Máster Universitario en Ingeniería Informática

Gestión de Información en Dispositivos Móviles

Introducción a la plataforma Android

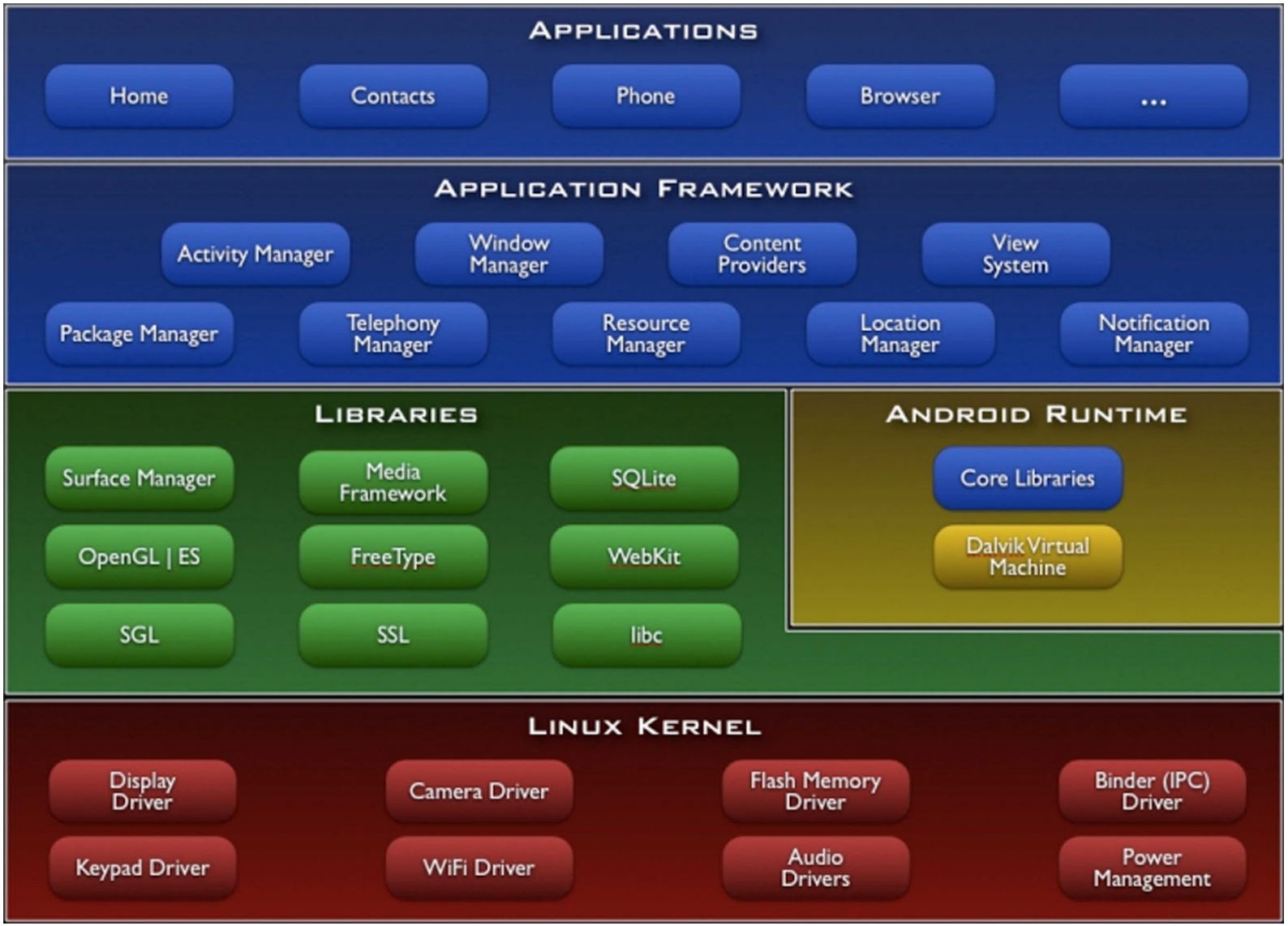
Javier Abad (abad@decsai.ugr.es)
Dpto. de Ciencias de la Computación e Inteligencia Artificial
http://decsai.ugr.es

La plataforma Android

- Paquete de software diseñado principalmente, pero no exclusivamente, para dar soporte a dispositivos móviles (móviles y tabletas).
- Estructurado en capas: núcleo del SO, bibliotecas, marco de aplicaciones y aplicaciones básicas.
- Software Development Kit (SDK): herramientas de desarrollo de aplicaciones Android.
- Documentación: tutoriales, blogs, ejemplos, etc.

