Ing. Jorge Nolasco Valenzuela

DESARROLLO DE APLICACIONES MÓVILES CON ANDROID

- Mis primeros pasos con Android
- Estructura de una aplicación Android
- Interfaz de usuario
- Estilos y temas
- Seguridad de las aplicaciones móviles



EDITORIAL MACRO®

DESARROLLO DE APLICACIONES MÓVILES CON

ANDROID

EDITORIAL MACRO®

España - México - Colombia - Chile - Ecuador - Perú - Bolivia - Uruguay - Guatemala - Costa Rica

Desarrollo de aplicaciones móviles con Android
Autor: Jorge Santiago Nolasco Valenzuela

© Derechos de autor registrados:
Empresa Editora Macro EIRL

© Derechos de edición, arte gráfico y diagramación reservados:
Empresa Editora Macro EIRL

Jefe de edición:
Cynthia Arestegui Baca

Diseño de portada:
Alejandro Marcas León

Edición a cargo de:
© Empresa Editora Macro EIRL
Av. Paseo de la República N.° 5613, Miraflores, Lima, Perú

Teléfono: (511) 748 0560
E-mail: proyectoeditorial@editorialmacro.com
Página web: www.editorialmacro.com

Primera edición: mayo de 2013
Segunda edición: noviembre de 2014
Primera reimpresión: junio de 2015
Tiraje: 1000 ejemplares

Impresión
Talleres gráficos de la Empresa Editora Macro EIRL
Jr. San Agustín N.° 612-624, Surquillo, Lima, Perú

ISBN N.° 978-612-304-244-8
Hecho el depósito legal en la Biblioteca Nacional del Perú N.° 2015-07331

Prohibida la reproducción parcial o total, por cualquier medio o método, de este libro sin previa autorización de la Empresa Editora Macro EIRL.

Ing. Jorge Santiago Nolasco Valenzuela

Ingeniero de sistemas y egresado de la Escuela de Posgrado de la Universidad Nacional Federico Villarreal. Catedrático en la Universidad Inca Garcilaso de la Vega y en la Universidad Nacional Federico Villarreal.

Entre sus principales pasiones se encuentra la informática y las nuevas tecnologías. Además, colabora frecuentemente en proyectos empresariales, organizaciones relacionadas al mercado de valores, líneas aéreas y constructoras.

Dedicatoria

A Dios, por darme una vida llena de dichas y
guiar mi camino todos los días de mi vida

A mis padres, Félix y Eva, y a mis hermanos,
quienes siempre estuvieron a mi
lado en todo momento de mi vida

Introducción

El presente libro titulado *Desarrollo de aplicaciones móviles con Android* tiene como objetivo constituir un texto básico para aquellos que quieran incursionar en el mundo de Android y desean mantenerse a la vanguardia tecnológica.

Está estructurado en 19 capítulos; en los cuales se desarrolla temas relacionados a la plataforma móvil de Android desde los conceptos generales hasta la estructura misma de la aplicación, los fragmentos (usos y ejemplos), elementos y características de la interfaz de usuario, estilos, temas, sistemas de gestión de base de datos (SQLite, MySQL), la plataforma software JADE, entre otros.

Emplea una metodología práctica y visual que facilita el aprendizaje del lector de una manera sencilla y directa; asimismo presenta ejemplos de aplicación que ayudan a reforzar lo aprendido.

Como se mencionó anteriormente, Android es una plataforma móvil, que incluye un sistema operativo móvil basado en Linux apoyado por Google. Actualmente es el gran competidor de iPhone; además, la API de Android se estabilizó en estos últimos tiempos. Recientemente, Google ha publicado su última versión del SDK llamado Android 4.1 (en el año 2012). Cabe resaltar, que este texto fue escrito en el SDK 4.1 para Android.

Por su parte el diseño de Android se realizó con el objetivo de que sea compatible con una amplia gama de *hardware*. Todo esto gracias al núcleo kernel Linux, que progresivamente ha evolucionado para apoyar una gran variedad de *hardware*. Esta característica de la plataforma es importante, ya que permite a los fabricantes la libertad de diseñar y adquirir componentes ideales. Para el desarrollo de las primeras aplicaciones, se ha utilizado las mejores herramientas tal como el IDE Eclipse.

Índice

Capítulo 1: Tecnologías móviles

1.1. Introducción a las tecnologías móviles 15
1.2. Lenguaje Java 15

Capítulo 2: Mis primeros pasos con Android

2.1. Definición de Android 19
2.2. Arquitectura 20
2.2.1. Núcleo de Android 21
2.2.2. Librerías 21
2.2.3. Entorno de ejecución 21
2.2.4. Framework 22
2.2.5. Aplicaciones 22
2.3. Dalvik 23
2.4. Requerimientos mínimos de hardware 23
2.5. Versiones de Android 24
2.5.1. Android 4.1 26
2.5.2. Android 4.2 30
2.5.3. Cuota de las versiones 31
2.6. Fragmentación en Android 31
2.7. Mostrar la versión actual de un dispositivo móvil 33
2.7.1. Códigos 33
2.7.2. Ejemplos 34
2.8. Dispositivos Android en el mercado 35
2.8.1. Smartphones 35
2.8.2. Tablets 36
2.8.3. E-reader 37
2.8.4. Netbooks 37
2.8.5. Reproductores MP4 37
2.8.6. Internet TV 37
2.8.7. Android 4.0, Mini Android TV, Zero Devices Z802, Wi-Fi y HDMI 38
2.8.8. Quake 3 para Android 38
2.8.9. ARCHOS GamePad 38
2.8.10. Radio 38
2.8.11. Automóviles 38
2.8.12. Kit Car 39
2.8.13. Otros dispositivos 39
2.9. Noticias sobre Android 40
2.10. Aplicaciones 42
2.10.1. Google Play 42
2.11. Paquetes Android 45
2.12. Requerimiento de software 52
2.12.1. Java JDK 52
2.12.2. ADT Bundle 52
2.13. Instalación de los paquetes 54
2.14. Android Virtual Devices (AVD) 55
2.14.1. Crear Android Virtual Devices 56
2.14.2. El emulador 58
2.14.3. Herramientas de desarrollo de Android (ADT) 62

2.14.4. Configurar Eclipse para acceder al SDK 65
2.15. Herramienta alternativa 66
2.15.1. NDK (Native Development) 66
2.16. Rootear un dispositivo móvil 67
2.16.1. CyanogenMod 67

Capítulo 3: Mi primera aplicación

3.1. Crear la primera aplicación 71
3.2. Explorar la aplicación 75
3.2.1. src/ 75
3.2.2. gen/ 75
3.2.3. res/ 75
3.3. Modificar texto 78
3.4. Atributos de diseño 80
3.5. Fichero 81
3.5.1. Fichero activity_main.xml 81
3.5.2. Fichero AndroidManifest.xml (metadatos de la aplicación) 82
3.6. Llevar el aplicativo al dispositivo móvil para ser instalado y ejecutado 86
3.7. App gratuita o pagada 91
3.8. Recomendaciones para el desarrollo de aplicaciones Android 91

Capítulo 4: Estructura de una aplicación Android

4.1. Actividad (Activity) 95
4.1.1. Descripción del código de la actividad 96
4.1.2. Ciclo de vida de las aplicaciones Android 98
4.1.3. Orden de los eventos en Android 100
4.1.4. Etapas de una actividad (Activity) 101
4.1.5. Antes de codificar el Archivo MainActivity.Java 103
4.2. Intents 105
4.3. Vistas (Views) 105
4.4. Servicios (Services) 105
4.5. Proveedores de contenido (Content Providers) 106
4.6. Receptores de difusión (Broadcast Receivers) 106

Capítulo 5: Fragmentos

5.1. Concepto de fragmento 109
5.2. Uso de fragmentos 109
5.2.1. Recomendaciones 110
5.2.2. Ejemplo 110
5.3. Crear fragmentos 112
5.3.1. Primer fragmento 112
5.3.2. Segundo fragmento 114
5.4. Presentación de los fragmentos en forma vertical 117
5.5. Barra de acciones 118
5.5.1. Añadir elementos a la barra de herramientas 119
5.5.2. Añadir métodos 120
5.5.3. Uso de notificaciones o toast 124
5.5.4. Ejemplo Meses 128

Capítulo 6: Interfaz de usuario

6.1. Vista (View) 137
6.1.1. ViewGroup 137
6.1.2. Organización de las vistas 137
6.2. Layout 138
6.2.1. Tipos de layout 138
6.3. Tipos de botones 139
6.3.1. Button 139
6.3.2. ToggleButton (on/off) 141
6.3.3. ImageButton 142
6.4. Vistas de texto 144
6.4.1. TextView 144
6.4.2. EditText 145
6.5. EjemploBotones 147
6.6. Eventos de los botones 149
6.7. EjemploMail 151
6.8. Ejemplo_Operaciones 153
6.9. EjemploControles2 159
6.9.1. AbsoluteLayout 161
6.10. ListView 165
6.10.1. Crear un archivo XML 165
6.10.2. Diseñar una ListView simple 167
6.10.3. Diseñar una ListView múltiple 170
6.10.4. Diseñar la vista de los elementos de la lista en res/layout/ lista.xml 171
6.10.5. Aplicación Pisco 175
6.11. SlidingDrawer (cajón deslizante) 182
6.11.1. ScrollView y HorizontalScrollView 183
6.11.2. Aplicación Wallpaper 183
6.11.3. Añadir funcionalidad 188
6.11.4. Modificar el manifiesto 192
6.12. WebView 193

Capítulo 7: Uso de *Intents* and *Services*

7.1. *Intents* *199*
7.2. Acción 199
7.3. Aplicación del uso de Intents 200

Capítulo 8: Estilos y temas

8.1. Aplicar estilos y temas en Android 207
8.1.1. EjemploEstilo 207

Capítulo 9: SQLite

9.1. Aspectos generales 213
9.1.1. Versión actual 214
9.2. Instalación de SQLite 214
9.3 Ejemplo Almacenamiento de datos 216

Capítulo 10: MySQL

10.1. Aspectos generales 231
10.1.1. Requisitos de software adicional necesario 232

10.1.2. Configurar un servidor web localmente 232
10.2. PHP 239
10.2.1. Codificación de la clase 239
10.3 JavaScript Object Notation (JSON) 243
10.3.1. Conceptos básicos relacionados 249
10.3.2. Sintaxis de JSON 249
10.3.3. Codificar y decodificar JSON 250

Capítulo 11: SMS

11.1. Introducción 253
11.2 EjemploSMS 253
11.3 EjemploSMS2 256
11.4. Ejemplo SMS con mensaje de notificación 259

Capítulo 12: *Marketing*

12.1. Aspectos generales 267
12.2. Orientar la aplicación para un cliente 268
12.3. Clasificación de las aplicaciones móviles referente a su funcionalidad 268
12.4. Tecnologías móviles en la cadena de valor 270
12.5. Red publicitaria móvil AdMob 271
12.5.1. Mediación de redes publicitarias de AdMob 271
12.5.2. AdMob para desarrolladores 272
12.5.3. Desarrollar para AdMob 274
12.5.4. Estrategias para la comercialización de las aplicaciones móviles 276
12.6. Ejemplo_Marketing 277
12.7. Facebook para Android 285
12.7.1. Requisitos 285
12.7.2. Instalación del SDK de Facebook para Android y el APK Facebook 285
12.7.3. Importar el SDK dentro de Eclipse 286
12.7.4. Ejecutando los ejemplos 287

Capítulo 13: Seguridad de las aplicaciones móviles

13.1. Introducción 291
13.2. Seguridad 291
13.3. Características del dispositivo móvil 291
13.4. ROM 292
13.4.1. Proceso de inicio en Android 292
13.4.2. Gestor de arranque 293
13.4.3. El Linux kernel 293
13.4.4. Proceso init 294
13.4.5. Zygote y Dalvik 296
13.4.6. El servidor del sistema 297
13.4.7. Arranque completado 297
13.5. Uso del Linux dentro del dispositivo móvil Android con terminal Emulador 298
13.5.1. Comandos básicos 298
13.6. Ingeniería inversa básica 302
13.6.1. Herramientas a utilizar 303
13.6.2. Formatos de archivo .DEX y .JAR 303
13.6.3. Contenido del archivo pisco.PAK 304
13.7. Santoku 305

13.7.1. Requerimientos para la instalación 306
13.7.2. Configurar su máquina virtual 306
13.7.3. Instalar Santoku 310
13.7.4. Comando wget 317
13.7.5. Comando VBoxManage list 319
13.8. Sqliteman 320
13.9. Herramientas de desarrollo 321
13.9.1. Android SDK Manager 321
13.9.2. AFLogical OSE 322
13.10. Características de un dispositivo móvil 323
13.11. Estrategia de Google 324
13.11.1. Aplicaciones 325
13.11.2. Seguridad en Android: Reactivo-proactivo 325
13.11.3. Técnicas de seguridad 326
13.12. Compelson MOBILedit! 327
13.12.1. Uso del MOBILedit! 327
13.12.2. Pantalla principal 329
13.13. File Expert 331

Capítulo 14: Juegos

14.1. Aspectos generales 335
14.2. Aplicación de canvas: Líneas, rectángulo y círculos 336
14.2.1. Dibujar una línea vertical 339
14.2.2. Dibujar rectángulos 340
14.2.3. Dibujar un círculo 341
14.3. Aplicación de canvas: Texto 342
14.4. Aplicación de fuentes externas: Texto 343
14.5. Datos 345
14.5.1. Persistencia y preferencia 345
14.5.2. Aplicacion sobre el uso de Preferencias 345
14.6. DisplayMetrics 350

Capítulo 15: Introducción a la inteligencia artificial: Agentes Android y dispositivos móviles

15.1. Introducción 373
15.2. Descripción de la aplicación 374
15.3. JADE programación para Android 374
15.4. Conceptos básicos 380
15.4.1. Web semántica 380
15.4.2. Implementar la web semántica 383
15.4.3. Componentes de programación 386
15.4.4. Herramientas 387
15.4.5. Impactos de la web semántica en la programación 388
15.4.6. Razonador 388
15.4.7. Lógica descriptiva 390
15.4.8. Lenguajes de la web semántica 391
15.4.9. Pellet 393
15.5. Jena 394
15.5.1. Características 394
15.5.2. Procesamiento de la API RDF y OWL 395
15.5.3. Mecanismo de inferencia en Jena 396
15.5.4. Consultas SPARQL (estándar de la W3C) en Jena 396

15.5.5. Procesamiento de importaciones ontológicas 408
15.5.6. Insertar un razonador al modelo 410
15.5.7. Resumen de razonadores 412
15.5.8. Consultar un modelo 412
15.5.9. Jena persistencia 413
15.5.10. Jena SDB 413
15.5.11. Instalación en Windows 413
15.5.12. Construcción del modelo persistente en la base datos Mysql y carga de los datos por la consola Cygwin 416
15.5.13. Aplicación Java utilizando la API Jena/SDB 417
15.5.14. Jena/SDB diseños de base de datos 422
15.6. Anexos 423
15.6.1. RDF (Resource Description Framework) 423
15.6.2. RDFS (rdfschema) 425
15.6.3. D2r Server 425
15.6.4. D2RQ con Jena 428

Capítulo 16: HTML5

16.1. Introducción 431
16.2. Requerimiento de Software 431
16.2.1. PhoneGap 431
16.3. Canvas 434
16.3.1. Conceptos básicos sobre Canvas 434
16.3.2. drawimage() 441
16.4. Manejo de eventos en HTML5 442
16.4.1. Método addEventListener() 442
16.5. EjemploHTML5_basico 443

Capítulo 17: Google Maps Android API V3

17.1. Introducción 467
17.2. Coordenadas 473
17.3. Ejemplo Mapa1 474
17.4. Ejemplo Mapa2 480
17.5. Ejemplo Mapa3 485
17.6. Marcadores 488
17.7. Ejemplo Mapa4 489

Capítulo 18: Códigos ocultos en su dispositivo móvil

18.1. Código 501
18.1.1. Lista de los diferentes códigos 501
18.1.2. Códigos para obtener información del firmware 502
18.1.3. Códigos para ejecutar diferentes test de fábrica 502

Capítulo 19: Las mejores aplicaciones para Android

19.1. WhatsApp 505
19.2. Facebook 505
19.3. Spotbros 505
19.2.1. Características 505
19.4. Twitter 506
19.5. Otras aplicaciones 506

CAP.

1

Tecnologías móviles

1.1. INTRODUCCIÓN A LAS TECNOLOGÍAS MÓVILES

Durante los últimos años, la telefonía celular ha alcanzado niveles altos en el desarrollo de equipos y su evolución continúa siendo extremadamente acelerada. En efecto, la evolución de la tecnología móvil ha permitido llevar al mercado soluciones que brindan rentabilidad y ofrecen una mejor calidad de vida, ya que ponen al servicio del cliente la integración de las comunicaciones con la información. Hoy en día, se puede observar cómo el desarrollo de aplicaciones para dispositivos móviles
va creciendo en el mercado comercial, asimismo se puede apreciar cómo aplicaciones que antes solo se podían manejar en una computadora funcionan ahora en forma similar en un dispositivo móvil.



Las características concretas de este tipo de dispositivos han obligado a los desarrolladores de Java a construir un subconjunto de lenguaje y a reconfigurar sus principales bibliotecas para permitir su adaptación a un entorno limitado, propio de los dispositivos móviles. Todo esto hace que se necesite una nueva plataforma de desarrollo y ejecución sobre la que centrará nuestro estudio: Java 2 Micro Edition (o de forma abreviada J2ME) y Google Android.

1.2. LENGUAJE JAVA

Java es un un lenguaje compilado e interpretado. Todo programa en Java tiene que compilarse y el código que se genera, bytecodes, es interpretado por una máquina virtual.



De esta manera, se consigue la independencia de la máquina. El código compilado se ejecuta en la máquina virtual que sí es dependiente de la plataforma.

JAVA es un lenguaje orientado a objetos de propósito general. Aunque Java fue conocido como un lenguaje de programación de applets que se ejecutan en el entorno de un navegador web, se puede utilizar para construir cualquier tipo de proyecto. Su sintaxis es similar a la de C y C++.

El gran auge de Android y su crecimiento en forma exponencial abre una puerta muy importante para los desarrolladores lo que conduce a entender la razón del por qué está subiendo la demanda de desarrolladores para Android. El futuro de los desarrolladores es prometedor y quien piense que el mundo digital siempre se va a reducir a la telefonía está muy equivocado.

El presente libro tiene como objetivo la programación de Android la cual se basa en el lenguaje Java. Para facilitar el aprendizaje del lector, se utilizará Eclipse como entorno de desarrollo (IDEs).



Mis primeros pasos con Android

2.1. DEFINICIÓN DE ANDROID



Cuál es su historia

Android es más que un sistema operativo, es un conjunto de software para dispositivos móviles que incluye lo siguiente:

- Sistema operativo
- Middleware
- Aplicaciones

Android como sistema operativo para móviles fue construido sobre Linux Kernel 2.6 y está diseñado para plataformas móviles.

Android fue lanzado bajo la licencia Apache, una licencia libre y de código abierto lo que significa que cualquiera que quiera usar Android puede hacerlo descargando el código fuente. Por otra parte, los fabricantes de hardware pueden agregar sus propias extensiones propietarias. Este modelo de desarrollo simple hace de Android muy atractivo y, por ello, ha despertado el interés del mercado de consumidores de aplicaciones móviles. Actualmente, se ha incrementado el uso de dispositivos móviles como celulares, tablets, etc.

La principal ventaja de la adopción de Android es que ofrece un enfoque unificado para el desarrollo de aplicaciones. Los desarrolladores solo necesitan desarrollar para Android y sus aplicaciones deben ser capaces de funcionar con numerosos dispositivos diferentes, siempre y cuando los dispositivos funcionen con Android. En el mundo de los smartphones, las aplicaciones constituyen la parte más importante de la cadena de éxito. Los fabricantes de dispositivos ven a Android como su mejor esperanza para desafiar en el mercado a los iPhone, que cuenta con una gran base de aplicaciones.

A continuación, se describirá de forma resumida la historia de Android:

- En julio de 2006, Google compra Android Inc., empresa que desarrolla software para móviles de poco relevancia.
- En noviembre de 2007, nace Open Handset Alliance, una alianza comercial de 78 compañías para desarrollar estándares abiertos para dispositivos móviles. Algunos miembros son Google, HTC, Dell, Intel, Motorola, Qualcomm, Texas Instruments, Samsung, LG, T-Mobile, Nvidia, Wind River Systems (ver http://es.wikipedia.org).
- En octubre de 2008, se publica el proyecto Open Source Android (licencia Apache), se apertura Android Market, HTC primer inicio de Android.
- En diciembre de 2008, nuevos miembros aparecen como ARM Holdings, Athreos Communications, Asustek Computer Inc., Garmin Ltd, Softbank, Sony Ericsson, Toshiba Corp y Vodafone Group Pic.

