure (840.32), humidity (95), temperature minimum (287.167), temperature maximum (287.167), sea level (1024.97), and ground level (840.32).

```
{
  - coord: {
    lon: -74.08,
    lat: 4.61
  },
  - weather: [
    - {
      id: 802,
      main: "Clouds",
      description: "scattered clouds",
      icon: "03n"
    }
  ],
  base: "cmc stations",
  - main: {
    temp: 287.167,
    pressure: 840.32,
    humidity: 95,
    temp_min: 287.167,
    temp_max: 287.167,
    sea_level: 1024.97,
    grnd_level: 840.32
  },
  - wind: {
    speed: 1.15,
    deg: 60.0031
  },
  - clouds: {
    all: 32
  },
  dt: 1446091714,
  - sys: {
    message: 0.0028,
    country: "CO",
    sunrise: 1446115260,
    sunset: 1446158336
  },
  id: 3688689,
  name: "Bogota",
  cod: 200
}
```

# Consumo APIs

Se requiere de una conexión a la URL, de tipo `HttpURLConnection`

La conexión devuelve un `Stream`, el cual lo lee un objeto tipo `Scanner`, y lo convierte a `String`

El `String` se convierte a objetos `JSON` o `XML` según el caso, y se extraen los datos necesarios

Toda la gestión se hace a través de un `AsyncTask`

# AsyncTask

Es una tarea que se ejecuta asíncronamente, con los siguientes métodos principales:

- `onPreExecute`
- `doInBackground`
- `onPostExecute`
- `onProgressUpdate`

Durante la ejecución de la tarea, se puede informar del progreso al hilo principal

Recibe un tipo de dato de entrada, uno de progreso, y uno de entrega o resultado

```
new AsyncTask<String, Integer, User>();
```



# Ejercicio APIs

Consultar el API de OpenWeatherMap

Parsear un objeto JSON

Obtener los datos de Clima:

- Nombre de la ciudad (reconocida)
- Clima
- Temperatura
- Presión
- Humedad

# Parser XML

Clase XmlPullParser (desde Factory)

Eventos next(), nextToken()

Escaneo de eventos a medida que se lee el input

```
<bookstore>
  <book category="children">
    <title>Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title>Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

# Ejercicio API: XML

Usar la misma estructura del proyecto:

- Un Parser común
- La misma interfaz
- Una opción de consulta JSON/XML desde vista

# Navigation Drawer

Componente raíz: `DrawerLayout`

FrameLayout contenedor de las transacciones de fragmentos

ListView que define los comportamientos del frame principal de contenido

Se realizan transacciones para cambiar entre fragmentos

# Ejercicio Navigation Drawer

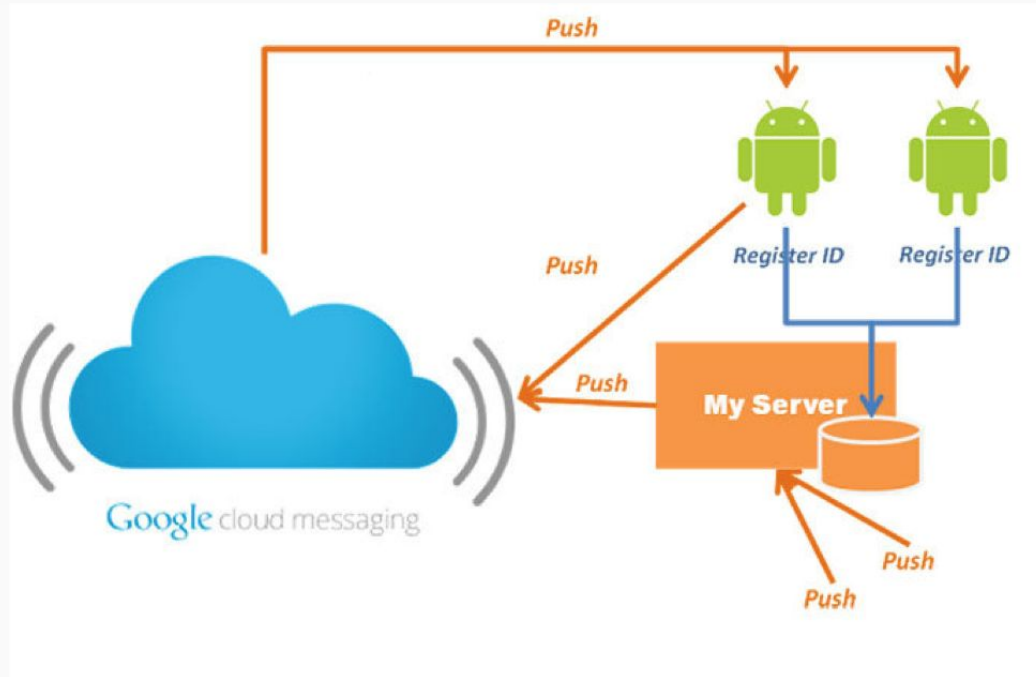
Crear un NavigationDrawer que permita cambiar entre fragmentos varios

Cambiar los títulos de la actividad

Crear un NavigationDrawer que permita navegar entre insertar y ver usuarios

Agregar Header y Footer personalizado al NavigationDrawer

# GCM: Google Cloud Messaging



# GCM: Server

Dos tipos de servidores:

- HTTP: Bajada
- XMPP: Bajada/Subida

Implementaciones nuevas GCM:

<https://developers.google.com/cloud-messaging/server>

# Paso a paso GCM

- SENDER\_ID (API Console)
- Verificar SharedPreferences
- registerInBackground
  - Pedir un regId (registration ID) del server GCM
- Alojar el ID en la DB propia
- Alojar el ID en SharedPreferences
- BroadcastReceiver escuchando push
- IntentService que notifica



# Ejercicio GCM

Formulario para enviar la información de registro

BroadcasterReceiver | IntentService para controlar la notificación

URL: [http://sikuani.net/gcm\\_example/register.php](http://sikuani.net/gcm_example/register.php)

Enviar coordenadas formato GMAPS, y recibir la notificación con un Marker y la dirección