Android Developers: https://developer.android.com

El núcleo Linux



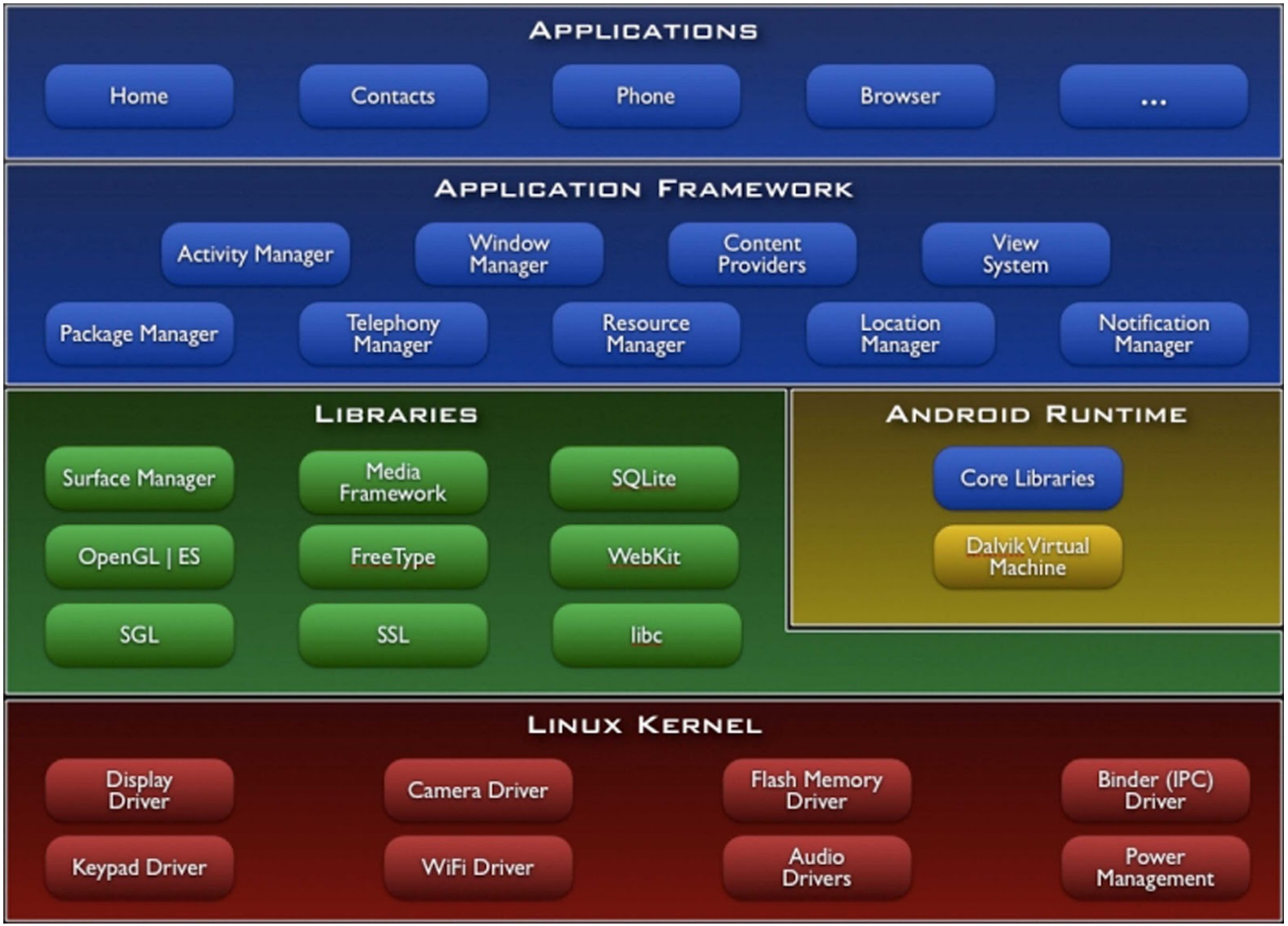
El núcleo Linux

- Proporciona los servicios básicos de los que depende todo dispositivo Android.
- Servicios estándar:
  - Seguridad
  - Gestión de memoria
  - Gestión de procesos
  - E/S a ficheros y a la red
  - Controladores de dispositivos

El núcleo Linux

- Servicios específicos de Android:
  - Gestión de energía
  - Gestión de memoria
    - Compartición de memoria
    - Eliminador de procesos
  - Mecanismo de comunicación entre procesos
  - Otros...

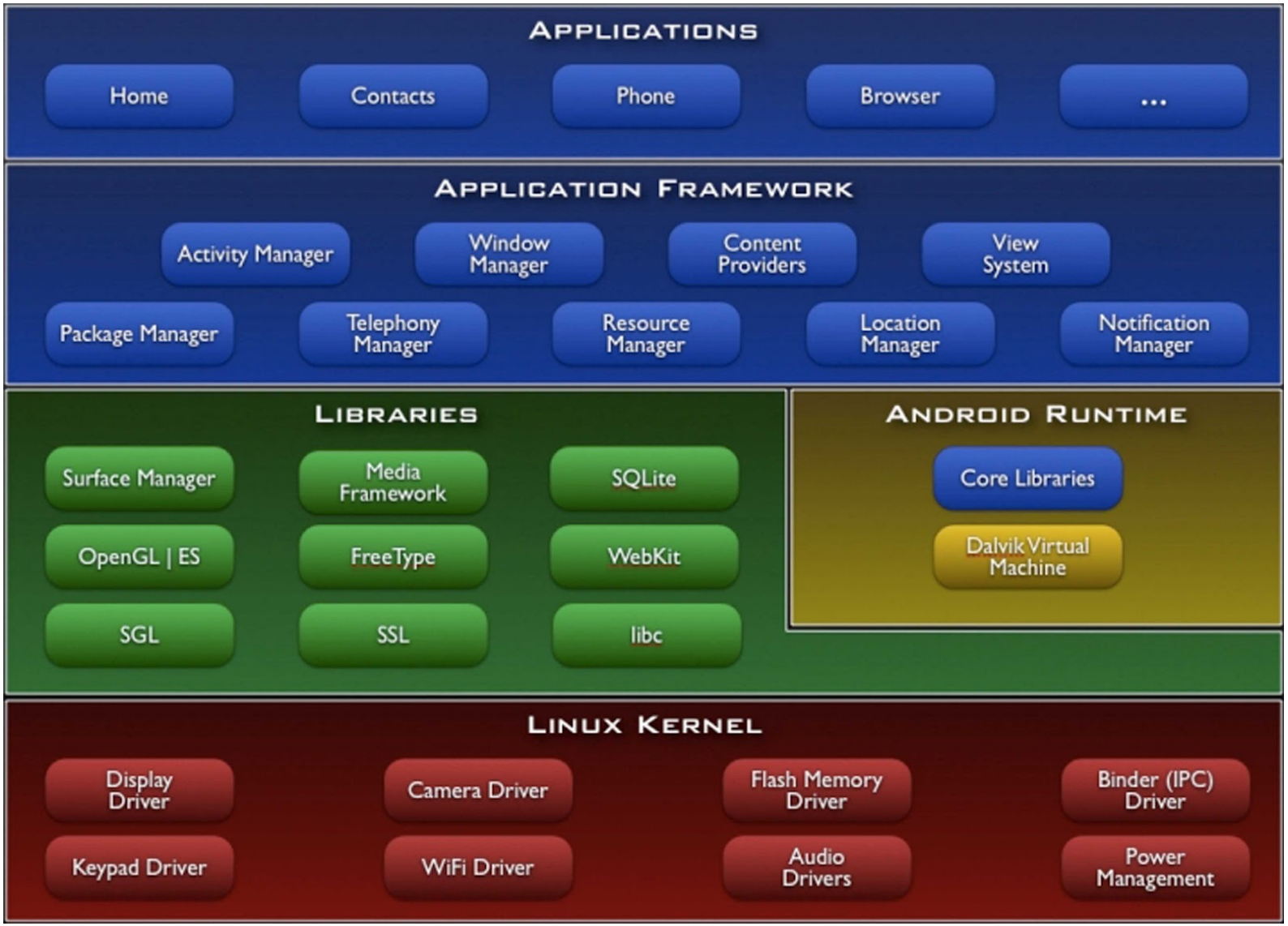
Bibliotecas



Bibliotecas

- Biblioteca C del sistema: Bionic libc
- Surface Manager: actualización de la pantalla
- Media Framework: reproducción de audio/vídeo
- Webkit: motor del browser
- OpenGL: motor de gráficos
- SQLite: gestión de bases de datos relacionales