- A finales de 2009, se produce aumentos de ventas de móviles con Android y la consolidación de Android Market.
- Para febrero de 2010, se venden alrededor de 60 000 móviles con Android.

2.2. ARQUITECTURA

Como se ha mencionado anteriormente, Android es una plataforma para dispositivos móviles que contiene un sistema operativo, librerías (bajo C y C++), framework para el desarrollo de aplicaciones y una suite de aplicaciones iniciales. Su diseño cuenta principalmente con las siguientes características:

- Los componentes básicos de las aplicaciones se pueden sustituir fácilmente por otros.
- Busca el desarrollo rápido de aplicaciones que sean reutilizables y verdaderamente portables entre diferentes dispositivos.
- Cuenta con su propia máquina virtual, Dalvik, que interpreta y ejecuta el código escrito en Java.
- Permite la representación de gráficos 2D y 3D.
- Posibilita el uso de bases de datos.
- Soporta un elevado número de formatos multimedia.
- Brinda servicio de localización GSM.
- Controla los diferentes elementos hardware: Bluetooth, Wi-Fi, cámara fotográfica o de video, GPS, acelerómetro, infrarrojos, etc., siempre y cuando el dispositivo móvil lo contemple.



- Cuenta con un entorno de desarrollo muy cuidado mediante un SDK disponible de forma gratuita.
- Ofrece un plug-in para uno de los entornos de desarrollo más populares, Eclipse, y un emulador integrado para ejecutar las aplicaciones.



2.2.1. Núcleo de Android

La capa más inmediata es la que corresponde al núcleo de Android. Android utiliza el núcleo de Linux 2.6 como una capa de abstracción para el hardware disponible en los dispositivos móviles. Esta capa contiene los drivers necesarios para que cualquier componente hardware pueda ser utilizado mediante las llamadas correspondientes.

Siempre que un fabricante incluya un nuevo elemento de hardware, lo primero que debe realizar para que pueda utilizarlo desde Android es crear las librerías de control o drivers necesarios, dentro de este kernel de Linux, embebido en el propio Android.

La elección de Linux 2.6 se ha debido principalmente a dos razones: la primera, su naturaleza de código abierto y libre se ajusta al tipo de distribución que se buscaba para Android (cualquier otra opción comercial disponible hoy día hubiera comprometido la licencia de Apache); la segunda, es que el kernel de Linux incluye numerosos drivers, además de contemplar la gestión de memoria, gestión de procesos, módulos de seguridad, comunicación en red y muchas otras responsabilidades propias de un sistemas operativo.

2.2.2. Librerías

La siguiente capa se corresponde con las librerías utilizadas por Android. Estas han sido escritas utilizando C/C++ y proporcionan a Android la mayor parte de sus capacidades más características. Junto al núcleo basado en Linux, estas librerías constituyen la parte central de Android. Entre las librerías más importantes de este nivel, se pueden mencionar las siguientes:

- Librería libc: Incluye todas las cabeceras y funciones según el estándar del lenguaje C. Todas las demás librerías se definen en este lenguaje.
- Librería Surface Manager: Es la encargada de componer los diferentes elementos de navegación de pantalla. Gestiona también las ventanas pertenecientes a las distintas aplicaciones activas en cada momento.
- OpenGL/SL y SGL: Representan las librerías gráficas y, por tanto, sustentan la capacidad gráfica de Android. OpenGL/SL maneja gráficos en 3D y permite utilizar, en caso de que esté disponible en el propio dispositivo móvil, el hardware encargado de proporcionar gráficos 3D. Por otro lado, SGL proporciona gráficos en 2D, por lo que será la librería más habitualmente utilizada por la mayoría de las aplicaciones. Una característica importante de la capacidad gráfica de Android es que es posible desarrollar aplicaciones que combinen gráficos en 3D y 2D.
- Librería Media Libraries: Proporciona todos los códecs necesarios para el contenido multimedia soportado en Android (video, audio, imágenes estáticas y animadas, etc.).
- FreeType: Permite trabajar, de forma rápida y sencilla, con distintos tipos de fuentes.
- Librería SSL: Posibilita la utilización de dicho protocolo para establecer comunicaciones seguras.
- Librería SQLite: Android ofrece la creación y gestión de bases de datos relacionales, pudiendo transformar estructuras de datos en objetos fáciles de manejar por las aplicaciones.
- Librería WebKit: Proporciona un motor para las aplicaciones de tipo navegador y forma el núcleo del actual navegador incluido por defecto en la plataforma Android.

2.2.3. Entorno de ejecución

Al mismo nivel que las librerías de Android, se sitúa el entorno de ejecución conformado por las Core Libraries que son librerías con multitud de clases de Java y la máquina virtual Dalvik.

2.2.4. Framework

Los dos últimos niveles de la arquitectura de Android están escritos enteramente en Java. El framework de aplicaciones representa fundamentalmente el conjunto de herramientas de desarrollo de cualquier aplicación. Toda aplicación que se desarrolle para Android; ya sean propias del dispositivo, las desarrolladas por Google o por terceras compañías o incluso las que el propio usuario cree, utiliza el mismo conjunto de API y el mismo framework, representado por este nivel. Entre las API más importantes ubicadas aquí, se pueden encontrar las siguientes:

- Activity Manager: Es un importante conjunto de API que gestiona el ciclo de vida de las aplicaciones en Android.
- Window Manager: Gestiona las ventanas de las aplicaciones y utiliza la librería ya vista en Surface Manager.
- Telephone Manager: Incluye todas las API, vinculadas a las funcionalidades propias del teléfono (llamadas, mensajes, etc.).
- Content Providers: Permite que cualquier aplicación comparta sus datos con las demás aplicaciones de Android. Por ejemplo, gracias a esta API, la información de contactos, agenda, mensajes, etc. será accesible para otras aplicaciones.
- View System: Proporciona un gran número de elementos para poder construir interfaces de usuario (GUI) como listas, mosaicos, botones, check-boxes, tamaño de ventanas, control de las interfaces mediante tacto o teclado, etc. Incluye también algunas vistas estándar para las funcionalidades más frecuentes.
- Location Manager: Posibilita la obtención de información de localización y posicionamiento a las aplicaciones, así como funcionar según esta.
- Notification Manager: Mediante la cual las aplicaciones, usando un mismo formato, comunican al usuario eventos que ocurren durante su ejecución: una llamada entrante, un mensaje recibido, conexión Wi-Fi disponible, ubicación en un punto determinado, etc. Si lleva asociada alguna acción, en Android se la denomina «Intent» (por ejemplo, atender una llamada recibida) la cual se activa mediante un simple clic.
- XMPP Service: Colección de API para utilizar este protocolo de intercambio de mensajes basados en XML.

2.2.5. Aplicaciones

El último nivel del diseño arquitectónico de Android son las aplicaciones. Este nivel incluye tanto las aplicaciones incluidas por defecto de Android como aquellas que el usuario vaya añadiendo posteriormente, ya sean de terceras empresas o de su propio desarrollo. Todas estas aplicaciones utilizan servicios, API y librerías de los niveles anteriores.

2.2.4. Framework

Los dos últimos niveles de la arquitectura de Android están escritos enteramente en Java. El framework de aplicaciones representa fundamentalmente el conjunto de herramientas de desarrollo de cualquier aplicación. Toda aplicación que se desarrolle para Android; ya sean propias del dispositivo, las desarrolladas por Google o por terceras compañías o incluso las que el propio usuario cree, utiliza el mismo conjunto de API y el mismo framework, representado por este nivel. Entre las API más importantes ubicadas aquí, se pueden encontrar las siguientes:

- Activity Manager: Es un importante conjunto de API que gestiona el ciclo de vida de las aplicaciones en Android.
- Window Manager: Gestiona las ventanas de las aplicaciones y utiliza la librería ya vista en Surface Manager.
- Telephone Manager: Incluye todas las API, vinculadas a las funcionalidades propias del teléfono (llamadas, mensajes, etc.).
- Content Providers: Permite que cualquier aplicación comparta sus datos con las demás aplicaciones de Android. Por ejemplo, gracias a esta API, la información de contactos, agenda, mensajes, etc. será accesible para otras aplicaciones.
- View System: Proporciona un gran número de elementos para poder construir interfaces de usuario (GUI) como listas, mosaicos, botones, check-boxes, tamaño de ventanas, control de las interfaces mediante tacto o teclado, etc. Incluye también algunas vistas estándar para las funcionalidades más frecuentes.
- Location Manager: Posibilita la obtención de información de localización y posicionamiento a las aplicaciones, así como funcionar según esta.
- Notification Manager: Mediante la cual las aplicaciones, usando un mismo formato, comunican al usuario eventos que ocurren durante su ejecución: una llamada entrante, un mensaje recibido, conexión Wi-Fi disponible, ubicación en un punto determinado, etc. Si lleva asociada alguna acción, en Android se la denomina «Intent» (por ejemplo, atender una llamada recibida) la cual se activa mediante un simple clic.
- XMPP Service: Colección de API para utilizar este protocolo de intercambio de mensajes basados en XML.

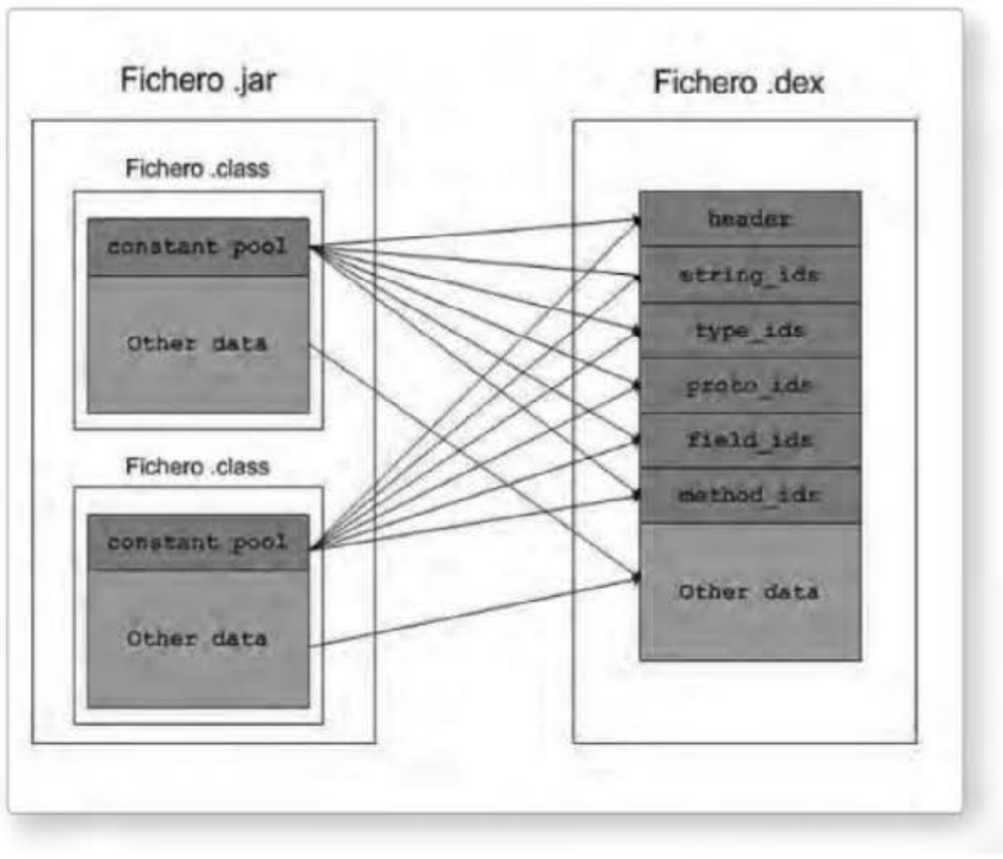
2.2.5. Aplicaciones

El último nivel del diseño arquitectónico de Android son las aplicaciones. Este nivel incluye tanto las aplicaciones incluidas por defecto de Android como aquellas que el usuario vaya añadiendo posteriormente, ya sean de terceras empresas o de su propio desarrollo. Todas estas aplicaciones utilizan servicios, API y librerías de los niveles anteriores.

2.3. DALVIK

En Android, todas las aplicaciones se programan en el lenguaje Java y se ejecutan mediante una máquina virtual de nombre Dalvik, específicamente diseñada para Android. Esta máquina virtual ha sido optimizada y adaptada a las peculiaridades propias de los dispositivos móviles (menor capacidad de proceso, baja memoria, alimentación por batería, etc.) y trabaja con ficheros de extensión .dex (Dalvik ejecutables). Dalvik no trabaja directamente con el bytecode de Java, sino que lo transforma en un código más eficiente que el original, pensado para procesadores pequeños.

Gracias a la herramienta dx, esta transformación es posible por los ficheros .class de Java que se compilan en ficheros .dex, de forma que cada fichero .dex puede contener varias clases. Después, este resultado se comprime en un único archivo de extensión .apk (Android Package) encargada de distribuir en el dispositivo móvil. Dalvik permite varias instancias simultáneas de la máquina virtual y, a diferencia de otras máquinas virtuales, está basada en registros y no en pila lo que implica que las instrucciones son más reducidas y el número de accesos a la memoria es menor. Asimismo, Dalvik no permite la compilación Just-in-Time.



Según los responsables del proyecto, la utilización de esta máquina virtual responde a un deseo de mejorar y optimizar la ejecución de aplicaciones en dispositivos móviles, así como evitar la fragmentación de otras plataformas como Java ME. A pesar de esta aclaración oficial, no ha pasado inadvertido que con esta maniobra Android ha esquivado la necesidad de utilizar directamente Java ME, evitando así que los futuros desarrolladores de aplicaciones tengan que comprar alguna licencia ni tampoco tengan que publicar el código fuente de sus productos. El lenguaje Java utilizado en Android no sigue ningún JSR determinado y Google se afana en recordar que Android no es una tecnología Java, sino que simplemente utiliza este lenguaje en sus aplicaciones.

2.4. REQUERIMIENTOS MÍNIMOS DE HARDWARE



No hay requisitos mínimos de un dispositivo Android. Sin embargo, Google ha recomendado las especificaciones de hardware siguientes, que prácticamente todos los dispositivos disponibles para Android deberán cumplir significativamente más a menudo:

- *CPU*: Android se ejecuta no solo en una arquitectura ARM, sino también en la arquitectura x86.

- *128 MB de RAM*: Esta especificación es mínima; ya que los actuales dispositivos de gama alta incluyen 512 MB de RAM y 1 GB de RAM, dispositivos que se espera en un futuro muy cercano.
- *256 MB de memoria Flash*: Esta cantidad mínima de memoria para almacenar la imagen del sistema y aplicaciones. Durante mucho tiempo, esta falta de memoria era la mayor queja de los usuarios de Android ya que las aplicaciones de terceros puede ser instalado solo en la memoria Flash. Esto cambió con el lanzamiento de Froyo.
- *Mini o microtarjeta de almacenamiento SD*: La mayoría de los dispositivos vienen con unos cuantos gigabytes de almacenamiento de tarjeta SD, que puede ser sustituido por grandes tarjetas SD por el usuario.
- *Color de 16 bits de tamaño medio de gráficos de video array (HVGA) y TFT LCD con pantalla táctil*: Antes de la versión 1.6 para Android, solo las pantallas HVGA (480x320 píxeles) fueron apoyados por el sistema operativo. Desde la versión 1.6, pantallas de baja y alta resolución son compatibles. La corriente de dispositivos de gama alta han Wide Video Matriz Graphis (WVGA) pantallas (800 x 480, 848 x 480, 852 x 480 píxeles) y algunos de gama baja deporte dispositivos de un cuarto de tamaño de video Graphics Array (QVGA) con 320 x 280 píxeles pantallas. Las pantallas táctiles capacitivas son casi siempre y solo son capaces de un solo toque en la mayoría de los dispositivos más antiguos.

Por supuesto, hay equipos mucho más en reales de los dispositivos Android. Casi todos los teléfonos tienen GPS, un acelerómetro y una brújula. Muchos de ellos también disponen de proximidad y sensores de luz. Estos periféricos ofrecen a los desarrolladores nuevas formas de juego para permitir al usuario interactuar con el juego (ver CS). Algunos dispositivos tienen un teclado QWERTY completo, así como una bola de seguimiento. Este último se encuentra con más frecuencia en los dispositivos de HTC.





2.5. Versiones de Android

Android posee las siguientes versiones:

Versión	Release	CodeName
1.1	9 febrero 2009	
1.5	30 abril del 2009	Cupcake
1.6	15 de setiembre 2009	Donut
2.0/2.1	26 October 2009	Eclair
2.2	20 May 2010	Froyo
2.3	6 December 2010	Gingerbread
3.0/3.1/3.2	22 February 2011	Honeycomb
4.0 (Ice Cream Sandwich)	octubre 2011	Ice Cream Sandwich
4.1 (Jelly Bean)	Junio 2012	Jelly Bean
4.2 (Jelly Bean)	**Noviembre 2012**	Jelly Bean


The first Android device, the HTC Dream (G1), featuring Android 1.0.
23 September, 2008
Android 1.0
► Integration with Google Services
► Web browser to show, zoom and pan full HTML and XHTML web pages; multiple pages shown as windows.
► Android Market app downloads and updates.
► Multitasking, Instant Messaging, Wi-Fi and Bluetooth.
Android 1.1 update for Android was released for T-Mobile G1 only.
9 February, 2009
Based on Linux kernel 2.6.27, the official 1.5 (Cupcake) update for Android was released.
30 April 2009
Android 1.5 (Cupcake)
► Faster Camera start-up and image capture.
► Much faster acquisition of GPS location (powered by SUPL AGPS).
► On-screen soft keyboard.
► Directly upload videos to YouTube and Picasa.
Based on Linux kernel 2.6.29, the 1.6 (Donut) SDK was released.
15 September 2009
Android 1.6 (Donut)
► Quick Search Box and Voice Search
► integrated camera, camcorder, and gallery, toggle between still and video capture modes
► Battery usage indicator
► CDMA Support
► Multilingual text-to-speech function
Based on Linux kernel 2.6.29, the 2.0 (Eclair) SDK was released.
26 October 2009
Android 2.0 (Eclair)
► Multiple accounts for email and contact synchronization.
► Microsoft Exchange Support for syncing of e-mail.
► Bluetooth 2.1 support
► New browser User Interface and support for HTML5.
► New calendar features
3.90 %
The 2.0.1 SDK was released.
3 December 2009
The 2.1 SDK was released.
12 January 2010
Based on Linux kernel 2.6.32, the 2.2 (Froyo) SDK was released.
20 May 2010
Android 2.2 (Froyo)
► New tips widget for homescreen.
► Improved Exchange support.
► Hotspot support.
► Multiple keyboard languages.
► Adobe Flash 10.1.
17.70 %
The 2.3 (Gingerbread) SDK was released.
6 December 2010
Android 2.3 (Gingerbread)
► UI refinements for simplicity and speed.
► New keyboard for faster text input.
► One-touch word selection and copy/paste.
► Near Field Communication (NFC).
► Internet Calling.



2.5.1. Android 4.1

Fue anunciado en el 2012 por Google I/O conferencia, Android 4.1 Jelly Bean es sin duda más importante que su versión 0.1. Incrementó sobre Ice Cream Sandwich que trata de hacernos creer. Representa tanto un reinicio en la estrategia de abanderamiento de la tablet de Google (después de haber sido introducido junto con el Nexus Asus) y un gran refinamiento en la experiencia de usuario, completamente rediseñada que debutó en Android 4.0.

SAMSUNG GALAXY S3 CON ANDROID 4.1.1 JELLY BEAN: ANÁLISIS Y RENDIMIENTO (II)

EXPERIENCIA DE USUARIO



«A pesar de que estas impresiones debéis interpretarlas con cierta cautela pues no se trata de la versión final, creemos que las últimas Beta se acercarán mucho al resultado final que ofrecerá la OTA oficial de Android 4.1.1 Jelly Bean para el Samsung Galaxy S3. En primer lugar, hemos destacar la mejora notable de la fluidez de la interfaz gráfica. La optimización de Android en esta última versión se hace notar. Tal vez pueda ser debido a la novedad Butter Proyect, pero lo que sí es cierto es que el sistema operativo del Galaxy S3 va más fluido y los efectos gráficos son más "elegantes". También mejora la percepción de fluidez de la transición de pantallas y aplicaciones del launcher. Creíamos que este punto era más que satisfactorio en ICS, pero tras probar Jelly Bean nos damos cuenta de la mejoría. No hay desvanecimiento, en nuestras pruebas hemos realizado incesantes transiciones entre una y otra pantalla, con numerosos widgets sin problema alguno. Lo mismo sucede con el pinch zoom en la pantalla de inicio, que nos permite seleccionar directamente una pantalla del escritorio.

RENDIMIENTO



Asimismo hemos notado que la ejecución de aplicaciones y funciones es muy rápida, más que antes. En algunos casos apenas existe retardo. Lo podemos comprobar a la hora de acceder al menú de ajustes, por poner un ejemplo. Lo mismo sucede con el Dialer. En Ice Cream Sandwich era una labor que requería cierto nivel de paciencia. Ahora ese retardo es apenas apreciable. Si acaso se presenta alguna ralentización la primera vez que lo pulsamos, pero posteriormente es instantáneo. Al igual que la lentitud en la ejecución del listado de contactos, Samsung también ha mejorado el retardo que sufría la acción del botón físico Home. Ahora la pulsación del botón nos deriva rápidamente a la pantalla inicial del sistema. Sin duda, la mejoría de estos dos puntos será del agrado de los propietarios del dispositivo puesto que se trata de algunas de las quejas más frecuentes.

NAVEGADOR WEB



Otro de los puntos en los que hemos notado una mejora muy apreciable es el navegador web. Su funcionamiento es muy suave y fluido, ya sea el scroll a través de la página que visites, el acceso a favoritos o cualquier otra acción. La carga de las páginas, aunque en gran medida influenciada por la velocidad de la red, es verdaderamente muy rápida. Lo mismo sucede con gestos

táctiles como el zoom con dos dedos. Lo que no nos ha gustado es el ajuste de la página a las dimensiones de la pantalla cuando hacemos zoom con un doble toque en el display. No se ajusta el texto correctamente. Aunque de serie Jelly Bean no ofrece soporte para Flash Player, el hecho de conservarlo desde Ice Cream Sandwich nos ha permitido usarlo sin mayores problemas. Sin lugar a dudas el rendimiento del navegador es de los aspectos que más nos ha impresionado.

TEST



Hablando de rendimiento, hemos probado diversas aplicaciones para medir el grado de optimización de Android 4.1.1 Jelly Bean Beta. Con AnTuTu hemos registrado una puntuación superior a los 10.000 puntos. En el caso de Quadrant, la puntuación varía entre 5.500 y 5.700 puntos, aunque se han registrado casos en la red de usuarios que han sobrepasado los 6.000 puntos sin modificación alguna. No obstante, os ofrecemos estos detalles como mero dato informativo puesto que en la mayor parte de las ocasiones no es el método más certero para cuantificar la potencia de los smartphones.

AUTONOMÍA



Todos esperamos que con cada actualización mejore notablemente la autonomía. Sin embargo, la actualización Android 4.1.1 Jelly Bean no trae grandes novedades en esta parcela. Durante los días que hemos vigilado los puntos más interesantes en un uso diario no hemos experimentado una ganancia apreciable. En realidad hemos tenido la necesidad de cargar el móvil como habitualmente hacíamos con ICS, todas las noches. Eso nos arroja una autonomía de escasas 24 horas. No obstante, es importante matizar que el brillo de la pantalla lo hemos mantenido al máximo, sin pretender prolongar la duración de la batería. Además, hemos hecho un uso medio-alto del smartphone, buscando el máximo rendimiento, sin activar el ahorro de energía, con llamadas, sincronización de datos activada para todas las aplicaciones, Whatsapp, navegación web, Google Maps y redes sociales. Por tanto, no dudamos que los más cuidadosos en este aspecto puedan conseguir mejores tiempos».

Fuente: www.movilzona.es

ANDROID 4.1 JELLY BEAN INTRODUCE LOS SISTEMAS DE SEGURIDAD ASLR Y DEP PARA HACERSE AÚN MÁS FUERTE

«[...] Con Android 4.1, Google introduce dos nuevas tecnologías llamadas ASLR y DEP, que ya encontrábamos en otros sistemas operativos como Windows o iOS. En el caso de ASLR, ya contábamos con ella en Android 4.0 Ice Cream Sandwich pero su funcionamiento dejaba que desear y era muy vulnerable de cara a los hackers. Es por esto que con Jelly Bean la mejoran y además introducen DEP.



Básicamente, ASLR se encarga de establecer los datos de las aplicaciones de forma aleatoria en la memoria de forma que el hacker, antes de atacar, debe de saber como se están establecidos dichos datos. Por otra parte, DEP se encarga de asegurar que ninguna aplicación ejecuta ningún tipo de código de otra parte de la memoria que no es correspondiente a esa aplicación.

Con estas dos mejoras, Jelly Bean hace a Android mucho más seguro y fuerte ante los constantes ataques que reciben los sistemas operativos en general. Es evidente que no es perfecto, pero estos dos nuevos sistemas de seguridad nos permiten estar mucho más tranquilos frente a los hackers [...]».

Fuente: www.elandroidelibre.com

ANALIZAMOS TODO LO QUE JELLY BEAN PUEDE OFRECER AL OJITO DERECHO DE SAMSUNG, EL GALAXY S III



GOOGLE NOW

«Seguro que la mayoría ya sabeis de que se trata, pues otra de las grandes innovaciones que nos trajeron los chicos de Google fue la llegada de una nueva aplicación interna que se asemejaba considerablemente al asistente virtual "Siri" de Apple pero con resultados de búsqueda más veloces. Google Now registra nuestras ubicaciones y nos muestra información en forma de tarjetas y notificaciones que pueden hacer nuestro día a día más fácil.

Lástima que las opciones que han llegado a España sean mas bien escasas y nos deje una sensación agridulce, sobre todo por el consumo de batería que supone. Para poder hacer uso de esta aplicación, bastará con pulsar el boton de "Menú" durante un segundo aproximádamente».

Fuente: andro4all.com

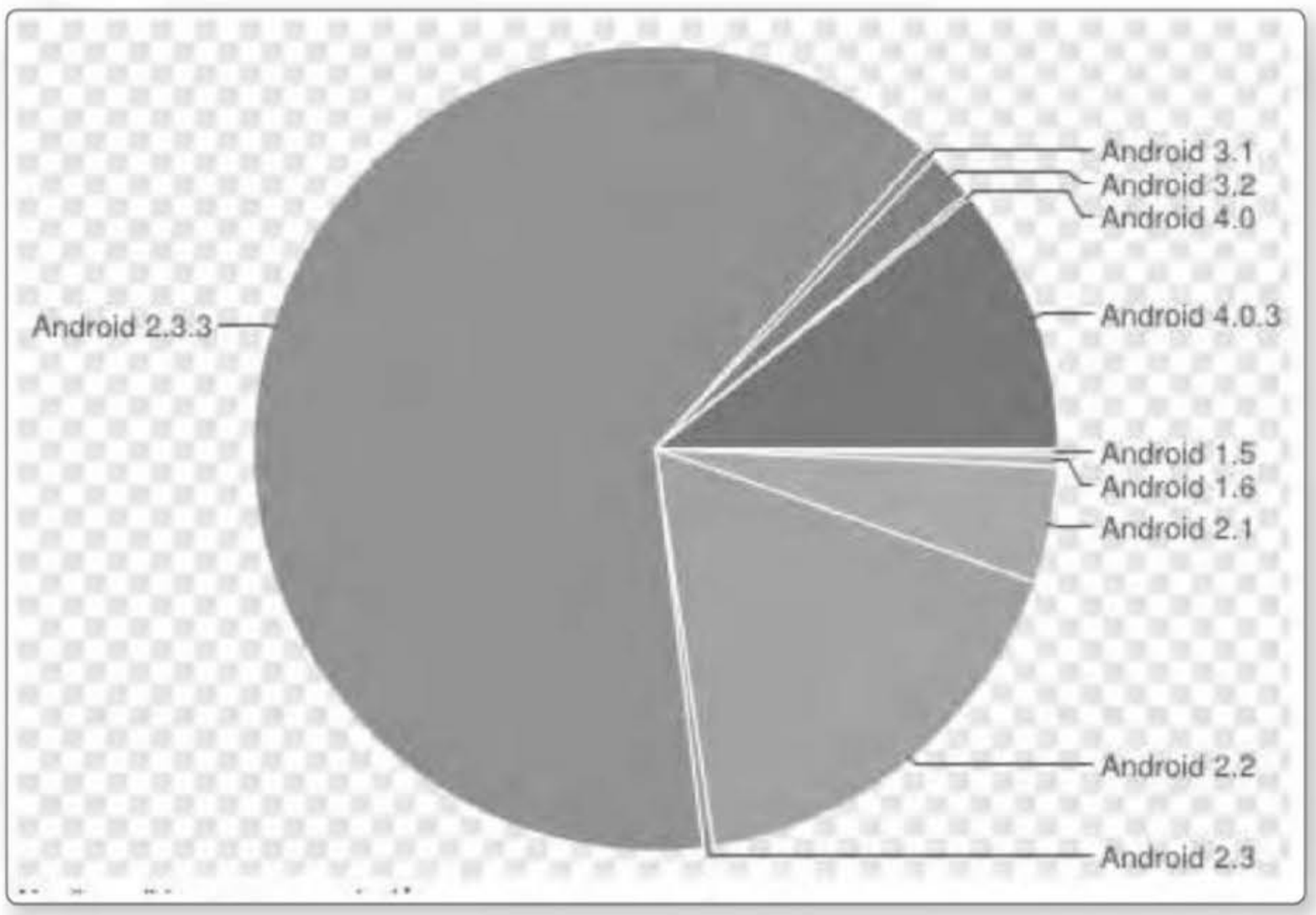
2.5.2. Android 4.2

[ocultar]Android 4.2 Jelly Bean			
Versión	**Fecha lanzamiento**	**Características**	**Imagen(es)**
4.2		• Fotos panorámicas con "Photo Sphere" • teclado con escritura gesticular • Mejoras al Lock screen, incluyendo soporte a widgets y la posibilidad de deslizar con el dedo e ir directamente a la cámara84 • Controles en las notificaciones ("Quick Settings") • Protectores de pantalla "Daydream",que muestran información cuando el equipo está inactivo o conectado por USB (docked). • Múltiples cuentas de usuario (sólo en tablets) • Soporte para pantallas inalambricas (Miracast) • Mejoras en accesabilidad: triple-tap to magnify the entire screen, pan and zoom with two fingers. Speech output and Gesture Mode navigation for blind users • Nueva aplicación del reloj con función de reloj mundial,cronómetro y temporizador.r • Todos los dispositivos ahora usan el mismo diseño de interfase, previamente adaptado desde los teléfonos en 4.1 para las tablets pequeñas (con botones centrados en la pantalla, la barra de sistema en la parte superior de la pantalla, y una pantalla de inicio con un dock y el menu de aplicaciones centrado), independientemente del tamaño de la pantalla • Incrementado el número de notificaciones extendidas y notificaciones accionables para más aplicaciones, permitiendo responder a ciertas notificaciones con la barra de notificaciones sin lanzar la aplicación directamente • SELinux • Siempre-activa VPN • Confirmación SMS Premium SMS85 86 87	
4.2.1		• Arreglado un fallo que en la aplicación "People", diciembre no era mostrado cuando se agregaba a un contacto[89] • Agregado Bluetooth gamepads y joysticks como dispositivos HID soportados	
4.2.2		• Arreglados fallos del streaming de audio por Bluetooth[91] . • Arreglado fallo de cámara que dejaba flash pegado[92] . • El Presionado-largo sobre los íconos de Wi-Fi y Bluetooth en Quick Settings ahora cambia el estado activado / desactivado (on/off) • Nuevas notificaciones de descarga, cuales ahora muestran el porcentaje y el tiempo estimado que queda para las descargas activas de aplicaciones. • Nuevos sonidos para la carga en forma inalámbrica y batería baja. • Removida la opción de mostrar todas las llamadas en la lista de llamadas de la aplicaciñon teléfono. • Nueva animación de la aplicación galería la cual permite carga más rápida. • lista blanca de depuración USB. • Mejoras en rendimiento y arreglo de fallos menores.	

Fuente: es.wikipedia.org

2.5.3. Cuota de las versiones

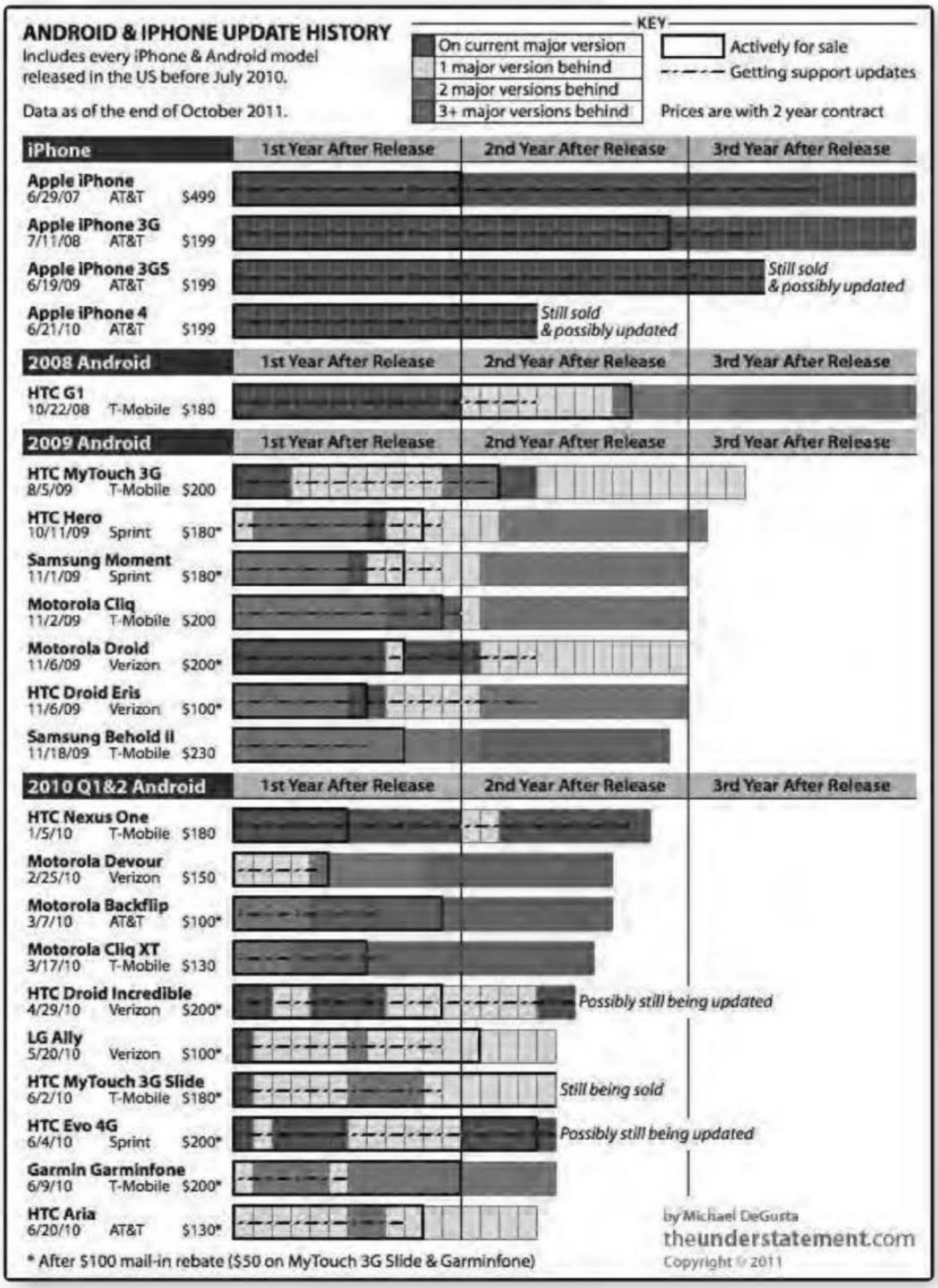
PLATAFORMA	NIVEL DE API	%
4.0. x Ice Cream Sandwich	14-15	10,9%
3.x.x Honeycomb	11-13	2,4%
2.3.x Gingerbread	9-10	64,0%
2.2 Froyo	8	17,3%
2.1 Eclair	7	4,7%
1.6 Donut	4	0,5%
1.5 CupcakeS	3	0,2%



2.6. FRAGMENTACIÓN EN ANDROID

La fragmentación es uno de los problemas más críticos que tiene que enfrentar Android. Se trata que, al contrario que Apple cuyos dispositivos siempre tienen la última versión de su sistema operativo, rara vez ocurre esto en Android.

- *Ventajas:* La fragmentación permite a los usuarios una mayor variedad y facilidad al momento de elegir su dispositivo, pese a que tienen que esperar para disfrutar de las actualizaciones de su sistema operativo y todas las novedades que ofrece. Esto se debe a que Google no está gestionando demasiado bien esta área.
- *Desventajas:* Esta fragmentación también ocasiona algunas dificultades en la tarea de los desarrolladores; debido a que el diseño de una aplicación cambia según el dispositivo y la versión. Por ello, existen muchas aplicaciones en Google Play que desempeñan la misma función, pero cada una para un dispositivo y su versión.



Nota:

Fuente: comunidad.movistar.es. Color verde: teléfono soporta la última versión del sistema. Color amarillo: 1 versión anterior. Color naranja: 2 versiones anteriores. Color rojo: 3 versiones anteriores.

«En resumen, creo que es cierto que los usuarios pueden beneficiarse de esta variedad de versiones del Androide, pero también es cierto que esta fragmentación es probablemente el mayor problema al que Google tiene que poner solución» (Fuente: comunidad.movistar.es).

2.7. MOSTRAR LA VERSIÓN ACTUAL DE UN DISPOSITIVO MÓVIL

2.7.1. Códigos

En este apartado, se mostrarán unas líneas de código para determinar la versión actual:
Main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"    android:layout_width="fill_parent"
    android:layout_height="fill_parent"    >
<TextView    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"    />
<TextView    android:id="@+id/device"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"    />
<TextView    android:id="@+id/model"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"    />
<TextView    android:id="@+id/product"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"    />
<TextView    android:id="@+id/codename"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"    />
<TextView    android:id="@+id/incremental"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"    />
<TextView    android:id="@+id/release"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"    />
<TextView    android:id="@+id/sdk"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"    />
<TextView    android:id="@+id/sdk_int"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"    />
</LinearLayout>
```

EjemploVersionActivity.java

```
public class EjemploVersionActivity extends Activity {
    /** Called when the activity is first created. */
    @Override    public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.main);
            TextView device = (TextView)findViewById(R.id.device);
            TextView model = (TextView)findViewById(R.id.model);
            TextView product = (TextView)findViewById(R.id.product);
            TextView codename = (TextView)findViewById(R.id.codename);
            TextView incremental = (TextView)findViewById(R.id.incremental);
            TextView release = (TextView)findViewById(R.id.release);
            TextView sdk = (TextView)findViewById(R.id.sdk);
            TextView sdkInt = (TextView)findViewById(R.id.sdk_int);
            device.setText("android.os.Build.DEVICE: " + android.os.Build.DEVICE);
            model.setText("android.os.Build.MODEL: " + android.os.Build.MODEL);
            product.setText("android.os.Build.PRODUCT: " + android.os.Build.PRODUCT);
            codename.setText("android.os.Build.VERSION.CODENAME: " + android.os.Build.
VERSION.CODENAME);
```