Android Runtime (Sistema de ejecución)



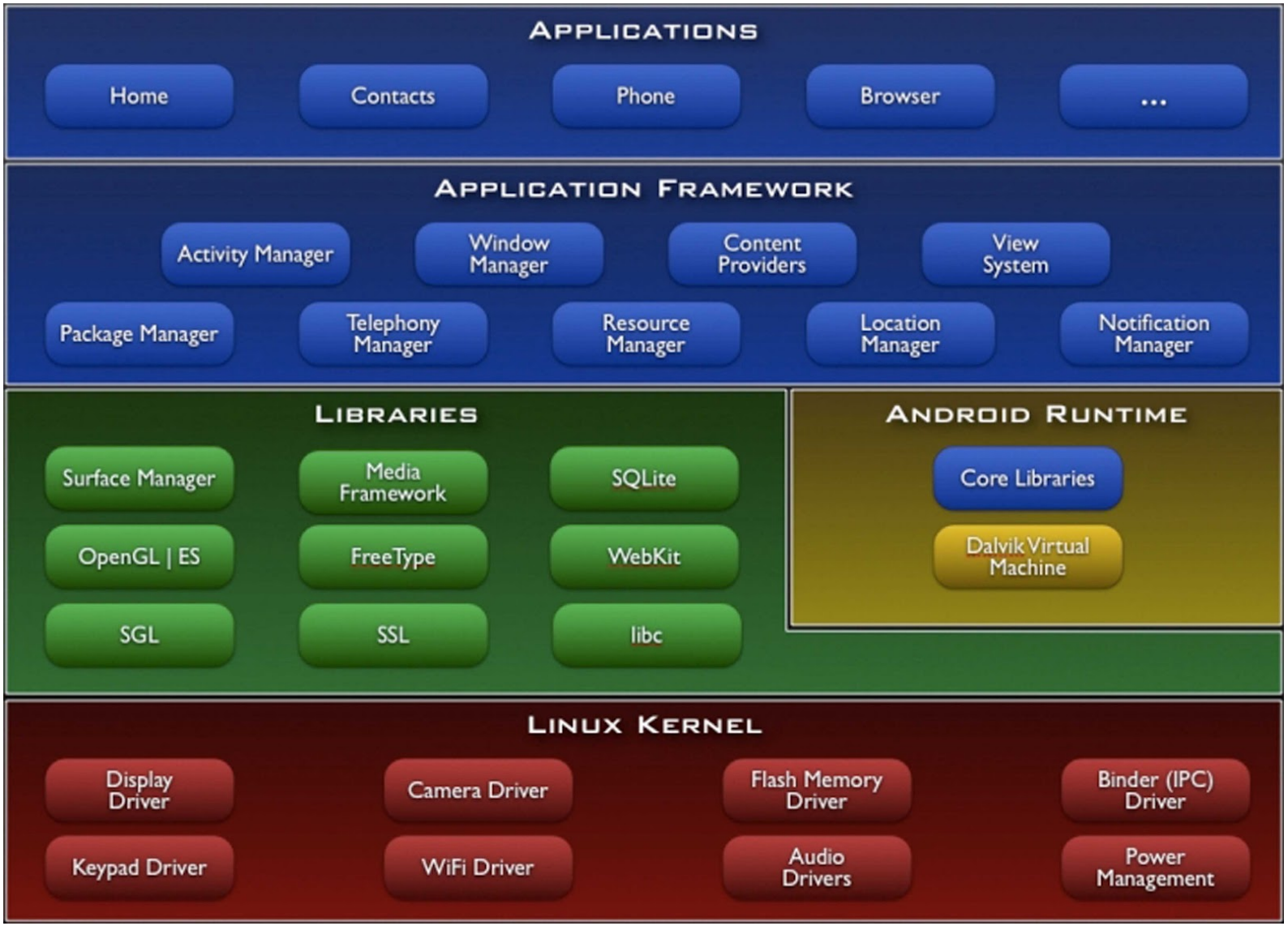
Android Runtime (Sistema de ejecución)

- Bibliotecas básicas Java
  - Clases básicas Java – java.*, javax.*
  - Ciclo de vida – android.*
  - Servicios de Internet/web – org.*
  - Pruebas unitarias – junit.*
- Máquina Virtual Dalvik (*Dalvik Virtual Machine – DVM*)
  - Ejecuta las aplicaciones Android

Android Runtime (Sistema de ejecución)

- Flujo de trabajo típico
  - Aplicación escrita en Java
  - Compilación – ficheros bytecode Java
  - DX convierte los ficheros bytecode Java en un único fichero bytecode dex (classes.dex)
  - La DVM ejecuta el fichero classes.dex
- La DVM está diseñada para trabajar con recursos limitados
  - CPU menos potente
  - Menos memoria
  - Batería limitada
- Dalvik VM Internals (Dan Bornstein) [Google I/O 2008]
  https://sites.google.com/site/io/dalvik-vm-internals