```
            incremental.setText("android.os.Build.VERSION.INCREMENTAL: " + android.
os.Build.VERSION.INCREMENTAL);
            release.setText("android.os.Build.VERSION.RELEASE: " + android.os.Build.
VERSION.RELEASE);
            sdk.setText("android.os.Build.VERSION.SDK: " + android.os.Build.VERSION.
SDK);
            sdkInt.setText("android.os.Build.VERSION.SDK_INT:     "     +     String.
valueOf(android.os.Build.VERSION.SDK_INT));
            }
    }
```

2.7.2. Ejemplos

- *En un Galaxy Y pro*





- *En una Samsung Galaxy Tab 8.9*

- *En una ASUS Transformer TF101*



2.8. DISPOSITIVOS ANDROID EN EL MERCADO

Android está presente en una gran multitud de dispositivos, tanto teléfonos móviles como tablets, portátiles y próximamente en televisiones.

2.8.1. Smartphones



Los teléfonos inteligentes son el tipo más popular de los dispositivos Android. Contienen casi todos los componentes descritos anteriormente y, en general, son los más conocidos. A finales de 2010, los dispositivos Android representan casi la cuarta parte del mercado de smartphones en Estados Unidos. Es ampliamente aceptado que Android superará al iPhone y quizás en última instancia, será la plataforma de teléfonos inteligentes más populares.

«Dentro de los dispositivos móviles, un smartphone (cuya traducción en español sería "'teléfono inteligente") es una evolución del teléfono móvil tradicional que cuenta con ciertas características y prestaciones que lo acercan más a un ordenador personal que a un teléfono tradicional.

Entre dichas características, se puede encontrar una mejora en la capacidad de proceso y almacenamiento de datos, conexión a Internet mediante Wi-Fi, pantalla táctil, acelerómetro, posicionador geográfico, teclado QWERTY y diversas aplicaciones de usuario como navegador web, cliente de correo, aplicaciones ofimáticas, reproductores de video y audio, etc. incluyendo la posibilidad de descargar e instalar otras nuevas» (Jaime Aranaza Tudela).



2.8.2. Tablets



Google diseña una tablet de bajo costo y potente, para competir con Amazon. Se llamará Nexus 7 y será fabricada por Asus, además contará con el nuevo sistema operativo Android 4.1 Jelly Bean.

Entre sus especificaciones importantes cuenta con una pantalla de 7 pulgadas de 1280 por 800 HD, resistente ante arañazos y golpes gracias a su pantalla con la tecnología Corning glass. En cuanto al procesador, hablamos de un Nvidia Tegra 3 con 1 GB de RAM, dispositivo que estará disponible en versiones de 8 y 16 GB. Además cuenta con una cámara frontal, Wi-Fi, Bluethoth, nueve horas de batería y tecnología de pago NFC.

La tablet Nexus 7 pesa 340 g y se trata de un terminal específicamente, diseñado para escuchar música, ver video, leer libros y revistas, usar aplicaciones y juegos. Un dispositivo potente y veloz para la diversión y optimizar el consumo de contenidos multimedia.



«Una tableta o tablet PC es una computadora portátil con la que se puede interactuar a través de una pantalla táctil o multitáctil, sin necesidad de un teclado físico ni mouse» (Fuente: www.lostiempos.com). Para trabajar con la computadora, el usuario puede utilizar una pluma stylus o los dedos.

El término se hizo popular en el 2001, cuando la empresa estadounidense Microsoft presentó el concepto de «Microsoft Tablet PC». Actualmente, la tablet se usa principalmente para referirse a los gadgets o a los aparatos controlados principalmente mediante una pantalla táctil, pero no con la intención de ejecutar sistemas operativos de PC, en general, ni aplicaciones de ellos.

Por otro lado, algunas tablets incorporan un conector jack de 3.5 o un HDMI para poder conectar la tablet a un televisor común o a un monitor de computadora.



Actualmente, hay una serie de tablets con Android en el mercado. Sin embargo, el dispositivo más reciente y ampliamente publicitado es el Samsung Galaxy Tab. El dispositivo de 7 pulgadas tiene prácticamente todos los componentes de un inteligente Android teléfono, pero en un formato más grande.

Sony, actualmente, posee la tablet Xperia Tablet S , tiene una pantalla de 9.4", procesador NVIDIA tegra3 quad-core y cuenta con Android 4.0; asimismo, ofrece hasta 10 horas continuas de navegacion a Internet. Posee dos cámaras de HD, la frontal con 1 Mpx y la trasera es de 8 Mpx de resolución. Por último, cabe agregar que posee Bluetooth 3.0 y una conexión Wi-Fi. Hay modelos entre 16, 32 y 64 Gb.

2.8.3. E-reader



Los e-reader son lectores de libros electrónicos donde se almacena información y cuenta con herramientas tecnológicas adicionales, disponible a los usuarios. En otras palabras, e-reader:

«es un dispositivo electrónico que reproduce los contenidos de libros electrónicos, con una calidad de lectura como en papel gracias a la tecnología de tinta electrónica. En el mercado existe una gran confusión entre el término e-reader (dispositivo que actúa como soporte de lectura) y el término e-book (contenido digital o libro electrónico), utilizándose en muchos casos erróneamente el segundo para designar al soporte de lectura. La oferta de este tipo de dispositivos, y las prestaciones de los mismos, ha evolucionado espectacularmente durante el año 2009. En el caso del mercado español, existen varios modelos, los más conocidos son Reader de Sony, bq Avant de Bq readers, Papyre de Grammata y Amazon Kindle» (Fuente: Wikipedia).

2.8.4. Netbooks



«Un netbook es un subportátil, es decir, una categoría de ordenador portátil de bajo coste y generalmente reducidas dimensiones, lo cual aporta una mayor movilidad y autonomía. Son utilizados principalmente para navegar por Internet y realizar funciones básicas como procesador de texto y de hojas de cálculo» (Fuente: http://www.taringa.net) .

El término fue creado por la empresa británica Psion para utilizarlo en una gama de computadoras portátiles (llamadas laptops, en algunos países) similares a un palmar, lanzada en 1999. Psion obtuvo el registro de la marca en los Estados Unidos en noviembre de 2000. El término fue reintroducido por Intel en febrero de 2008 con su actual significado.

2.8.5. Reproductores MP4



Un reproductor MP4 multimedia es aquel dispositivo móvil que ha sido específicamente diseñado para proporcionar al usuario la reproducción de uno o varios formatos de datos digitales, ya sea audio, video o imágenes. Dentro de estos dispositivos están los reproductores de MP3, los DVD portátiles.

2.8.6. Internet TV

Google está intentando reducir la separación que hay entre ver la televisión de difusión y los contenidos de Internet. Los dispositivos van desde los aparatos de televisión completo con Android incorporado, para set-top boxes que se conectan a los televisores existentes. Pero la clave es aprovechar a Android como sistema operativo base, la integración de Internet y programas de televisión, así como proporcionar un marco para que los desarrolladores puedan crear nuevas aplicaciones específicas.



2.8.7. Android 4.0, Mini Android TV, Zero Devices Z802, Wi-Fi y HDMI



Convierte a tu TV en una computadora con un sistema operativo Android 4.0 (Smart TV).

Puedes navegar en Internet, recibir e-mails, ver videos en YouTube, ver películas y todo lo que quieras usando su TV.

2.8.8. Quake 3 para Android



Quake 3 ese mítico videojuego que trascendió durante toda una generación, ahora está disponible para Android. Este juego fue portado a distintos dispositivos, entre ellos el N900, que funciona con un núcleo Linux, igual que los androides. Por esta razón, ha sido portado para Android desde esa versión por un propietario de un MotorolaMilestone llamado «Thunderbird2k» gracias a la increíble capacidad 3D de los dispositivos.

2.8.9. ARCHOS GamePad



«El ARCHOS GamePad combina los 14 botones físicos y dos joysticks con un parametraje revolucionario de botones (Exclusividad ARCHOS), que permiten sensaciones reales» (Fuente: http://www.archos.com).

El ARCHOS GamePad funciona con Android 4.1 Jelly Bean. Es la plataforma ideal para los juegos, las aplicaciones, el Internet, los archivos multimedia, los e-mails, los libros, etc. Por tanto, se trata de una verdadera tablet.

2.8.10. Radio

Se trata de un teléfono inteligente muy interesante, que si bien muestra una configuración solvente, aún está alejado de los dispositivos de alta gama que manejan otros fabricantes y operadores. Cuenta con un diseño moderno y un acabado que es atractivo a la vista. Tiene una pantalla táctil de 3.8 pulgadas, cámara posterior de 5 megapíxeles, procesador de 800 MHz y usará el sistema operativo.



2.8.11. Automóviles



Un área que tiene posibilidades muy interesantes es la integración de dispositivos Android en los automóviles; por lo general, como parte del sistema de entretenimiento. Hasta la fecha, estos sistemas son específicos de cada fabricante de vehículos lo que ha desembocado en sistemas que varían en las características, la estabilidad y eficacia.

Si los fabricantes integran la funcionalidad completa del sistema operativo Android en constante evolución, les permitiría centrarse en la experiencia del usuario.

El primer coche de producción con Android es el Roewe 350 desarrollado y distribuido en China por la Corporación Automotriz de Shanghai.



Foto: http://es.engadget.com

2.8.12. Kit Car



Asteriod CK es un Kit Car de fácil instalación en su vehículo tras las oportunas conexiones, ofreciendo como novedad una pantalla de 3,2 pulgadas y todas las funciones que anteriormente se encontraban en la plataforma. Evidentemente, ofrece funciones de manos libres, pero se trata de un detalle más evidente teniendo en cuenta el fabricante.

2.8.13. Otros dispositivos

Cámara Galaxy Camera



La primera cámara Samsung con 3G. Haz fotos de calidad profesional y compártelas como en un Smartphone junto con la potencia del sistema operativo Android 4.1 Jelly Bean™. Es la cámara más inteligente que hayas visto, con una increíble calidad fotográfica y un amplio abanico de modos de disparo profesionales, funciones de edición y numerosas aplicaciones. Convertirte en un profesional nunca ha sido tan fácil, ni tan divertido.

Fuente: www.samsung.com/es/galaxycamera/

Hay un número creciente de nuevos dispositivos Android que se están convirtiendo en una necesidad. A continuación, se presentan algunos ejemplos de otros tipos de dispositivos Android:

- Artefactos para el hogar: lavadoras y microondas
- Equipo de oficina: fotocopiadoras
- Teléfonos fijos, audio y video
- Dispositivos especiales para juegos
- Impresoras

2.9. NOTICIAS SOBRE ANDROID

A. ESTIMACIÓN DE DISPOSITIVOS ANDROID EN EL MERCADO SEGÚN ANDY RUBIN

Cada día se activan 900.000 nuevos dispositivos Android

Andy Rubin, hombre fuerte de Google y creador del sistema operativo móvil, ha twiteado esta noticia pocas horas antes de que Apple comience su gran conferencia para presentar el iOS 6. ¿Estamos ante un nuevo episodio de guerra fría entre los dos sistemas móviles más populares? Sea como sea, los números de Android no dejan de sorprendernos.



.Fuente: planetared.com

B. ESTIMACIÓN DIARIA DE DISPOSITIVOS ANDROID EN EL MERCADO

Cada día se activan 1,3 millones de dispositivos con Android en el mundo



«Android, el sistema operativo de Google para dispositivos móviles continúa ganando adeptos. El presidente de la compañía, Eric Schmidt, informó que cada 24 horas se activan en el mundo hasta 1,3 millones de dispositivos con Android.

En diciembre del 2011, Android tenía una tasa de activación diaria de 700.000 terminales y en junio de 2012, la cifra llegó a los 900.000. Las cifras brindadas ahora por Schmidt no hacen más que demostrar su crecimiento.

El presidente de Google, quien estuvo presente anoche en un evento de Motorola (también propiedad del gigante de Internet), precisó que del total de activaciones diarias, la gran mayoría corresponden a smartphones, mientras que las tabletas avanzan más lentamente.

En concreto, hasta 1.230.000 teléfonos inteligentes se activan cada 24 horas con Android, es decir un 94,2%. Mientras que las tabletas acaparan el 5,8% (70.000 activaciones al día).

Según la firma Strategy Analytics, Android tiene el 68% del mercado global de smartphones, mientras que mientras Apple es dueño del 17%. Mientras que Windows ha capturado un 3,7%. Schmidt también anunció que hay hasta 480 millones de dispositivos Android en todo el mundo, precisó Silicon News».

Jueves 6 de septiembre del 2012

Fuente: elcomercio.pe

C. LA NSA PUBLICA SU VERSIÓN DE ANDROID



Esta versión del popular sistema operativo Android fue creado para darle mayor seguridad a los dispositivos móviles respecto a los código maliciosos.

La NSA publica una versión fortificada de Android

«Con objeto de limitar el ámbito de explotación de aplicaciones vulnerables y maliciosas en terminales Android, la Agencia de Seguridad Nacional de los Estados Unidos ha hecho publica una versión de Android con un gran número de mejoras de seguridad.

Dicha versión de Android, se basa en SELinux (Security-Enhanced Linux), conjunto de políticas de seguridad y controles de acceso implementados en el propio kernel del Sistema Operativo por medio de Linux Security Modules (LSM), y que también fue desarrollado por la NSA. SELinux aplica controles de acceso que proporcionan a los programas de usuario y servicios del sistema la mínima cantidad de privilegios necesarios para llevar a cabo sus funciones. Esto reduce o elimina la capacidad de estos programas de vulnerar el sistema al completo en el caso de ser comprometidos (por medio, por ejemplo, de desbordamientos de búfer o configuraciones incorrectas). Este sistema de seguridad funciona independientemente de los mecanismos tradicionales de control de acceso en los que se basa Linux.

Según se describe en la propia Wiki de proyecto, SE Android también se refiere a la implementación de referencia creada por el propio "SE Android project" destinada a proporcionar ejemplos prácticos de como habilitar y aplicar SELinux en las capas inferior de Android, además de mostrar su eficacia frente a exploits y aplicaciones vulnerables. El público objetivo, por tanto, al que está destinado este proyecto son desarrolladores de aplicaciones móviles, expertos en seguridad, fabricantes de dispositivos así usuarios avanzados de Android que deseen un mayor nivel de seguridad y privacidad en sus smartphones».

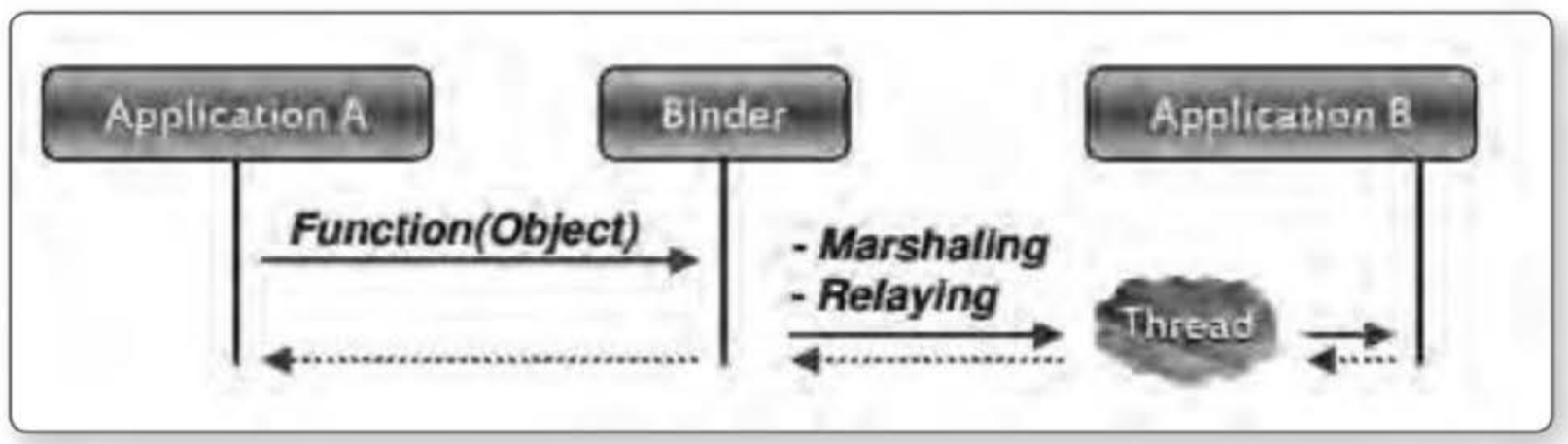
21 de enero de 2012

Fuente: seguinfo.wordpress.com

Algunas de las características de SE Android son las siguientes:

- **SISTEMAS DE FICHEROS YAFFS2 Y EXT4:**
  - Utilizado por muchos dispositivos Android.
  - Diseñado específicamente para las memorias Flash NAND.
  - Provee automáticamente el Wear-levelling (recolocación de datos).
  - Robusto ante fallos de energía.
  - Los datos de un fichero se almacenan por partes de 512 bytes, con un identificador.
  - Versiones:
    - YAFFS1 (tamaño máximo del sistema de ficheros: 1 GiB)
    - YAFFS2 (mejora de prestaciones. tamaño máximo del sistema de ficheros: 8 GiB)
- **CONTROL DE PERMISOS DEL KERNEL (BINDER IPC)**

Es una comunicación entre procesos (IPC). Permite que presten servicios a otros procesos a través de un conjunto de APIs de alto nivel que están disponibles en el estándar de Linux.



- **ETIQUETADO DE SOCKETS DE SERVICIO/FICHEROS CREADOS POR INIT**
- **ETIQUETADO DE NODOS DE DISPOSITIVOS CREADOS POR UEVENTD**
- **CONTROL DE PERMISOS DEL ESPACIO DE USUARIO (ZYGOTE SOCKET COMMANDS)**
- **SOPORTE SELINUX PARA ANDROID TOOLBOX**
- **USO DE CATEGORÍAS MLS PARA AISLAR APPST**

2.10. APLICACIONES

2.10.1. Google Play



Como se mencionó anteriormente, uno de los principales factores para determinar el éxito de una plataforma de Smartphone son las aplicaciones que lo soportan. Se desprende del éxito del iPhone que las aplicaciones de juego un papel muy importante en la determinación de si una nueva plataforma nada o se hunde. Además, como estas aplicaciones accesibles para el usuario general son muy importante.

En agosto de 2008, Google anunció Android Market, una tienda de aplicaciones online para los dispositivos Android, y puso a disposición de los usuarios unos meses después.

El uso de la aplicación en el mercado viene preinstalado en su dispositivo Android; por ende, los usuarios solo tienen que descargar las aplicaciones de terceras partes directamente en sus dispositivos. Aplicaciones tanto de pago y gratuitos se apoyan en el Android Market; pese a que las aplicaciones de pago están restringidas para los usuarios en ciertos países debido a problemas de legalidad. Del mismo modo, en algunos países, los usuarios pueden comprar aplicaciones de pago desde el Android Market, pero los promotores no pueden vender en ese país. A modo de ejemplo, en el momento de la escritura, los usuarios pueden comprar en la India aplicaciones del Android Market, pero los desarrolladores en la India no pueden vender aplicaciones en el Android Market. Todo lo contrario ocurre en Corea del Sur donde no se pueden comprar aplicaciones en el Android Market, pero los desarrolladores de dicho país pueden vender aplicaciones en ella.



Publicar una aplicación en el Android Market



Una vez que ya has realizado tu primera aplicación, lo normal es que crees una cuenta de desarrollador en el Android Market, y lo primero de todo, es pasar por taquilla.

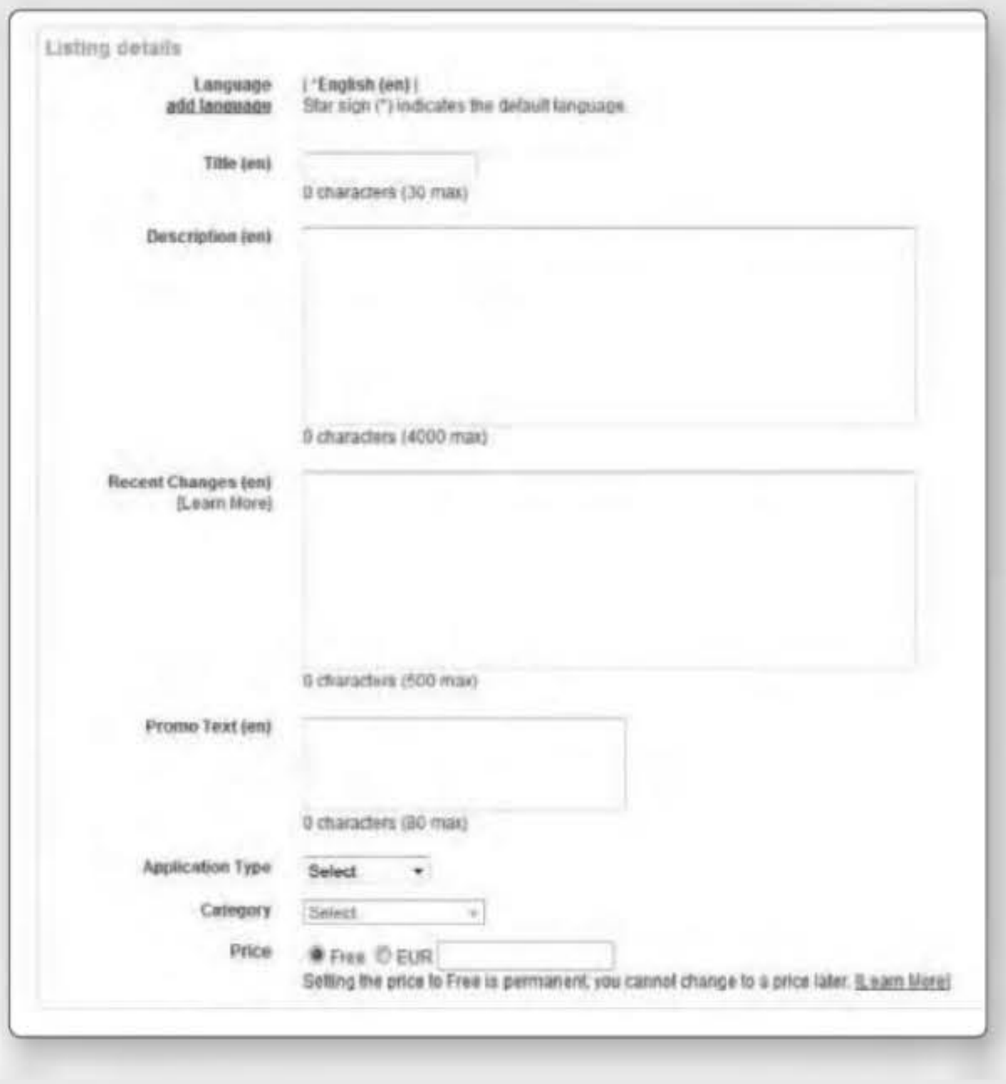
Crearte una cuenta en Google Checkout y abonar la tarifa de 25 $ para toda la vida (para desarrollar para Apple hay que abonar 100 $ al año). Después del pago ya tendrás acceso a las pantallas para subir aplicaciones.

El formulario, actualmente, se encuentra dividido en 3 secciones:

La parte de **"assets",** *donde se sube el .apk, las imágenes promocionales, un ícono (sic) enorme que ahora mismo no sirve para nada, un gráfico promocional, que es el más grande que se ve en la aplicación del Android Market y un video de youtube (sic).*



La parte de **"listing details"** *con la descripción de la aplicación en los diferentes idiomas.*



Y la parte de **"publishing options"** *con la protección y clasificación por edades, además de la información de contacto y aceptaciones correspondientes.*



Una vez publicada la aplicación, en breves minutos ya se encontrará en el Market, tan fácil y tan rápido y al de al menos un día comenzaremos a recibir feedback del número de descargas de nuestra aplicación, el número de descargas activas (estas son las buenas), así como de los comentarios y errores, pero esto será tema de otro artículo.

4 de febrero de 2011

Fuente: www.android.es

2.11. PAQUETES ANDROID



Android ofrece una serie de API para el desarrollo de sus aplicaciones.

A. SDK DE ANDROID

La siguiente lista proporciona una visión de lo que está disponible donde se da cuenta de los paquetes más importantes que se incluyen en el SDK de Android:

Contiene clases de recursos utilizados por las aplicaciones	
android.accessibilityservice	Las clases de este paquete se utilizan para el desarrollo de la accesibilidad del servicio que proporciona información alternativa o aumentada para el usuario.
android.accounts	
android.animation	Estas clases proporcionan la funcionalidad para el sistema de animación de la propiedad, que le permite animar las propiedades de objeto de cualquier tipo.
android.app	Contiene alto nivel de las clases que encapsula el modelo global de la aplicación Android.
android.app.admin	Proporciona las funciones del dispositivo de administración a nivel del sistema, lo que permite crear conciencia de seguridad de aplicaciones que son útiles en entornos empresariales, en el que los profesionales de TI requieren un control más ricos dispositivos de los empleados. Para más información, consulte la administración de dispositivos guía para desarrolladores.
android.app.backup	Contiene la funcionalidad de copia de seguridad y restauración disponibles en las aplicaciones. Si un usuario borra los datos en su dispositivo o actualización a un nuevo dispositivo con Android, todas las aplicaciones que han permitido copia de seguridad se puede restaurar los datos anteriores del usuario cuando la aplicación se vuelve a instalar. Para más información, consulte la copia de seguridad de datos de guía para el desarrollador
android.appwidget	Contiene los componentes necesarios para crear “widgets de aplicaciones”, que los usuarios pueden integrar en otras aplicaciones (como la pantalla de inicio) para acceder rápidamente a datos de aplicaciones y servicios sin necesidad de iniciar una nueva actividad. Para más información, consulte la aplicación Widgets guía para desarrolladores.
android.bluetooth	Proporciona clases que gestionan la funcionalidad de Bluetooth, tales como el escaneo de los dispositivos, la conexión con los dispositivos, y la gestión de transferencia de datos entre dispositivos. Para más información, consulte el Bluetooth guía para desarrolladores.
android.content	Contiene clases para el acceso y la publicación de datos en un dispositivo.
android.content.pm	Contiene clases para acceder a información sobre un paquete de aplicación, incluyendo información sobre sus actividades, permisos, servicios, firmas y proveedores.