El marco de aplicaciones



El marco de aplicaciones

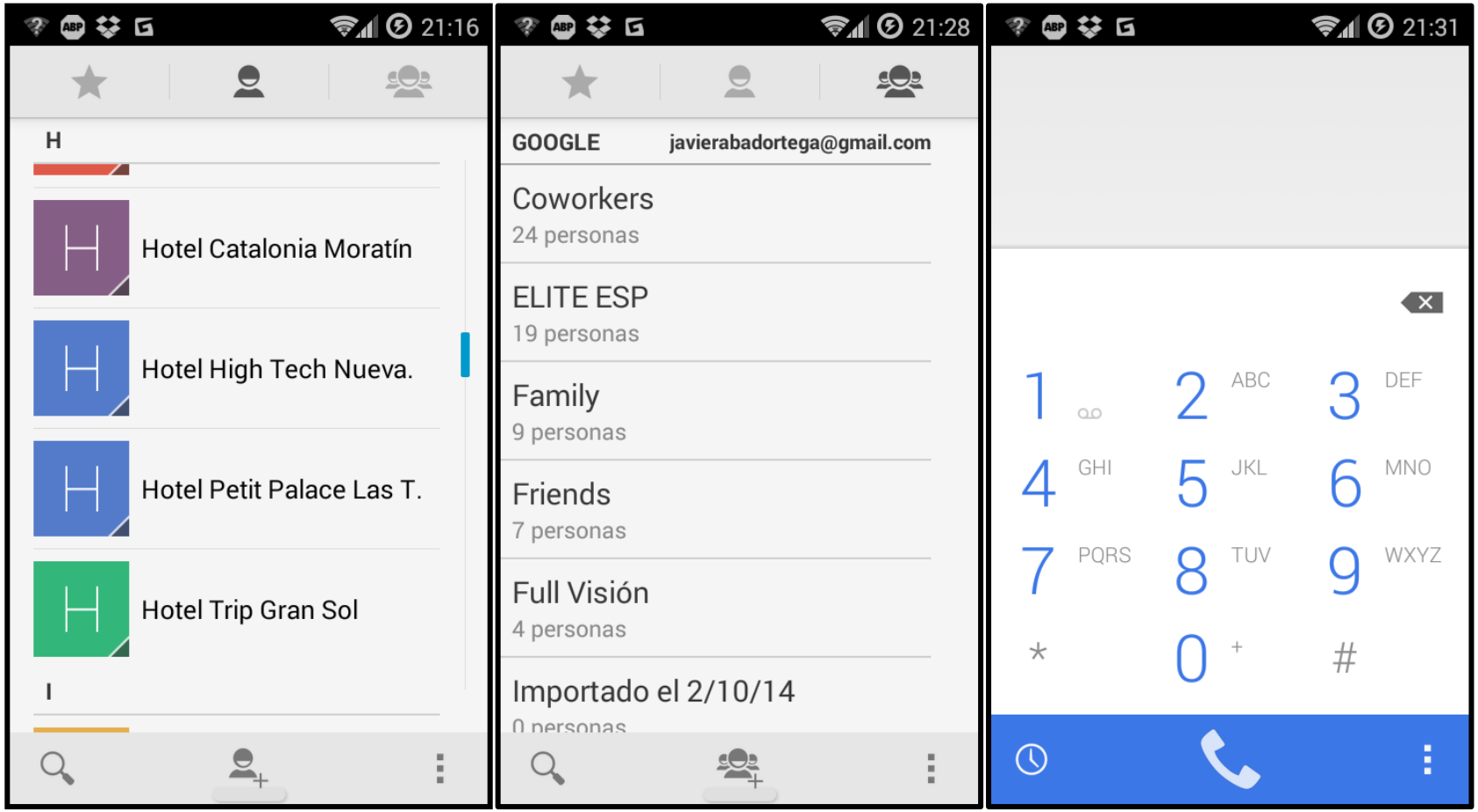
- Gestor de paquetes (*Package Manager*)
  - Mantiene registro de las aplicaciones instaladas en el dispositivo
- Gestor de ventanas (*Window Manager*)
  - Gestiona las ventanas que forman una aplicación

El marco de aplicaciones

El marco de aplicaciones

- Sistema de vistas (View System)
  - Proporciona elementos comunes de la interfaz de usuario