android.content.res	Contiene clases para acceder a los recursos de aplicaciones, tales como archivos de activos primas, los colores, dibujables, los medios de comunicación u otros archivos de otros en el paquete, además de importantes detalles de la configuración del dispositivo (la orientación, tipos de entrada, etc) que afectan la aplicación puede comportar. Para más información, consulte la Aplicación de Recursos guía para desarrolladores.
android.database	Contiene clases para explorar los datos devueltos a través de un proveedor de contenido.
android.database.sqlite	Contiene las clases base de datos SQLite de gestión que una aplicación se utiliza para administrar su base de datos privada.
android.drm	Proporciona clases para la gestión de contenidos con DRM y la determinación de las capacidades de DRM plugins (los agentes).
android.gesture	Proporciona clases para crear, reconocer, cargar y guardar los gestos.
android.graphics	Ofrece gráficos de bajo nivel de herramientas tales como telas, filtros de color, puntos, rectángulos y que le permiten manejar dibujar en la pantalla directamente.
android.graphics.drawable	Ofrece clases para manejar una variedad de elementos visuales que están destinados solo para visualización, tales como mapas de bits y gradientes.
android.graphics.drawable.shapes	Contiene clases para dibujar formas geométricas.
android.hardware	Proporciona soporte para las características de hardware, tales como la cámara y otros sensores.
android.hardware.usb	Proporciona soporte para comunicarse con los periféricos USB que están conectados a los dispositivos con Android.
android.inputmethodservice	Las clases base para la escritura de los métodos de entrada (como teclados de software).
android.location	Contiene clases que definen los servicios de Android basados en la localización y afines. Para más información sobre los servicios de localización, consulte la documentación de obtención de ubicación de usuarios .
android.media	Proporciona clases que manejan las interfaces de diversos medios de comunicación en audio y video.
android.media.audiofx	Proporciona clases que lidiar con los efectos de audio a cabo en el marco de los medios de comunicación.
android.media.effect	Proporciona clases que permiten aplicar una variedad de efectos visuales a las imágenes y videos.
android.mtp	Proporciona APIs que le permiten interactuar directamente con las cámaras conectadas y otros dispositivos, utilizando el PTP (Picture Transfer Protocol) subconjunto de la MTP (Media Transfer Protocol) especificación.
android.net	Las clases que ayudan a acceder a la red, más allá de la normal .* API java.net.
android.net.http	
android.net.rtp	Proporciona APIs de RTP (Real-time Transport Protocol), permitiendo que las aplicaciones para gestionar on-demand o streaming de datos interactivos.
android.net.sip	Proporciona acceso a Session Initiation Protocol (SIP) funcionalidad, tales como hacer y recibir llamadas VoIP usando SIP.

android.net.wifi	Proporciona clases para gestionar Wi-Fi la funcionalidad del dispositivo.
android.net.wifi.p2p	Proporciona clases para crear peer-to-peer (P2P) con conexiones Wi-Fi Direct.
android.nfc	Proporciona acceso a la Comunicación de Campo Cercano (NFC) la funcionalidad, permitiendo que las aplicaciones para leer el mensaje NDEF de etiquetas NFC
android.nfc.tech	Estas clases proporcionan el acceso a las funciones de una tecnología de etiqueta, que varían según el tipo de etiqueta que se escanea.
android.opengl	Proporciona utilidades OpenGL
android.os	Proporciona los servicios básicos del sistema operativo, el paso de mensajes y comunicación entre procesos en el dispositivo.
android.os.storage	
android.preference	Proporciona clases que administran las preferencias de la aplicación y ejecutar la interfaz de usuario preferencias.
android.provider	Ofrece clases de conveniencia para acceder a los proveedores de contenido suministrado por Android.
android.renderscript	La prestación renderScript y APIs de cálculo ofrecen un bajo nivel, los medios de alto rendimiento para llevar a cabo cálculos matemáticos y la representación de gráficos en 3D.
android.sax	Un marco que hace que sea fácil de escribir controladores SAX eficiente y robusto.
android.security	
android.service.textservice	Proporciona clases que permiten crear los correctores ortográficos de una manera similar a la estructura de métodos de entrada (por IME).
android.service.wallpaper	
android.speech	
android.speech.tts	
android.telephony	Proporciona APIs para el control de la información básica del teléfono, tales como el tipo de red y estado de la conexión, además de utilidades para la manipulación de cadenas de número de teléfono.
android.telephony.cdma	Proporciona APIs para la utilización de CDMA características específicas de la telefonía.
android.telephony.gsm	Proporciona APIs para la utilización de las características específicas de GSM de telefonía, tales como texto / datos / mensajes SMS PDU.
android.test	Un marco para la escritura de casos de prueba de Android y suites.
android.test.mock	Clases de utilidad proporcionar recibos o se burla de los distintos bloques de construcción de marcos de Android.
android.test.suitebuilder	Clases de servicios públicos de apoyo a las clases corredor de prueba.
android.text	Proporciona clases que se utilizan para hacer o pista de texto y se extiende por el texto en la pantalla.
android.text.format	
android.text.method	Proporciona clases que controlar o modificar la entrada del teclado.

android.text.style	Proporciona clases que se utilizan para ver o cambiar el estilo de un fragmento de texto en un objeto View.
android.text.util	Utilidades para convertir cadenas de texto identificable en enlaces puede hacer clic y la creación de RFC 822-tipo de mensaje (SMTP) tokens.
android.util	Proporciona métodos comunes de servicios públicos tales como la manipulación de fecha / hora, base64 codificadores y decodificadores, número de cadenas y técnicas de conversión, y las utilidades de XML.
android.view	Proporciona clases que exponen las clases básicas de interfaz de usuario que se encargan de diseño de la pantalla y la interacción con el usuario.
android.view.accessibility	Las clases de este paquete se utilizan para representar el contenido de la pantalla y cambios en él, así como API para consultar el estado de la accesibilidad global del sistema.
android.view.animation	Proporciona clases que se encargan de animaciones interpoladas.
android.view.inputmethod	Clases de marco para la interacción entre los puntos de vista y métodos de entrada (como teclados virtuales).
android.view.textservice	
android.webkit	Proporciona herramientas para navegar por la web.
android.widget	El paquete contiene flash (en su mayoría visuales) elementos de la interfaz para usar en la pantalla de la aplicación.
dalvik.bytecode	Ofrece clases relacionadas con el código de bytes Dalvik.
dalvik.system	Ofrece clases de utilidad y sistema de información específico para la máquina virtual Dalvik.
java.awt.Font	
java.beans	
java.io	
java.lang	
java.lang.annotation	
java.lang.ref	Ofrece el sistema de ReferenceQueue aplicación, así como las diferentes formas de los objetos de referencia que imponen un comportamiento especial en el recolector de basura.
java.lang.reflect	
java.math	
java.net	
java.nio	
java.nio.channels	
java.nio.channels.spi	
java.nio.charset	
java.nio.charset.spi	
java.security	Este paquete es para compatibilidad con código anterior solamente, y no debe ser utilizada o para hacer algo útil.

java.security.acl	Este paquete proporciona las clases y las interfaces necesarias para crear listas de control de acceso.
java.security.cert	Este paquete proporciona todas las clases y todas las interfaces necesarias para generar, administrar y verificar los certificados X.509.
java.security.interfaces	Este paquete proporciona las interfaces necesarias para generar: (1) claves para el algoritmo de cifrado RSA asimétrica utilizando el estándar PKCS # 1, (2) Claves para el algoritmo de firma digital (DSA) especificado por FIPS-186, (3) Claves para una genérica de curva elíptica algoritmo de cifrado asimétrico.
java.security.spec	Este paquete proporciona las clases e interfaces necesarias para especificar las claves y parámetros de algoritmos de cifrado y firma.
java.sql	Proporciona una interfaz de compatibilidad para el acceso a bases de datos basadas en SQL.
java.text	
java.util	
java.util.concurrent	Utilidad general útil en las clases de programación concurrente.
java.util.concurrent.atomic	Un juego de herramientas pequeñas de clases que admiten sin bloqueo thread-safe de programación en las variables individuales.
java.util.concurrent.locks	Interfaces y clases que ofrecen un marco para el bloqueo y en espera de condiciones que es distinta de la construida en la sincronización y los monitores.
java.util.jar	
java.util.logging	
java.util.prefs	
java.util.regex	
java.util.zip	
javax.crypto	Este paquete proporciona las clases e interfaces para aplicaciones de cifrado para la aplicación de algoritmos de cifrado, descifrado, o acuerdo de la clave.
javax.crypto.interfaces	Este paquete proporciona las interfaces necesarias para implementar el Diffie-Hellman (DH) algoritmo de clave de acuerdo con lo especificado por el PKCS # 3.
javax.crypto.spec	Este paquete proporciona las clases e interfaces necesarias para especificar las claves y parámetros de codificación.
javax.microedition.khronos.egl	
javax.microedition.khronos.opengles	
javax.net	Este paquete proporciona clases de fábrica para crear enchufes y tomas de corriente del servidor.
javax.net.ssl	Este paquete proporciona todas las clases e interfaces necesarias para implementar el programa y la abstracción de sockets seguros basados en el protocolo SSL o SSSLv3.0 TLSv1.2.
javax.security.auth	

javax.security.auth.callback	Este paquete proporciona clases e interfaces necesarias para interactuar con la aplicación con el fin de ejecutar los procesos de autenticación y autorización.
javax.security.auth.login	Este paquete proporciona un sistema de autenticación conectable y apilable basada en ideas y conceptos desde el módulo de Unix-PAM.
javax.security.auth.x500	Este paquete proporciona las clases necesarias para almacenar X.500 directores y sus credenciales.
javax.security.cert	Este paquete solo se proporciona por razones de compatibilidad.
javax.sql	
javax.xml	Proporciona una clase de utilidad con utilidad constantes XML.
javax.xml.datatype	XML / Java tipo de asignaciones.
javax.xml.namespace	Espacio de nombres XML procesamiento.
javax.xml.parsers	Proporciona clases que permite el procesamiento de documentos XML.