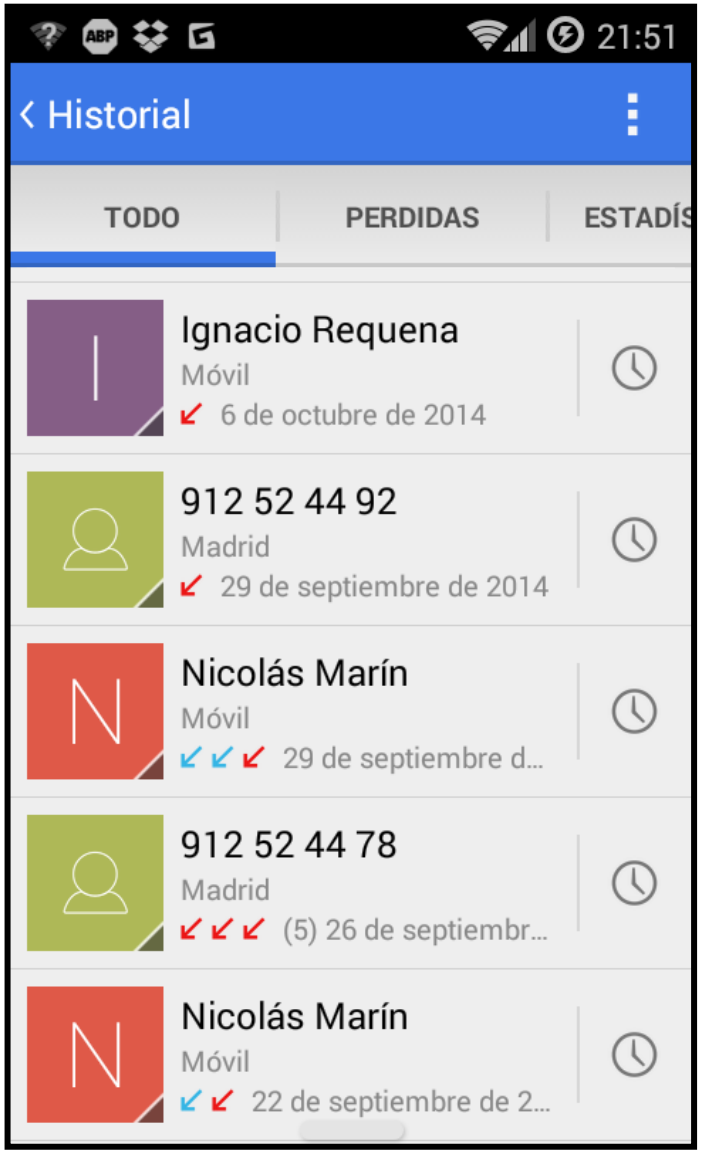
El marco de aplicaciones

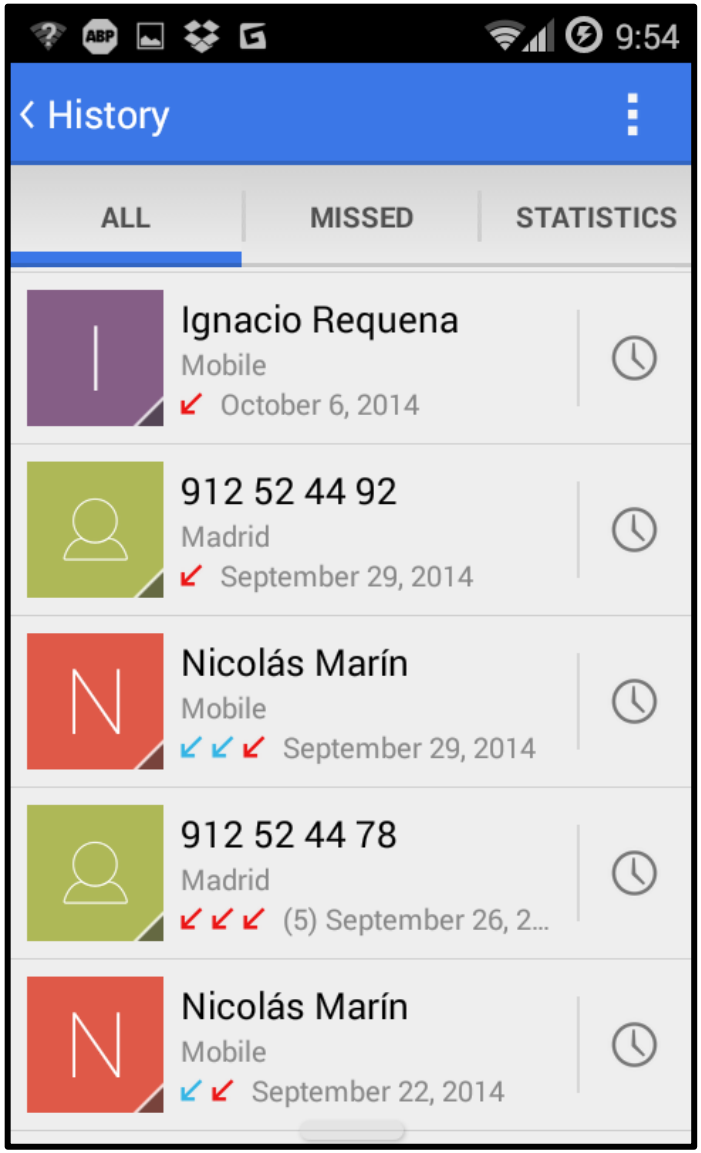


El marco de aplicaciones

- Gestor de recursos (Resource Manager)
  - Gestiona recursos no compilados de la aplicación

El marco de aplicaciones

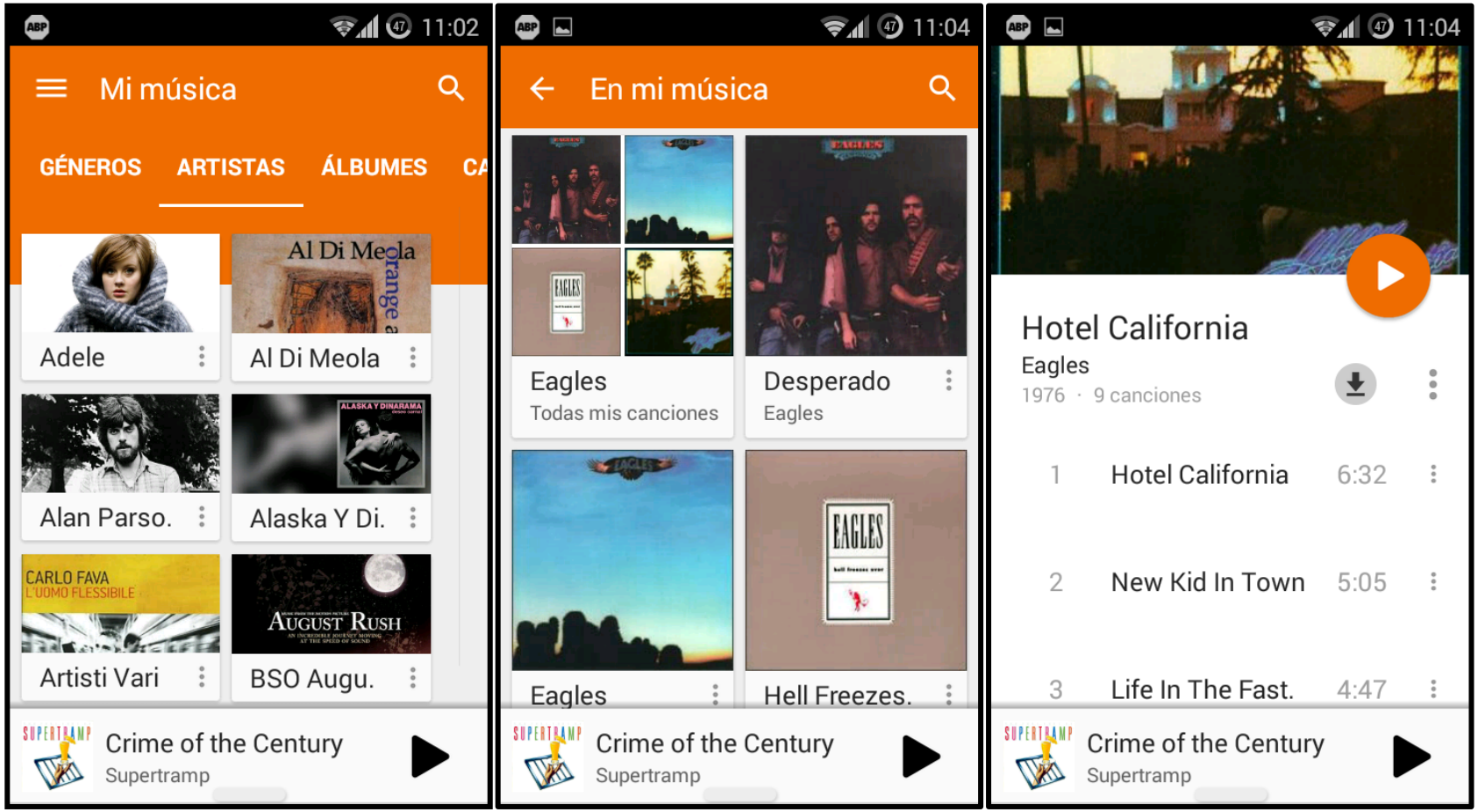




El marco de aplicaciones

- Gestor de actividades (Activity System)
  - Gestiona el ciclo de vida de las aplicaciones y la pila de navegación

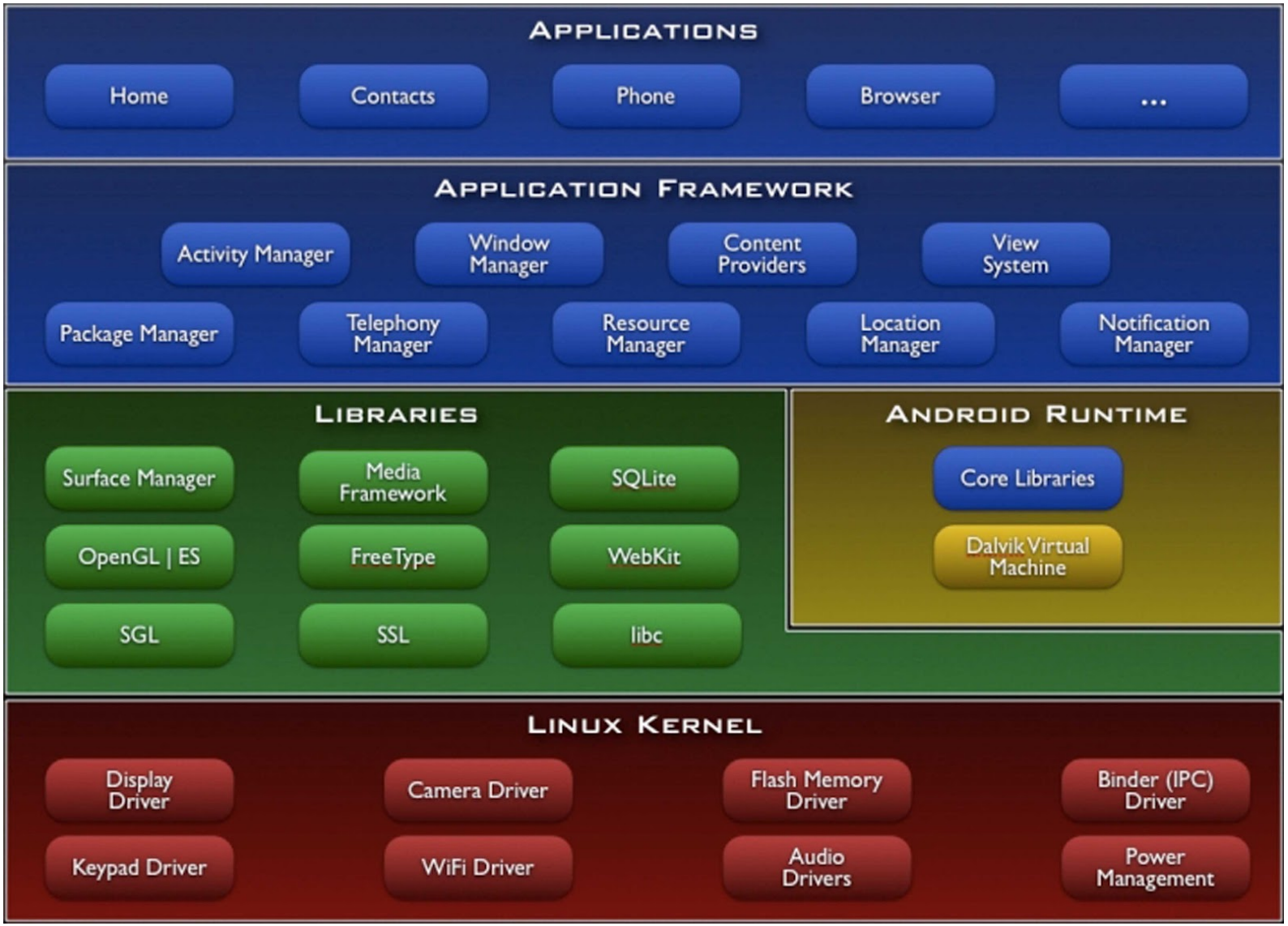
El marco de aplicaciones



El marco de aplicaciones

- Proveedores de Contenido (Content Providers)
  - Compartición de datos entre aplicaciones
- Gestor de localización (Location Manager)
  - Proporciona información de posición y movimiento
- Gestor de notificaciones (Notification Manager)
  - Sitúa iconos de notificación en la barra de status cuando se producen determinados eventos

Capa de aplicaciones



Capa de aplicaciones

- Aplicaciones estándar:
  - Home – Pantalla de Inicio
  - Contactos – Base de datos de contactos
  - Teléfono – Realiza llamadas telefónicas
  - Navegador – Visualiza páginas web
  - Lector de correo – Lee y compone e-mails
  - Etc.
- Podemos sustituir todas estas aplicaciones

Componentes de una aplicación Android

- Activity
  - Tiene Interfaz Gráfica de Usuario
- Servicios
  - Ejecución "larga" (background)
- Broadcast receivers
  - Escuchan y responden a eventos
- Content providers
  - Permiten a las aplicaciones almacenar y compartir datos

Las aplicaciones están formadas por diferentes componentes

Android inicia (instancia) y ejecuta estas componentes conforme es preciso.