javax.xml.transform	Este paquete define las API genéricas para el procesamiento de las instrucciones de la transformación, y llevar a cabo una transformación de la fuente al resultado.
javax.xml.transform.dom	Este paquete implementa DOM API específicas de transformación.
javax.xml.transform.sax	Este paquete implementa SAX2 API específicas de transformación.
javax.xml.transform.stream	Este paquete implementa la corriente y el URI de API específicas de transformación.
javax.xml.validation	Este paquete proporciona una API para la validación de documentos XML.
javax.xml.xpath	Este paquete proporciona un objeto-modelo neutral de la API para la evaluación de expresiones XPath y el acceso al entorno de evaluación.
junit.framework	El marco de pruebas JUnit.
junit.runner	Clases de utilidad apoyar el marco de pruebas JUnit.
org.apache.http	Las interfaces y las clases básicas de los componentes de HTTP.
org.apache.http.auth	El API para la autenticación de cliente HTTP en un servidor, comúnmente conocida como HttpAuth .
org.apache.http.auth.params	Parámetros para configurar HttpAuth .
org.apache.http.client	El API para la comunicación del lado del cliente HTTP y el punto de entrada a la HttpClient módulo.
org.apache.http.client.entity	
org.apache.http.client.methods	Implementaciones de la solicitud de los distintos métodos de HTTP como GET y POST.
org.apache.http.client.params	Parámetros para configurar HttpClient .
org.apache.http.client.protocol	Solicitar información adicional y la respuesta de los interceptores.
org.apache.http.client.utils	Ayudantes y clases de utilidad para HttpClient .
org.apache.http.conn	El cliente de conexión API de administración y manejo en el corazón de lo que se conoce como HttpConn .
org.apache.http.conn.params	Parámetros para configurar HttpConn .

org.apache.http.conn.routing	El lado del cliente vía la representación y la API de seguimiento, que forma parte de HttpConn .
org.apache.http.conn.scheme	
org.apache.http.conn.ssl	TLS / SSL partes específicas de la HttpConn API.
org.apache.http.conn.util	
org.apache.http.cookie	El API para la gestión de estado del lado del cliente a través de cookies, comúnmente conocida como HttpCookie .
org.apache.http.cookie.params	Parámetros para configurar HttpCookie .
org.apache.http.entity	Representaciones de las entidades mensaje HTTP.
org.apache.http.impl	Las implementaciones por defecto para las interfaces en org.apache.http .
org.apache.http.impl.auth	
org.apache.http.impl.client	
org.apache.http.impl.conn	
org.apache.http.impl.conn.tsccm	La implementación de un thread-safe administrador de conexión del cliente.
org.apache.http.impl.cookie	
org.apache.http.impl.entity	Las implementaciones por defecto para las interfaces en org.apache.http. entity .
org.apache.http.impl.io	Las implementaciones por defecto para las interfaces en org.apache.http.io .
org.apache.http.io	La abstracción de la capa de transporte de los componentes de HTTP.
org.apache.http.message	Una selección de HTTP mensaje de implementaciones
org.apache.http.params	El marco de la parametrización de los componentes de HTTP.
org.apache.http.protocol	El protocolo HTTP marco de la ejecución.
org.apache.http.util	Sobre todo las clases de utilidades con los métodos auxiliares estáticos para diversos fines.
org.json	
org.w3c.dom	Permite que el agente del W3C enlaces de Java para el Document Object Model, nivel 2 conductores.
org.w3c.dom.ls	
org.xml.sax	Este paquete proporciona el núcleo de APIs SAX.
org.xml.sax.ext	Este paquete contiene interfaces para SAX2 instalaciones conformes conductores SAX no necesariamente va a apoyar.
org.xml.sax.helpers	Este paquete contiene "ayudante" de las clases, incluido el apoyo para el arranque de aplicaciones basadas en SAX.
org.xmlpull.v1	
org.xmlpull.v1.sax2	

B. PAQUETES JAVA SOPORTADA POR ANDROID

java.io	java.text
java.lang	javax.xml.parsers
java.math	org.w3c.dom
java.net	org.xml.sax
java.nio	javax.sql
java.util	java.security
javax.crypto	java.sql
javax.net	javax.sound
javax.security	

C. PAQUETES NO SOPORTADO POR ANDROID

java.applet	javax.security.auth.kerberos
java.awt	javax.security.auth.spi
java.beans	javax.security.sasl
java.lang. management	javax.swing
java.rmi	javax.transaction
javax.accessibility	java.xml
javax.activity	org.ietf.*
javax.imageio	org.omg.*
javax.management	javax.print
javax.naming	javax.rmi

2.12. REQUERIMIENTO DE SOFTWARE

Las herramientas necesarias para comenzar a escribir su primera aplicación son las siguientes:

2.12.1. Java JDK



«Java 7 Development Kit (o JDK) es el kit de desarrollo oficial del lenguaje de programación Java, uno de los más populares entre los orientados a objetos. En la actualidad, Java 7 Development Kit es el entorno de desarrollo más popular para Java. No incluye IDE gráfico (como Eclipse), así que deberás instalarlo por separado» (Fuente: java-development-kit-jdk.softonic.com/).

El SDK de Anroid el kit de desarrollo hace uso del Java SE Development Kit (JDK). Por lo tanto, si su equipo no tiene el JDK instalado, usted debe comenzar por descargar el JDK de:

http://www.oracle.com/technetwork/java/javase/downloads/java-se-jdk-7-download-432154.html

Una vez descargado, instálelo.

2.12.2. ADT Bundle

Con una sola descarga, el paquete de ADT incluye todo lo necesario para comenzar a desarrollar las aplicaciones:

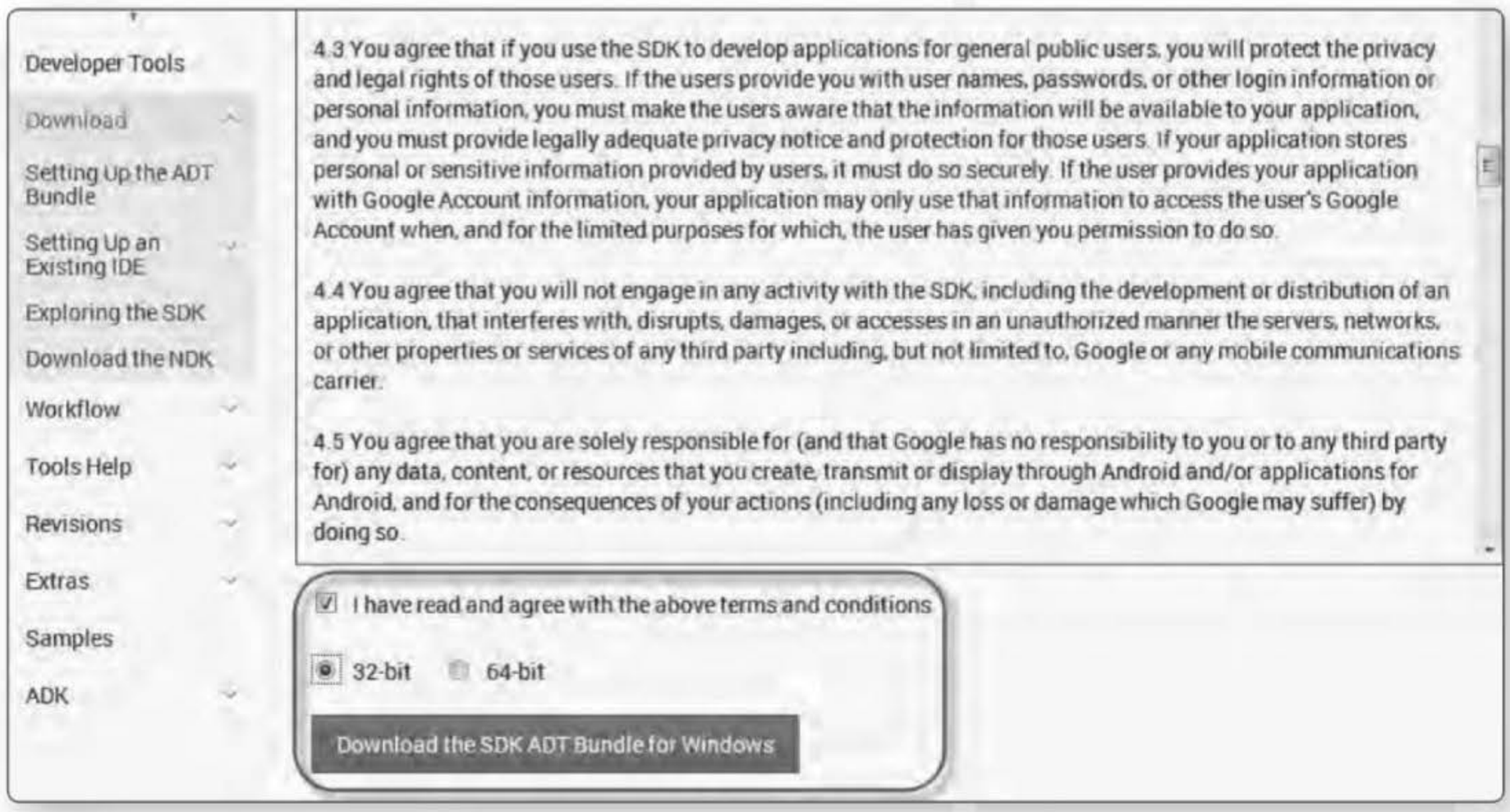
- ADT plugin de Eclipse +
- Android SDK herramientas
- Android Platform-tools
- La última plataforma Android
- La última imagen del sistema Android para el emulador

El sofware Developer Kit permite compilar aplicaciones en Android y emularlas. Para instalarlo, primero descárgue el SDK de Android que contiene un depurador, bibliotecas, un emulador, documentación, ejemplos de código y tutoriales (Fuente: ganaclin.blogspot.com).

Primero, descargue el SDK de Android del siguiente link:



Segundo, acepte las condiciones y luego descargar el SDK ADT Bundle for Windows:

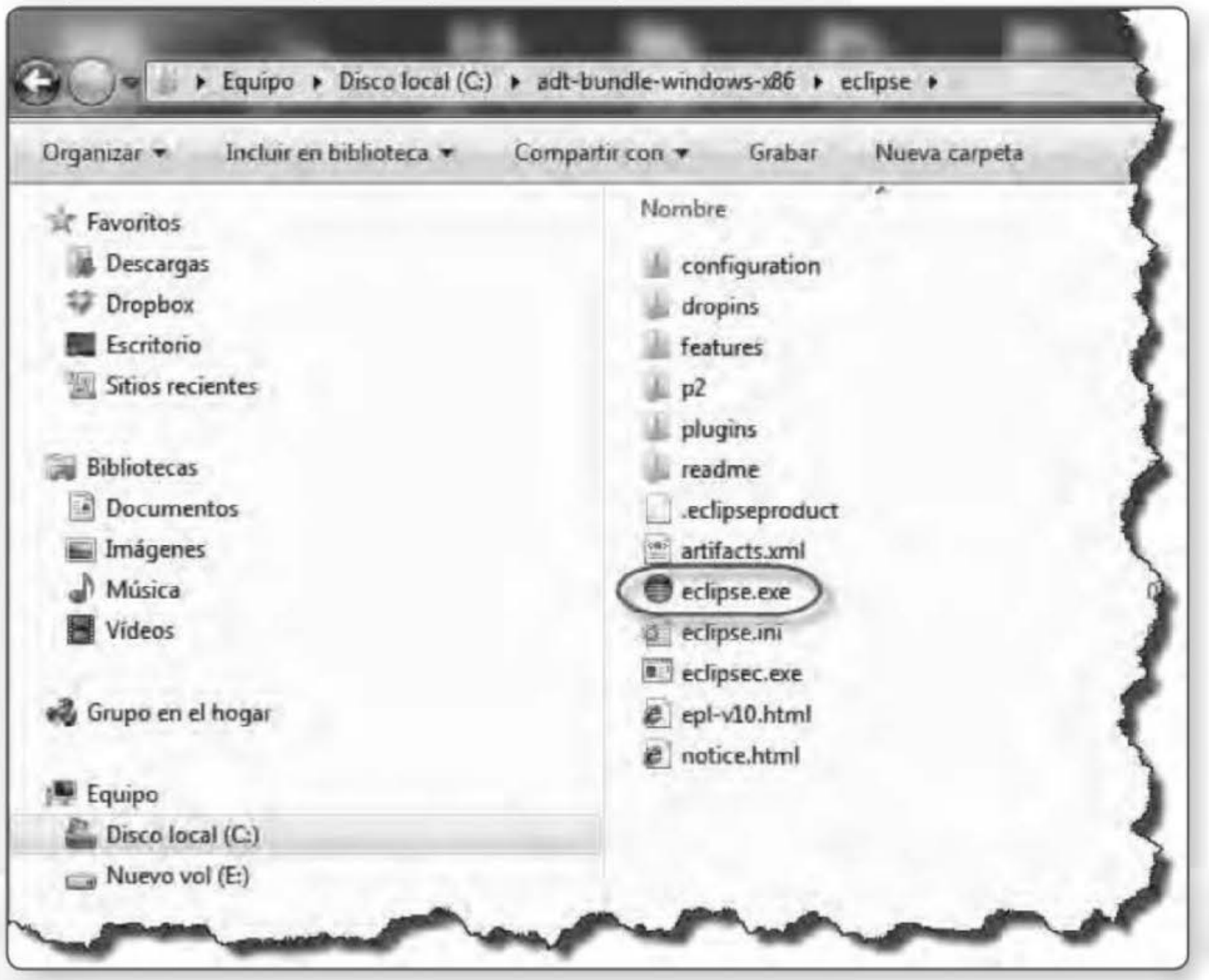


Nota:

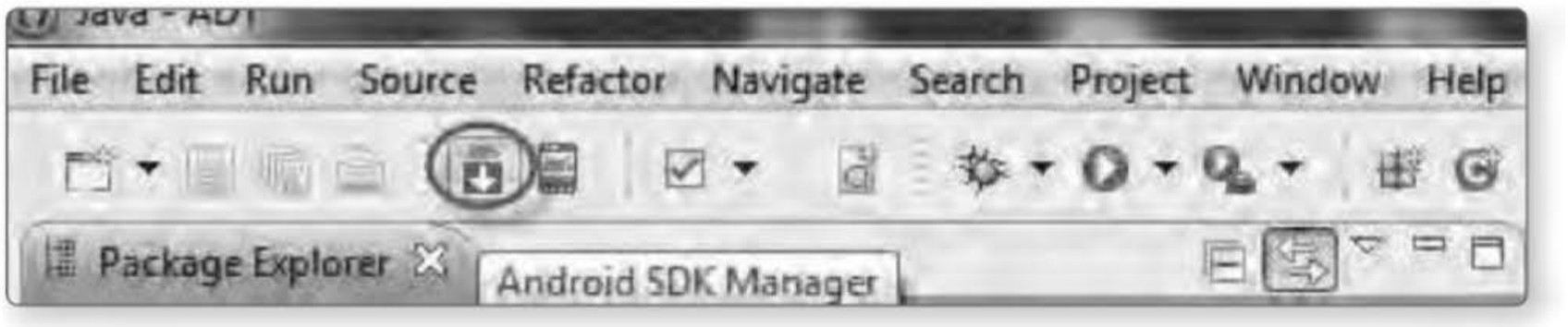
Se recomienda descargarlo en formato zip y guardarlo en un lugar diferente al archivo de programa, finalmente descomprímalo.

2.13. INSTALACIÓN DE LOS PAQUETES

Paso 1: Ubique el archivo Eclipse y haga doble clic para empezar.



Paso 2: Haga doble clic en el ícono Android SDK Manager.

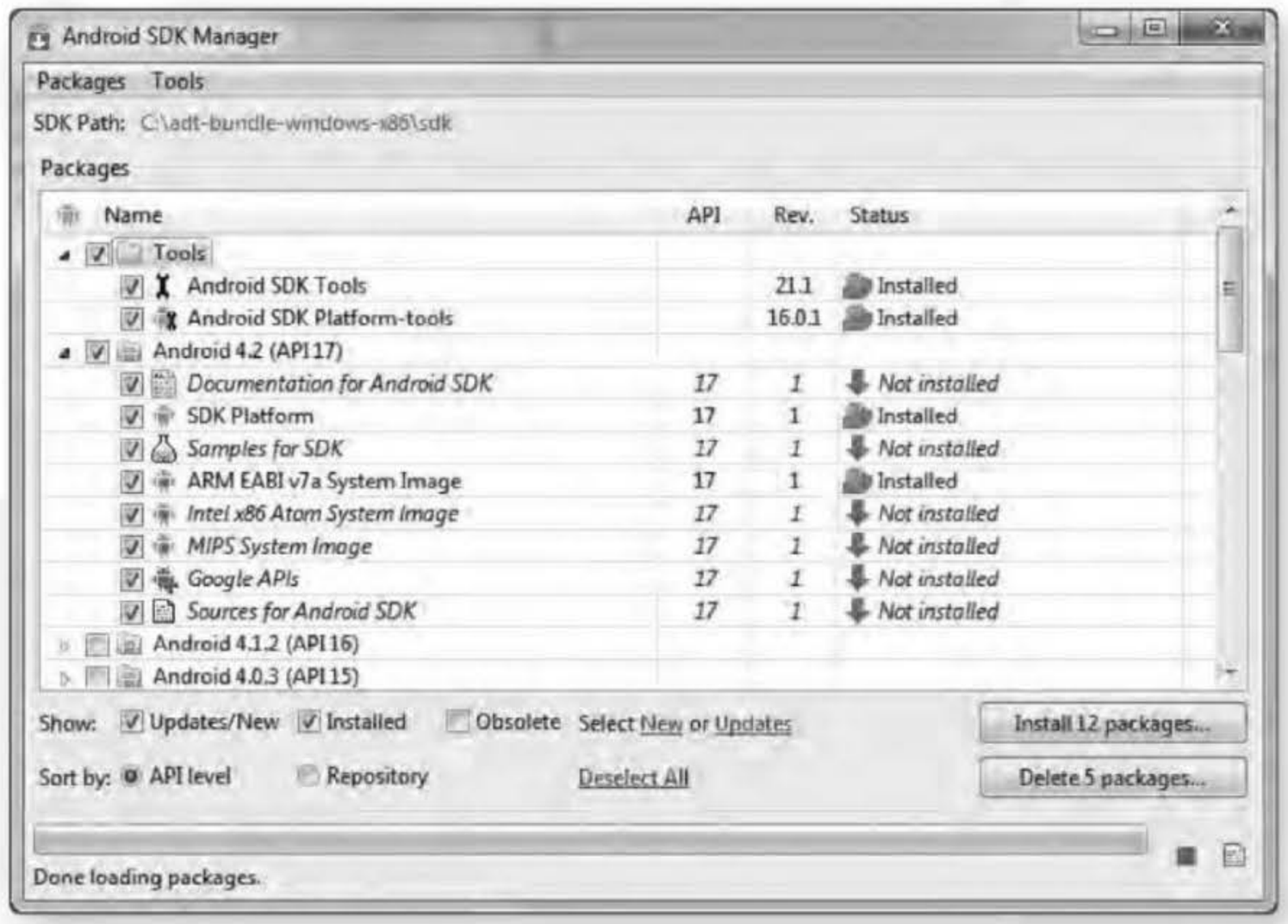


En primer lugar, compruebe la disponibilidad de los paquetes para la instalación. Estos contienen la documentación del SDK y específicos para cada versión del sistema operativo Android; también, contienen código de ejemplo y herramientas para las diversas plataformas.
La siguiente figura muestra los diversos paquetes SDK que se puede instalar en su computadora.

Paso 3: Haga doble clic en cada nombre de paquete para (des)activar un paquete si no está seguro de los paquetes de instalación. Es posible que desee seleccionar el botón *Aceptar* para descargar e instalar todos los paquetes.

Paso 4: Haga clic en *Instalar* para continuar con la descarga e instalación de los diferentes paquetes seleccionados. Cada versión del sistema operativo Android se identifica por un número de nivel API. Por ejemplo, Android 4.1 es el API 16. Descargar e instalar los paquetes lleva algún tiempo, así que tiene que esperar.

Paso 5: Cuando todos los paquetes estén instalados, haga clic en *Cerrar*. Ahora debería ver una lista de todos los paquetes instalados (ver fig.).



Paso 6: Haga clic en Instalar para continuar con la descarga e instalación de los diferentes paquetes seleccionados. Cada versión del sistema operativo Android se identifica por un número de nivel API. Por ejemplo, Android 4.2 es el API 17. Descargar e instalar los paquetes lleva algún tiempo, así que tiene que esperar.

Paso 7: Cuando todos los paquetes estén instalados, haga clic en Cerrar. Ahora debería ver una lista de todos los paquetes instalados.

2.14. ANDROID VIRTUAL DEVICES (AVD)



Los AVD (Android Virtual Devices) son unas herramientas imprescindibles para los desarrolladores y testers, ya que permiten emular en una computadora los dispositivos móviles a los que apunta una aplicación. Por defecto, cuando se instala el AVD Manager, no viene cargado ningún dispositivo virtual. En este apartado, se explicará cómo crearlo y así no solo testear su aplicación sino también probar y conocer Android desde su computadora.

2.14.1. Crear Android Virtual Devices

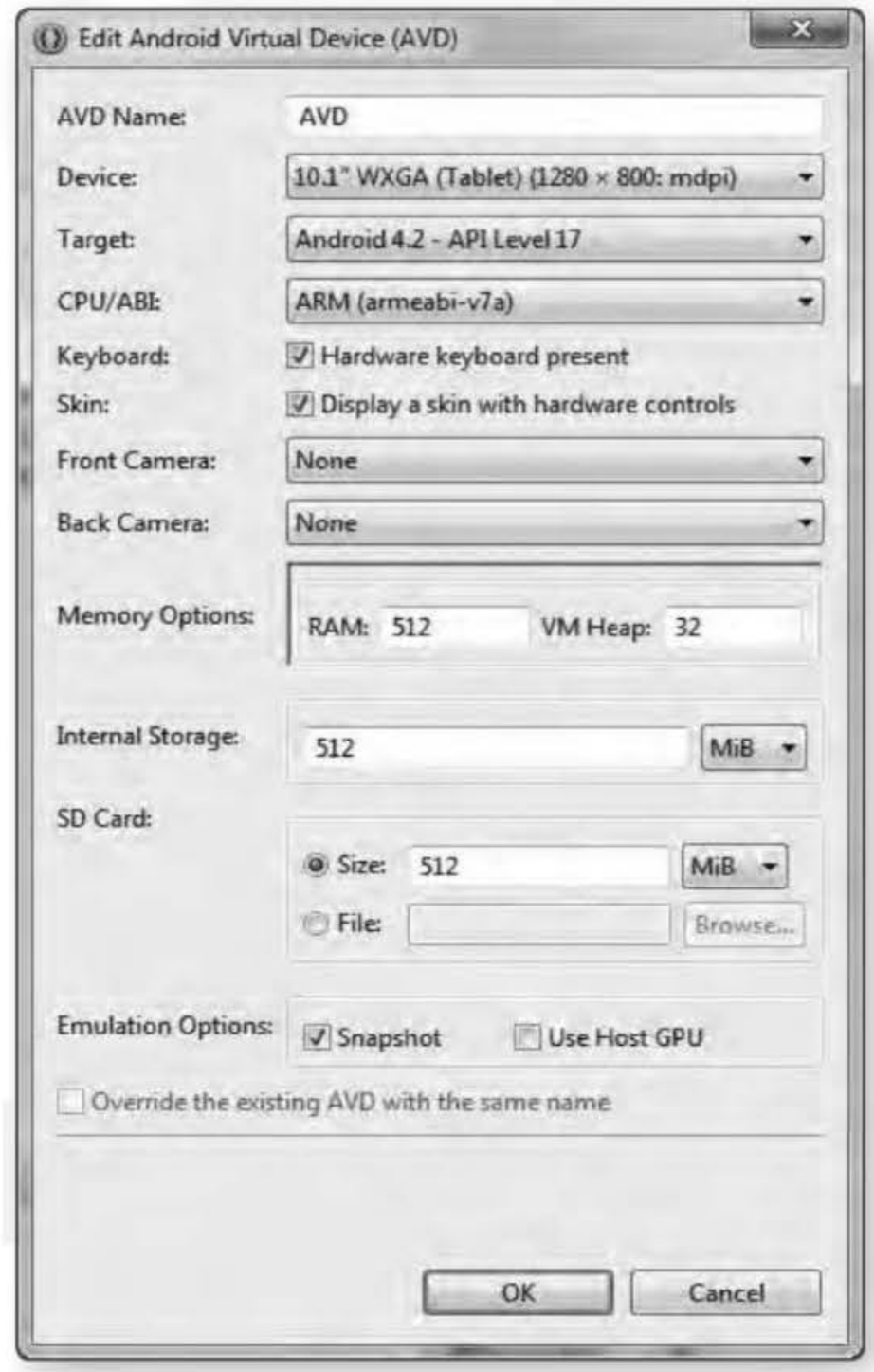


Una vez que los paquetes son descargados e instalados, el siguiente paso es crear un dispositivo virtual Android (AVD) que se utilizará para probar las aplicaciones para Android.

Un AVD es una instancia de emulador que permite modelar un dispositivo real. Cada AVD se compone de un perfil de hardware, una asignación a una imagen del sistema; así como el almacenamiento emulado, como una tarjeta Secure Digital (SD).

Puede crear AVD tanto como usted desea con el fin de probar sus aplicaciones con diferentes configuraciones. Esta prueba es importante para confirmar si su aplicación se comporta como se espera cuando se ejecuta en diferentes dispositivos con capacidades diferentes. Para crear una AVD, realice los siguientes pasos:

Paso 1: Seleccione la opción de dispositivos virtuales en el panel izquierdo de la ventana del SDK de Android y AVD Manager .



Paso 2: Haga clic en *Nuevo,* botón situado en el panel derecho de la ventana.

Paso 3: En *Crear nuevo dispositivo virtual de Android (AVD)*, introduzca los elementos como se muestra en la figura y haga clic en el botón *Crear AVD* cuando haya terminado. En este caso, se ha creado una AVD (es decir, un emulador de Android que emula un dispositivo Android corriendo la versión 4.1 del sistema operativo) y la opción de emular el dispositivo con una tarjeta SD de diferentes densidades de pantalla y resoluciones.

> ***Nota:***
> *Se explica cómo emular los diferentes tipos de dispositivos Android. Es preferible crear una AVD con niveles diferentes de la API para que su aplicación se puede probar en diferentes dispositivos.*

Paso 4: Proceda a arrancar el emulador seleccionando la AVD y luego presione el botón *Start*.



Paso 5: Observe cómo el emulador se pone en marcha.



El emulador es muy potente y completamente funcional. Por ejemplo, usted puede saltar fácilmente en línea, como se muestra en la figura y navegar por el sitio web. Puede configurar el correo, cuentas de correo, enviar mensajes SMS. Por supuesto, si usted es un desarrollador, podrá implementar y probar la aplicación.

2.14.2. El emulador