Cada componente tiene su propio objetivo y sus propias APIs

Actividades (Activity Class)

- Clase principal de interacción con el usuario
- Normalmente implementará una única tarea que el usuario pueda realizar.

```
/**
 * The dialer activity that has one tab with the virtual 12key
 * dialer, a tab with recent calls in it, a tab with the contacts and
 * a tab with the favorite. This is the container and the tabs are
 * embedded using intents.
 * The dialer tab's title is 'phone', a more common name (see strings.xml).
 */
public class DialtactsActivity extends TransactionSafeActivity
        implements View.OnClickListener {
    private static final String TAG = "DialtactsActivity";

    public static final boolean DEBUG = false;
```

Servicios (Service Activity)

- Se ejecutan en background. No tienen interfaz
- Realizan operaciones de ejecución "larga"
- Permiten la interacción con procesos remotos

```
/**
 * Provides "background" audio playback capabilities, allowing the
 * user to switch between activities without stopping playback.
 */
public class MediaPlaybackService extends Service {
    /** used to specify whether enqueue() should start playing
     * the new list of files right away, next or once all the currently
     * queued files have been played
     */
    public static final int NOW = 1;
    public static final int NEXT = 2;
    public static final int LAST = 3;
    public static final int PLAYBACKSERVICE_STATUS = 1;

    public static final int SHUFFLE_NONE = 0;
    public static final int SHUFFLE_NORMAL = 1;
    public static final int SHUFFLE_AUTO = 2;
```

BroadcastReceiver

- Escucha y responde a eventos
- Juega el papel de suscriptor en el patrón publicador/suscriptor
- Los eventos se representan mediante la clase Intent y después se transmiten (broadcast)
- El Broadcast Receiver recibe y responde al evento.

```
/**
 * Handle incoming SMSes.  Just dispatches the work off to a Service.
 */
public class SmsReceiver extends BroadcastReceiver {
    static final Object mStartingServiceSync = new Object();
    static PowerManager.WakeLock mStartingService;
    private static SmsReceiver sInstance;

    public static SmsReceiver getInstance() {
        if (sInstance == null) {
            sInstance = new SmsReceiver();
        }
        return sInstance;
    }
```

Content Providers

- Permiten a las aplicaciones almacenar y compartir datos
- Usa una interfaz tipo base de datos
- Gestiona la comunicación entre procesos

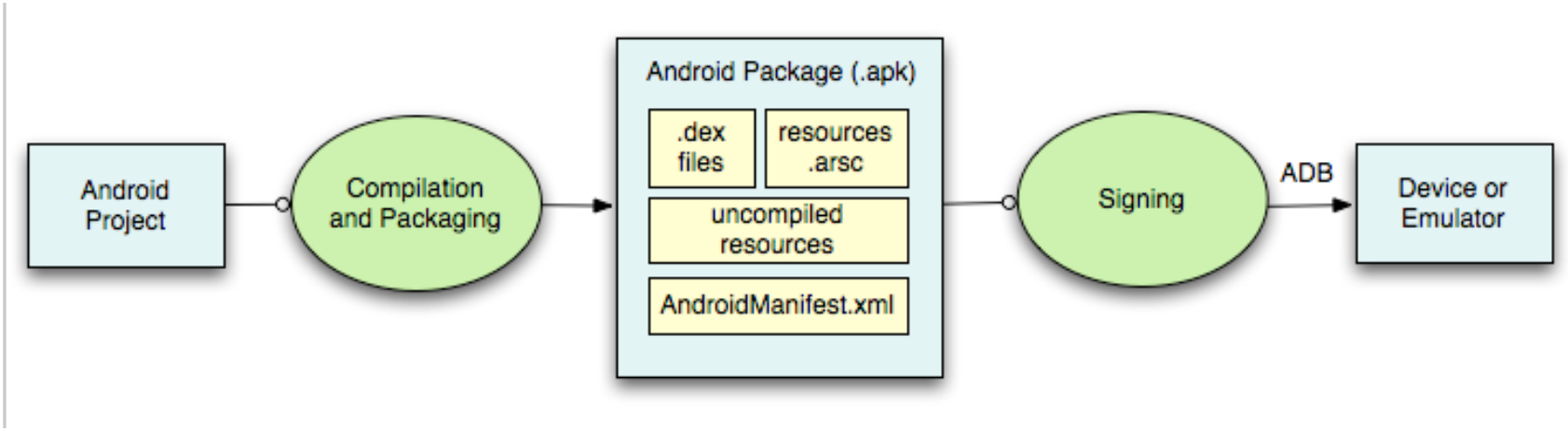
```
public class BrowserProvider extends ContentProvider {

    private SQLiteOpenHelper mOpenHelper;
    private BackupManager mBackupManager;
    static final String sDatabaseName = "browser.db";
    private static final String TAG = "BrowserProvider";
    private static final String ORDER_BY = "visits DESC, date DESC";

    private static final String PICASA_URL = "http://picasaweb.google.com/
m/" +
            "viewer?source=androidclient";

   static final String[] TABLE_NAMES = new String[] {
      "bookmarks", "searches"
   };
```

Construcción de aplicaciones



1. Definición de recursos
2. Implementación de las clases de la aplicación
3. Empaquetado de la aplicación
4. Instalación y ejecución de la aplicación

Construcción de aplicaciones

1. Definición de recursos
   - Entidades no compilables (no son código): ficheros layout, imágenes, strings, menús...
   - Separados del código
   - Permite adaptar la aplicación a diferentes dispositivos y usuarios
   - http://developer.android.com/guide/topics/resources
   - Strings:
     - string, string array y plurals
     - Se almacenan en res/values/*.xml
     - Especificados en xml

     <string name="hello">Hello World!</string>
     - Puede incluir formato
     - Acceso como @string/string_name
     - Acceso en Java como R.string.string_name

Recursos – Strings

- res/values/strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="show_map_string">Show Map</string>
    <string name="location_string">Enter Location</string>
</resources>
```

- res/values-es/strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="show_map_string">Mostrar mapa</string>
    <string name="location_string">Escriba la dirección</string>
</resources>
```

Recursos – Fichero de Layout

- El diseño (Layout) de las interfaces de usuario se especifican en ficheros XML
- Tenemos herramientas para crear los diseños visualmente. La herramienta generará el fichero XML.
- Los ficheros XML se almacenan normalmente en res/layout/ *.xml
- Acceso en Java como R.layout.layout_name
- Acceso en otros ficheros de recursos como @layout/ layout_name
- Podemos usar múltiples ficheros de layout. Android elige en tiempo de ejecución el layout en función de las características del dispositivo.

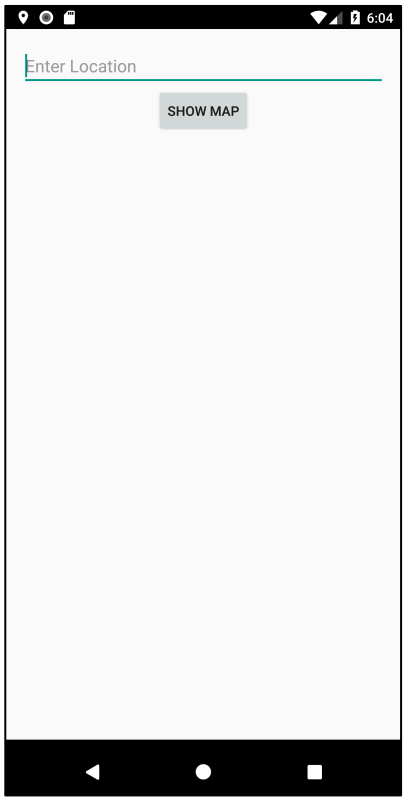
Recursos – Fichero de layout

- res/layout/activity_main.xml

```
<RelativeLayout 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:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
   android:paddingBottom="@dimen/activity_vertical_margin" tools:context=".MainActivity">

   <EditText
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:inputType="textPostalAddress"
      android:ems="10"
      android:id="@+id/location"
      android:layout_alignParentTop="true"
      android:layout_alignParentLeft="true"
      android:layout_alignParentStart="true"
      android:hint="@string/location_string" />

   <Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@string/show_map_string"
      android:id="@+id/map_button"
      android:layout_below="@+id/location"
      android:layout_centerHorizontal="true" />
</RelativeLayout>
```

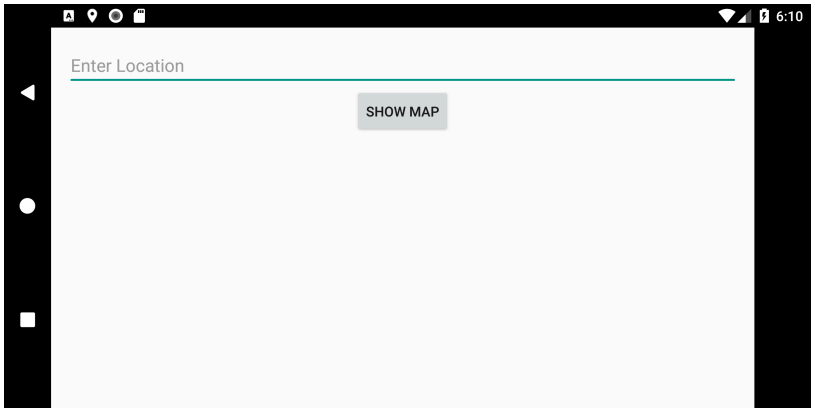


Recursos – Fichero de layout

- res/layout-land/activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/RelativeLayout1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
   <EditText
        android:id="@+id/location"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_toLeftOf="@+id/mapButton"
        android:ems="10"
        android:hint="@string/location_string"
        android:inputType="textPostalAddress" >
    </EditText>
    <Button
        android:id="@+id/mapButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_alignTop="@+id/location"
        android:text="@string/show_map_string"
        android:textSize="20sp" />

</RelativeLayout>
```



R.java

- En la compilación, los recursos se usan para generar la clase R.java
- El código Java usa la clase R para acceder a los recursos

```
/* AUTO-GENERATED FILE.  DO NOT MODIFY.
 *
 * This class was automatically generated by the
 * aapt tool from the resource data it found.  It
 * should not be modified by hand.
 */
package course.examples.MapLocation;
public final class R {
    public static final class attr {
    }
    public static final class drawable {
        public static final int ic_launcher=0x7f020000;
    }
    public static final class id {
        public static final int RelativeLayout1=0x7f050000;
        public static final int location=0x7f050001;
        public static final int mapButton=0x7f050002;
    }
    public static final class layout {
        public static final int main=0x7f030000;
    }
    public static final class string {
        public static final int location_string=0x7f040001;
        public static final int show_map_string=0x7f040000;
    }
}
```

Construcción de aplicaciones

2. Implementación de las clases de la aplicación
   - Implica crear al menos una activity
   - Método onCreate: inicialización de la actividad
     1. Restaurar estado almacenado
     2. Fijar la Content View (qué mostrar como interfaz de la actividad)
     3. Inicializar elementos de la GUI
     4. Vincular código a los elementos de la GUI

Construcción de aplicaciones

```
public class MapLocation extends Activity {
  private final String TAG = "MapLocation";
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    // Restore any saved state
    super.onCreate(savedInstanceState);
    // Set content view
    setContentView(R.layout.main);
    // Initialize UI elements
    final EditText addrText = (EditText) findViewById(R.id.location);
    final Button button = (Button) findViewById(R.id.mapButton);
...
```

Construcción de aplicaciones

```
    // Link UI elements to actions in code
    button.setOnClickListener(new Button.OnClickListener() {
      @Override
      public void onClick(View v) {
        try {
          String address = addrText.getText().toString();
          address = address.replace(' ', '+');
          Intent geoIntent = new Intent(
              android.content.Intent.ACTION_VIEW, Uri
                  .parse("geo:0,0?q=" + address));
          startActivity(geoIntent);
        } catch (Exception e) {
          Log.e(TAG, e.toString());
        }
      }
    });
  }
...
```

Construcción de aplicaciones

3. Empaquetado de la aplicación
   - El sistema empaqueta las componentes y los recursos de la aplicación en un fichero APK
   - El desarrollador proporciona la información necesaria en el fichero AndroidManifest.xml
     - Nombre de la aplicación
     - Lista de componentes
     - Otras informaciones: permisos, requerimientos hardware, nivel de API mínimo

Construcción de aplicaciones

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="course.examples.MapLocation"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="10"
        android:targetSdkVersion="17" >
    </uses-sdk>

    <application
        android:allowBackup="false"
        android:icon="@drawable/ic_launcher"
        android:label="MapLocation" >
        <activity
            android:name="course.examples.MapLocation.MapLocation">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Construcción de aplicaciones

4. Instalación y ejecución de la aplicación
   - Desde Android Studio, ejecutar en el emulador o dispositivo
   - Desde la línea de órdenes:
     - Habilitar depuración USB
     - adb install <path de la aplicacion>