«El SDK de Android incluye un emulador de dispositivo móvil, que puede ser ejecuta en su computadora. El emulador que permite crear prototipos, desarrollar y probar aplicaciones Android sin necesidad de utilizar un dispositivo físico» (Fuente: android-des. blogspot.com/).

El emulador de Android imita todas las funciones típicas, excepto que no puede recibir ni realizar llamadas telefónicas. Como se muestra en la figura de la izquierda, el emulador ofrece navegación y las teclas de control, que puede acceder con el mouse o el teclado para generar eventos para su aplicación; también ofrece una pantalla en la que su aplicación junto con otras aplicaciones de Android están en funcionamiento. Para ayudar a modelar y probar la aplicación, así como utilizar los servicios de la plataforma Android para invocar otras aplicaciones, el acceso la red, reproducir audio y datos de video, almacenar y recuperar, notificar al usuario y hacer que las transiciones gráficas y temas. El emulador incluye una variedad de capacidades de depuración, una consola desde donde se puede registrar la salida del núcleo y simular la aplicación (por ejemplo, enviar mensajes SMS o llamadas telefónicas).

Puede utilizar las opciones de inicio emulador de la consola y los comandos para controlar los comportamientos y características de la emulación. Una vez que el emulador se ejecuta, se puede controlar mediante el uso del puntero del mouse o las teclas del teclado para acceder a las teclas del dispositivo simulado.

La siguiente tabla resume las correlaciones entre las teclas del emulador y las claves de su teclado.

Emulator Key	Keyboard Key
Back	ESC
Menu	F1 *or* PgUp
Star	F2 *or* PgDn
Call	F3
End Call	F4
—	F5, F6 unassigned
Power button	F7
Disable/enable all networking	F8
Start tracing	F9 (only with `-trace` flag)
Stop tracing	F10 (only with `-trace` flag)
Home	HOME
DPad left/up/right/down	Keypad 4/8/6/2
DPad center click	Keypad 5
Volume down	Keypad minus sign (-)
Volume up	Keypad plus sign (+)

A. RESOLUCIÓN DE EQUIPOS MOVILES EN PÍXELES

Puedes ejecutar el emulador con cualquiera de los cuatro aspectos predeterminados, como se describe en la tabla de abajo. Para especificar un <skinID> al iniciar el emulador.

2.7in QVGA
2.7in QVGA slider
3.2in HVGA slider (ADP1)
3.2in HVGA (ADP2)
3.3in WQVGA
3.4in FWQVGA
3.7in WVGA (Nexus One)
3.7in FWVGA slider
4in WVGA (Nexus S)
4.65in 720p (Galaxy Nexus)
4.7in WXGA
5.1in WVGA
5.4in FWVGA
7in WSVGA (Tablet)
10.1in WXGA (Tablet)

skinID	Description	Skin
QVGA-L	320x240, landscape (default)	
QVGA-P	240x320, portrait	
HVGA-L	480x320, landscape	
HVGA-P	320x480, portrait	

B. ENTORNO DEL EMULADOR

El SDK de Android incluye un emulador de dispositivo móvil que se ejecuta en una computadora. El emulador permite el prototipo y desarrollo, así como probar aplicaciones Android sin necesidad de utilizar un dispositivo físico (por la experiencia del uso del emulador no todas las aplicaciones desarrolladas utilizan esta herramienta que tiene un 100% de éxito probado). El emulador de Android imita todas las características de hardware y software de un dispositivo móvil típico, excepto que no puede recibir ni realizar real llamadas de teléfono (funciones básicas de un teléfono). Proporciona una variedad de teclas de navegación y control.



El emulador de Android soporta muchas características de hardware:

- ARMv5 CPU y una unidad de administración de memoria (MMU)
- 16-bit LCD display
- Uno o más teclados (un teclado Qwerty y otro basado en Dpad/Phone buttons)
- Un chip de sonido con capacidad de entrada y salida
- Particones de memoria Flash (emulado a través de los archivos de imagen de disco en el equipo de desarrollo)
- GSM modem, incluyendo un simulador de tarjeta SIM
- Una cámara (usando una cámara web conectada a su equipo de desarrollo)
- Sensores como un acelerómetro (utilizando datos de un dispositivo conectado por USB Android)

C. TRABAJAR CON IMÁGENES DEL DISCO DEL EMULADOR

El emulador utiliza imágenes montables disco (imagen Android System) almacenado en su máquina de desarrollo para simular Flash, particiones en un dispositivo real. Por ejemplo, utiliza imágenes de disco que contiene lo siguiente:

- Un núcleo emulador específico
- El sistema Android
- Una imagen de disco RAM
- Imágenes grabables para datos de usuario y simulador de tarjeta SD



Por defecto, el emulador siempre busca las imágenes de disco en el área de almacenamiento privado del AVD en uso:

(c: \ android-sdk-windows \ platform \ ...)

Nota:

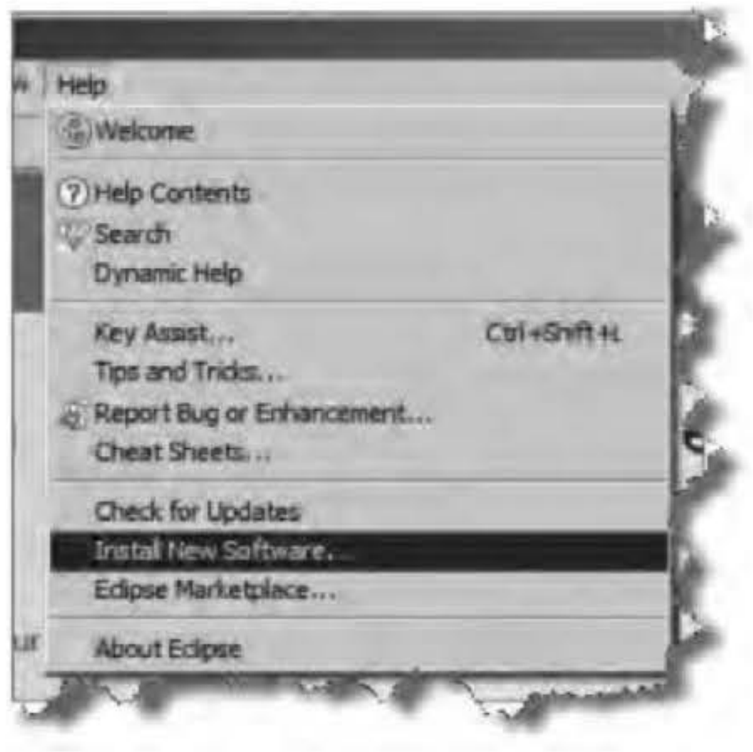
Si no hay imágenes de plataforma, existen cuando el emulador se inicia, se crea las imágenes en el directorio AVD (basado en versiones por defecto almacenados en el SDK).



Directorio AVD de la versión con ausencia de imagen (Version 4.1 – Api 16)



2.14.3. Herramientas de desarrollo de Android (ADT)



Con el SDK de Android y AVD establecido, es momento de configurar Eclipse para reconocer la plantilla del proyecto Android. La herramientas de desarrollo de Android (ADT) plug-in para Eclipse es una extensión de la Eclipse IDE que soporta la creación y depuración de aplicaciones de Android. Uso de la ADT, usted será capaz de hacer lo siguiente en Eclipse:

- Crear nuevos proyectos de aplicaciones de Android.
- Acceder a las herramientas para acceder a sus emuladores y dispositivos Android.
- Compilar y depurar aplicaciones de Android
- Exportar aplicaciones Android en paquetes de Android (APK).
- Crear certificados digitales para la firma de código de su APK.

Para instalar el ADT, Eclipse primer lanzamiento

Paso 1: Haga doble clic en el archivo eclipse.exe que se encuentra en la carpeta Eclipse. Cuando Eclipse empieza, le pedirá una carpeta para su uso como espacio de trabajo. En Eclipse, un espacio de trabajo es una carpeta donde guarda todos sus proyectos.

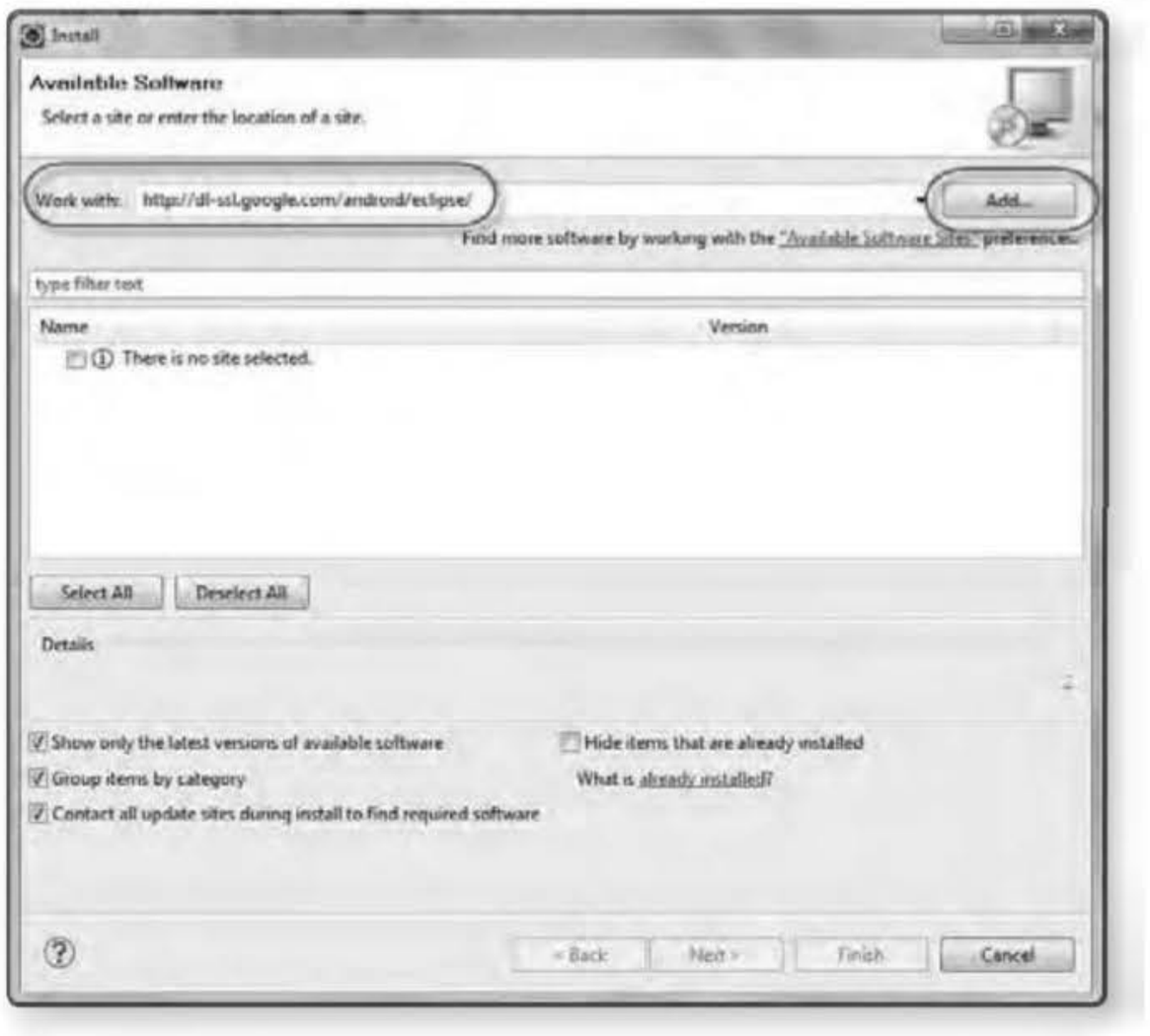
Paso 2: Tome la sugerencia por defecto y haga clic en *Aceptar*.

Paso 3: Una vez que Eclipse está en marcha, seleccione el software *Ayuda* o *Instalar nuevo...*

Paso 4: En la ventana de instalación que aparece, escriba lo siguiente:

http://dl-ssl.google.com/android/eclipse

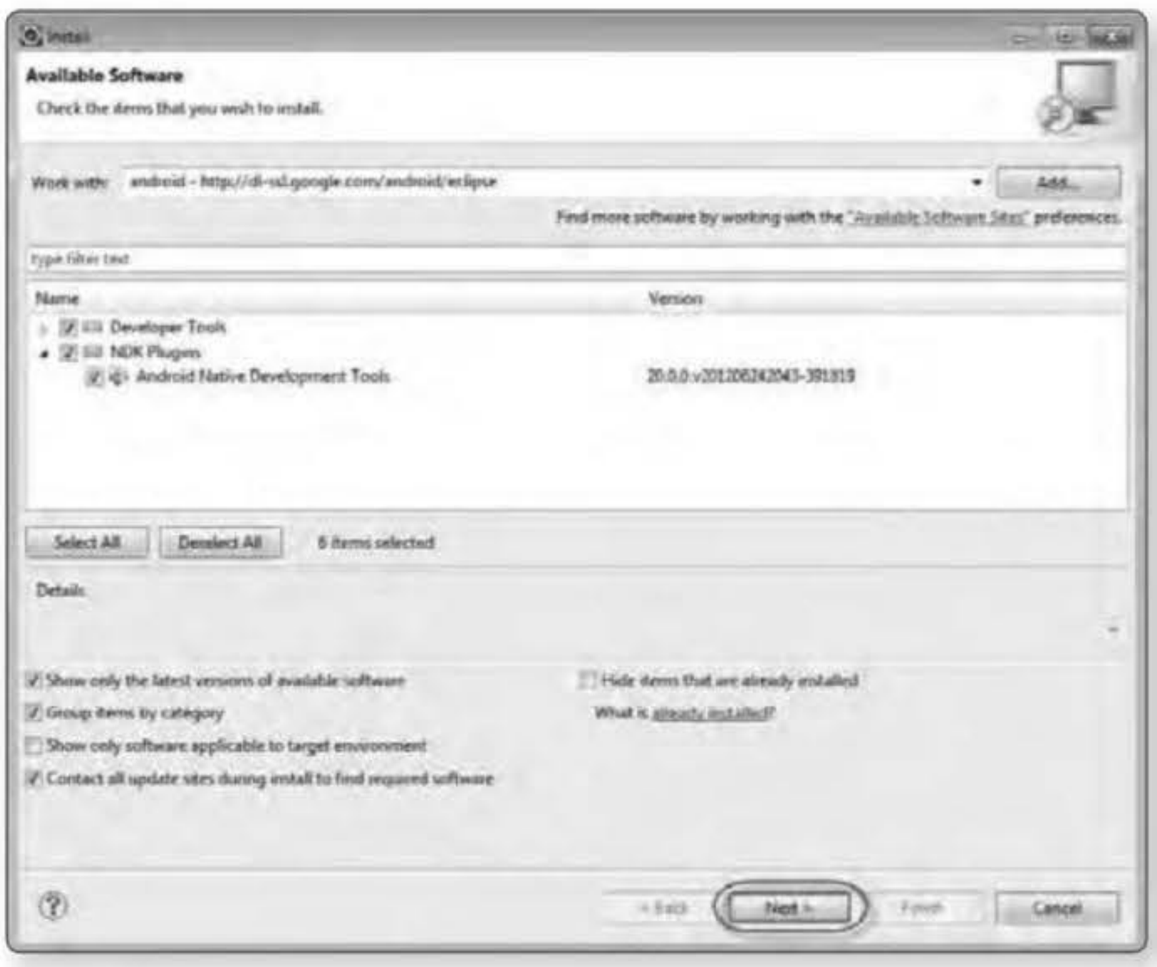
Paso 5: En el cuadro de texto superior, pulse Add (ver fig.).



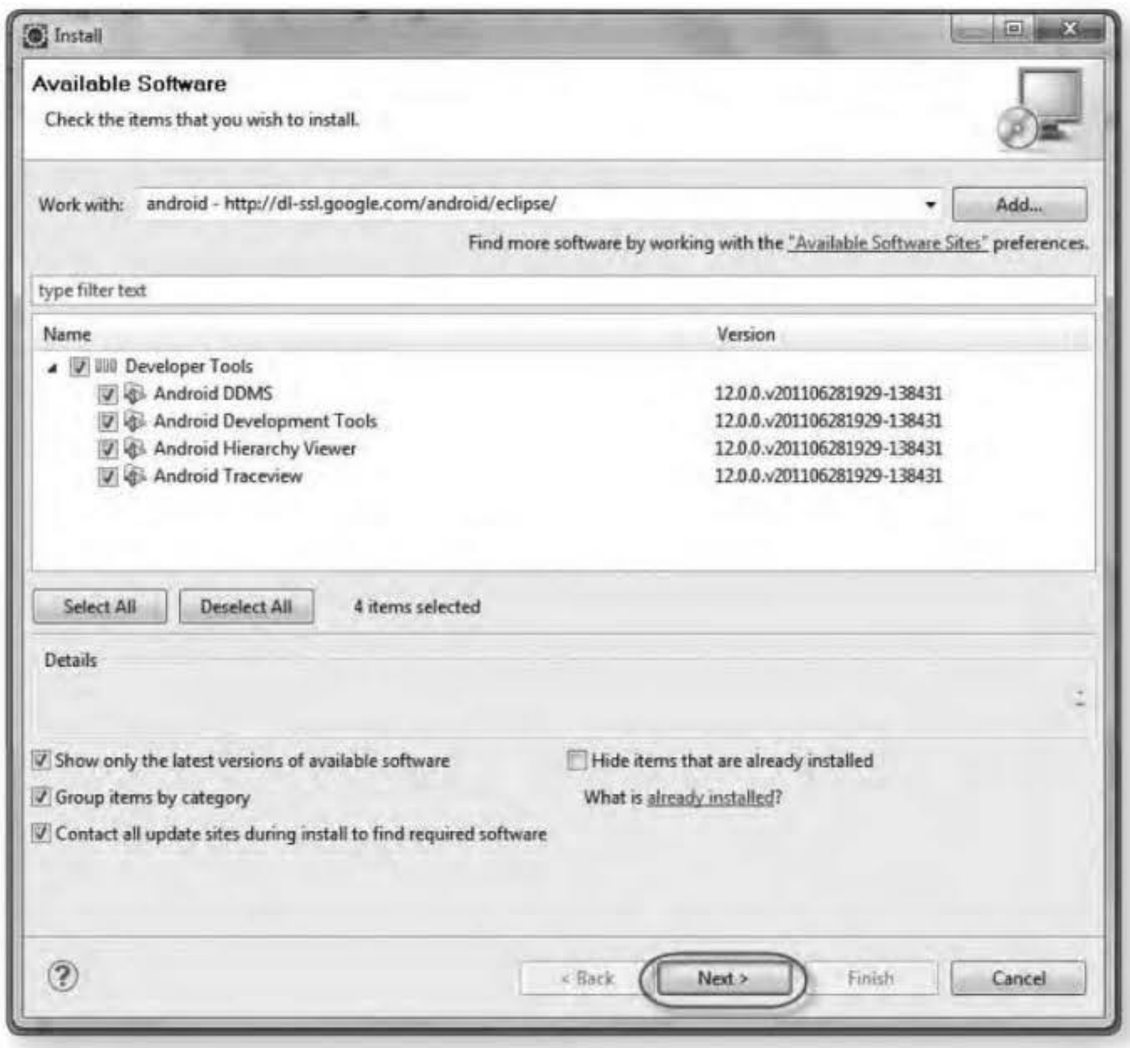
Paso 6: Digite lo siguiente:



Paso 7: Active la casilla *Developer Tools* y opcionalmente FALTA NDK.



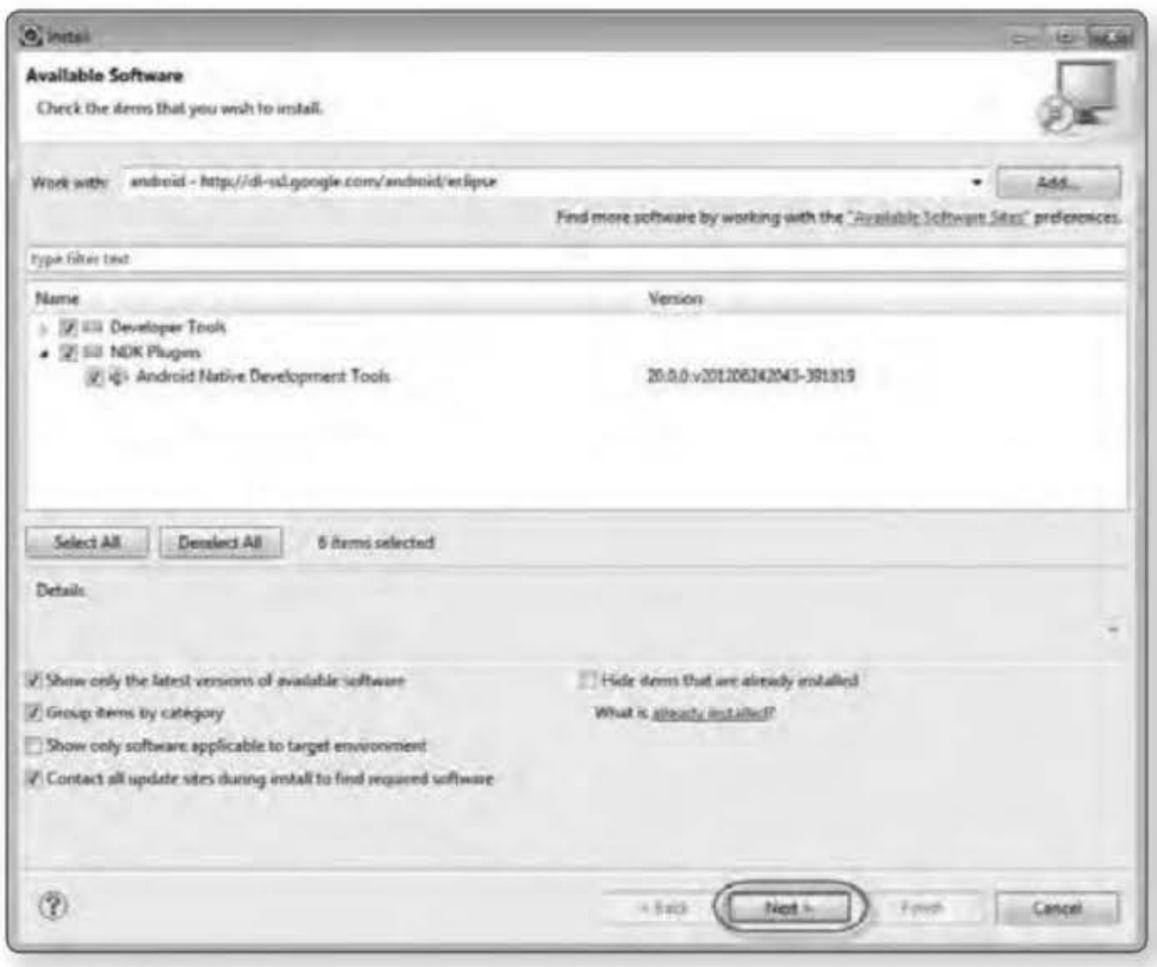
Paso 8: Presione el botón Siguiente (o Next).



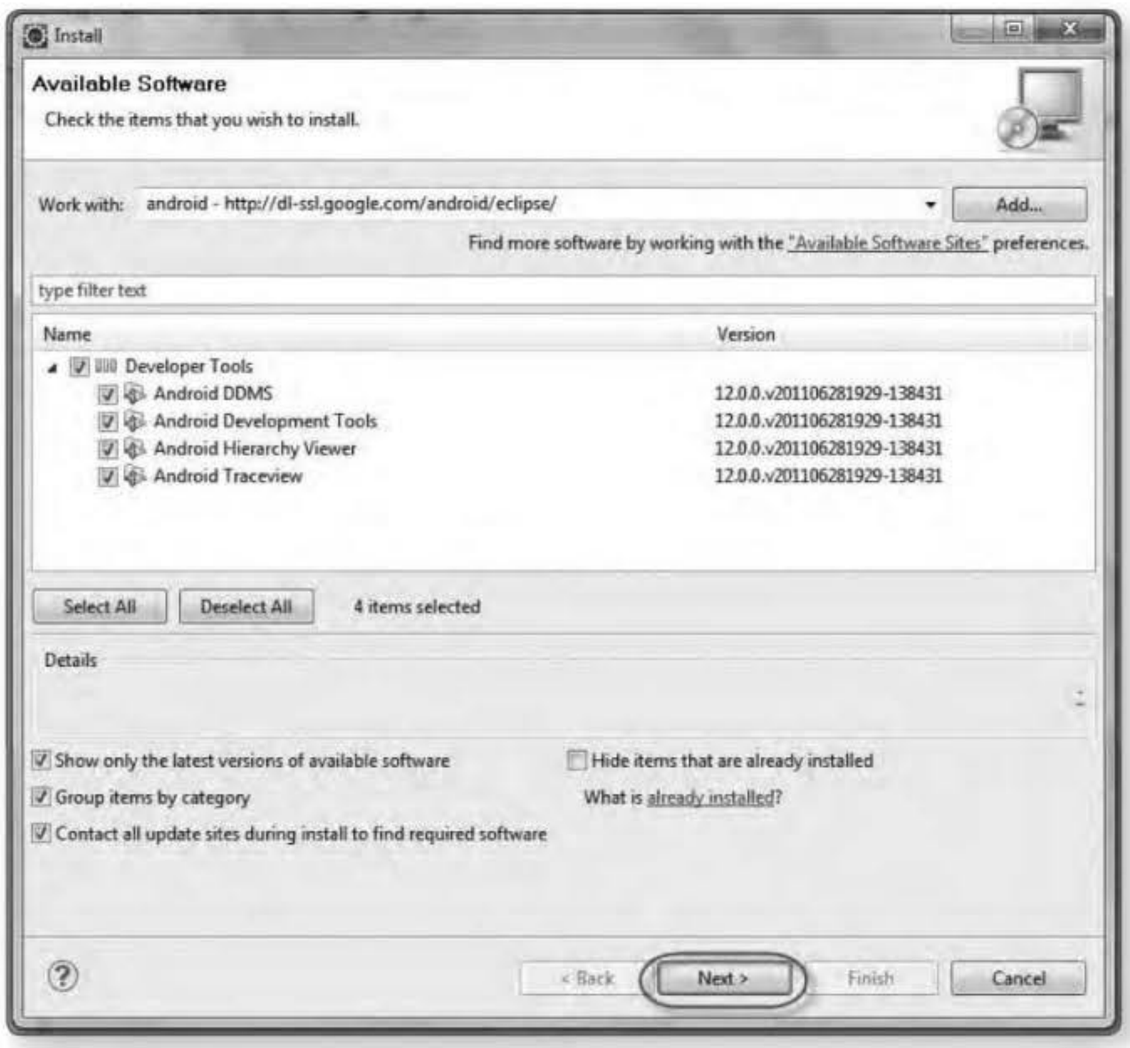
Paso 6: Digite lo siguiente:



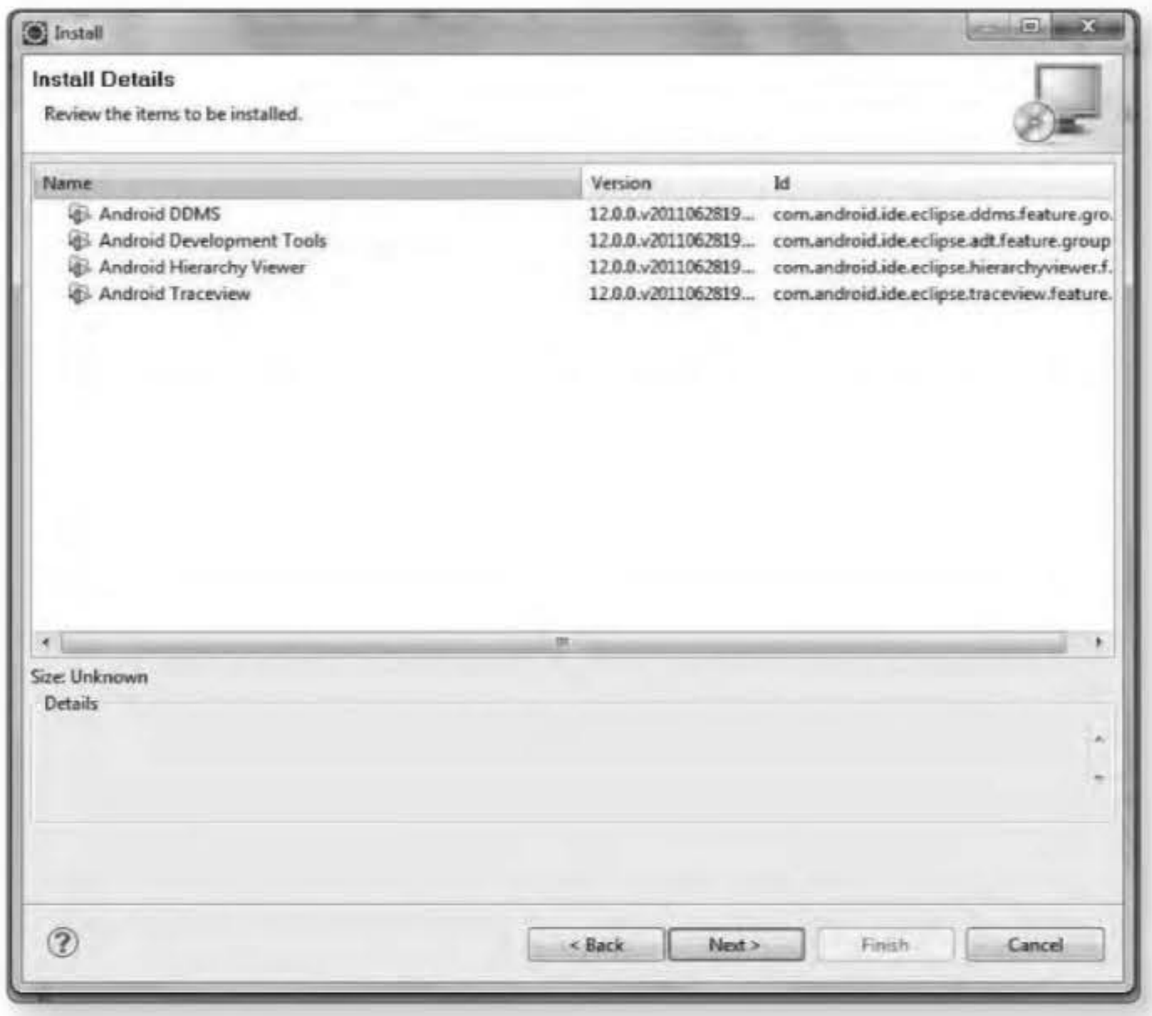
Paso 7: Active la casilla *Developer Tools* y opcionalmente FALTA NDK.



Paso 8: Presione el botón Siguiente (o Next).



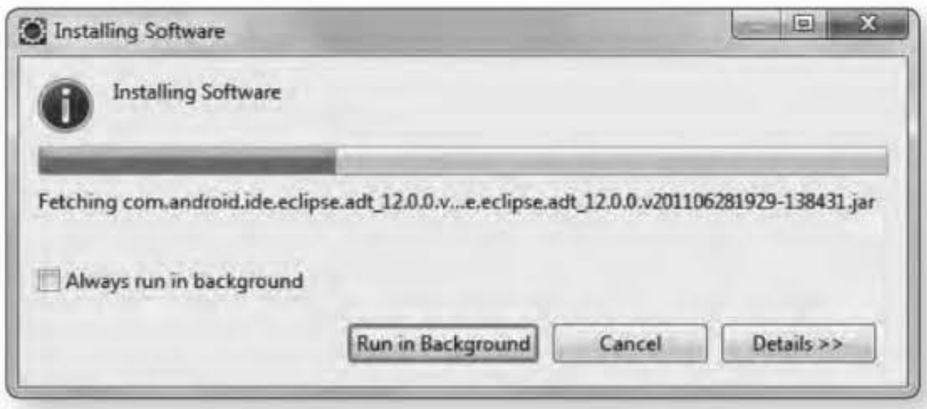
Paso 9: Cuando vea la ventana *Detalles de instalación* (ver fig.), haga clic en *Siguiente* (o *Next*).



Paso 10: Se le pedirá la revisión de las licencias de las herramientas. Para ello, marque la casilla de opción *Aceptar los acuerdos de licencia* (ver fig.) y haga clic en *Finalizar* para continuar.





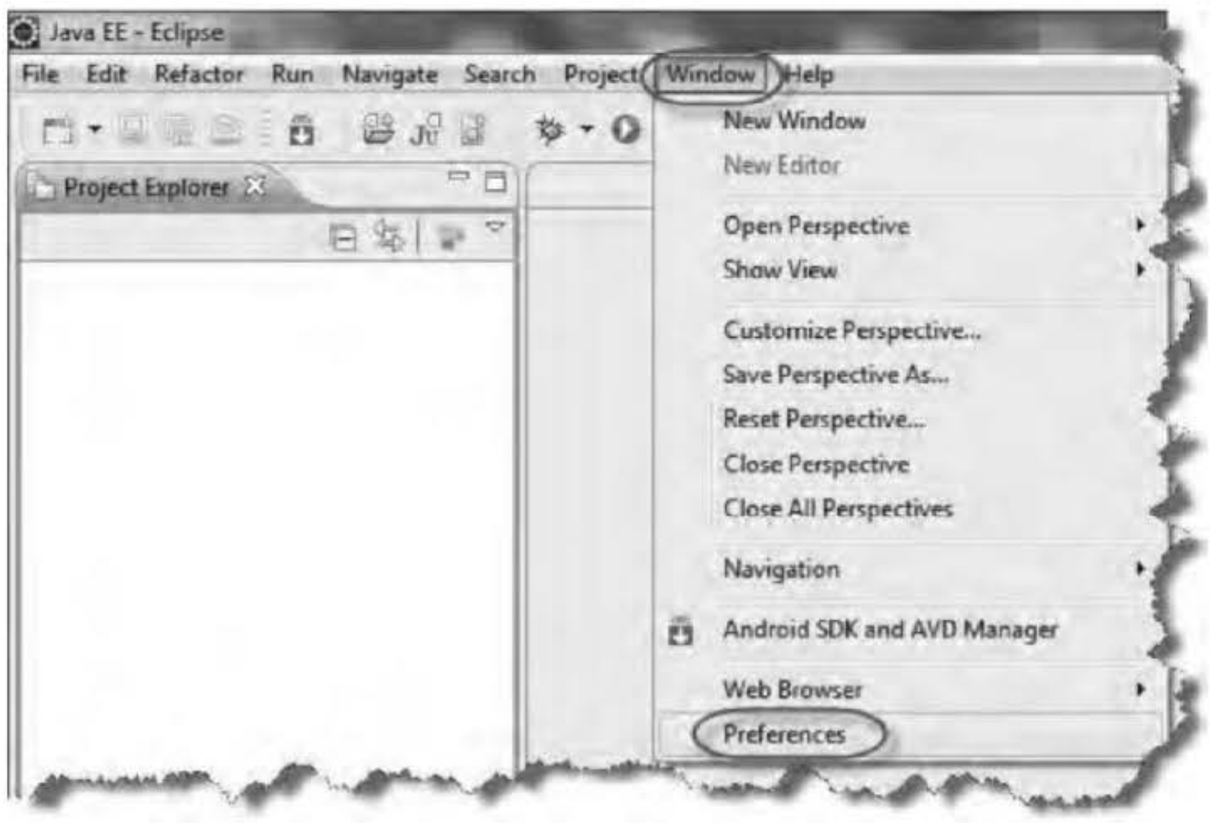


Paso 11: Resetee Eclipse para aplicar satisfactoriamente los cambios.

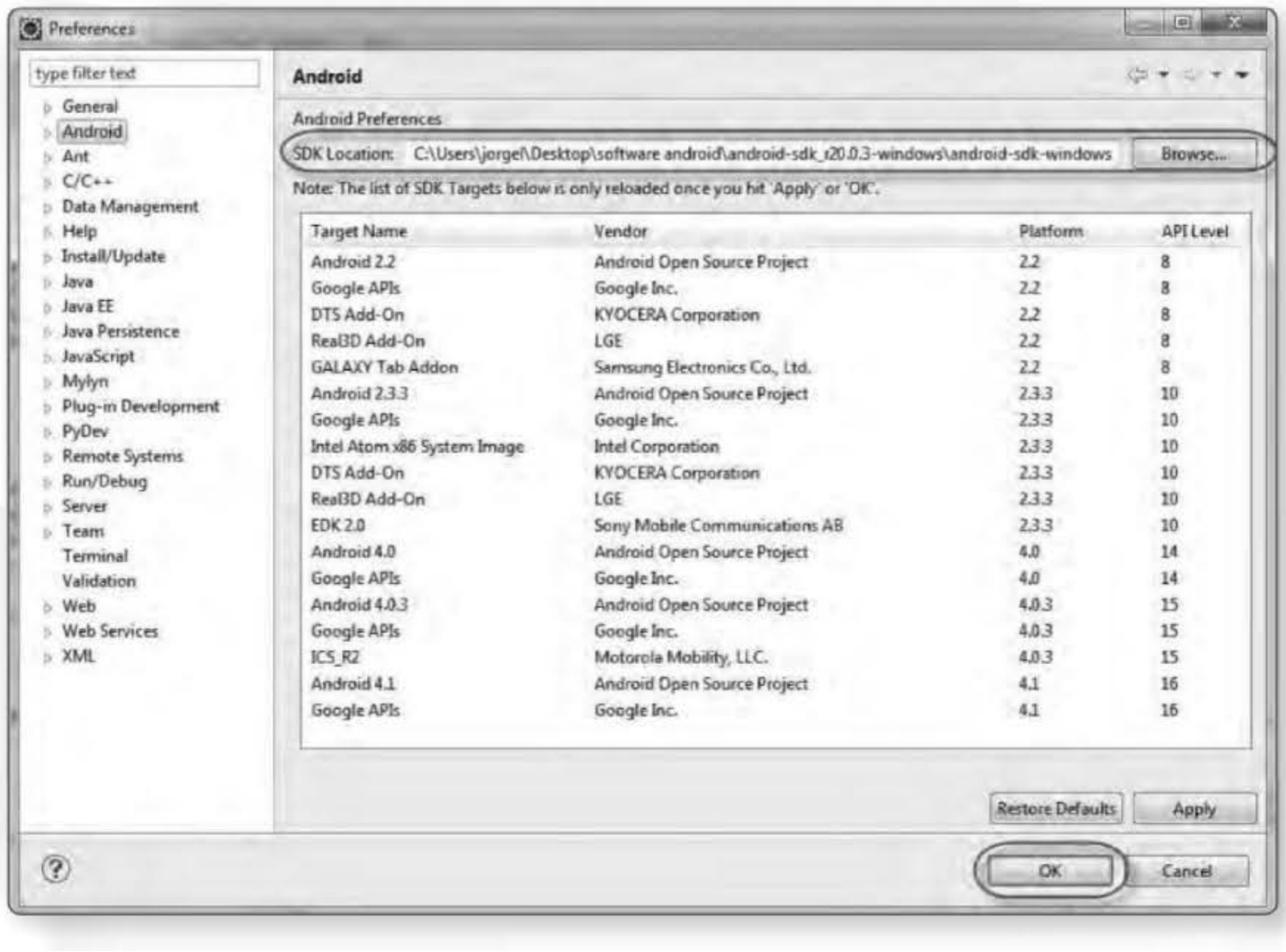


2.14.4. Configurar Eclipse para acceder al SDK

Una vez reiniciado, configure Eclipse para que acceda al SDK de Android ya instalado. Abra el menú Window y seleccione *Preferences*.



En el apartado Android, en SDK Location, introduzca la ruta donde se ha descomprimido el SDK de Android.



Finalmente, se obtiene el SDK de Android, integrado en Eclipse para empezar a programar.

2.15. HERRAMIENTA ALTERNATIVA

Puede utilizar en forma aternativa Motodev Studio for Android (developer.motorola.com) que se define como un entorno gratuito dado por Motorola, basado en Eclipse en el cual se incluye el ADT necesario para el desarrollo en Android (Fuente: www.puntogeek.com).

2.15.1. NDK (Native Development)

Conjunto de herramientas para compilar componentes nativos y poder utilizarlos en aplicaciones Android, permite utilizar librerías C, C++.

¿Qué es el NDK de Android? (traducción)

«El NDK de Android es un conjunto de herramientas que permite incorporar los componentes que hacen uso de código nativo en las aplicaciones de Android.

Las aplicaciones para Android se ejecutan en la máquina virtual Dalvik. El NDK permite implementar parte de tus aplicaciones usando código nativo con lenguajes como C y C + +. Esto puede proporcionar beneficios a ciertas clases de aplicaciones, en la medida que se puede reutilizar el código existente y en algunos casos obtener un aumento de la velocidad.

El NDK establece lo siguiente:

- Un conjunto de herramientas y archivos de construcción que se utilizan para generar bibliotecas de código nativo de fuentes en C y C + +.
- Una manera de integrar las correspondientes bibliotecas nativas en un archivo de paquete de aplicaciones (. Apk) que se pueden implementar en los dispositivos Android.
- Un conjunto de cabeceras y bibliotecas nativas del sistema que estarán soportadas en todas las futuras versiones de la plataforma Android, a partir de Android 1.5. Las aplicaciones que utilizan actividades nativas deben de ejecutarse sobre Android 2.3 o posterior.
- Documentación, ejemplos y tutoriales.

La última versión de la NDK soporta estos conjuntos de instrucciones ARM:

- ARMv5TE (incluyendo instrucciones Thumb-1)
- ARMv7-A (incluyendo instrucciones Thumb-2 y VFPv3-D16, con soporte opcional para instrucciones NEON/VFPv3-D32)

Las futuras versiones de la NDK también prestarán apoyo a:

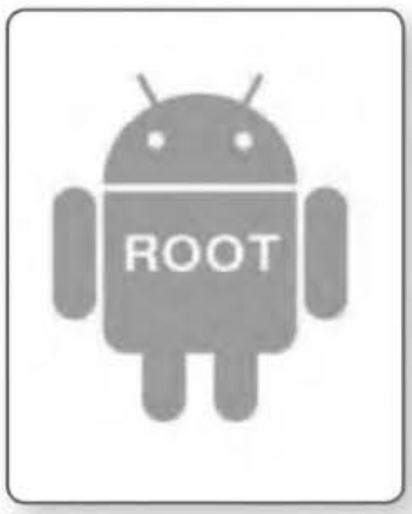
- Instrucciones x86 (ver CPU-ARCH-ABIS.HTML para más información).

*El código máquina ARMv5TE funcionará en todos los dispositivos ARM-Based de Android. ARMv7-A sólo se ejecutarán en los dispositivos tales como el Droid de Verizon o Google Nexus One que tengan una CPU compatible. La principal diferencia entre los dos conjuntos de instrucciones es que ARMv7-A soporta instrucciones hardware FPU, Thumb-2, y NEON. Usted puede apuntar (Target) a uno o ambos conjuntos de instrucciones – ARMv5TE es el valor predeterminado, pero cambiar a ARMv7-A es tan fácil como añadir una línea al archivo de aplicación Application.mk , sin necesidad de cambiar nada mas en el archivo. También se puede construir para ambas arquitecturas al mismo tiempo y tenerlo todo almacenado en el *.Apk final. Toda la información se encontrará en el archivo CPU-ARCH-ABIS.HTML del paquete NDK.*

El NDK proporciona las cabeceras estables para libc (la biblioteca de C), libm (la biblioteca matemática), OpenGL ES (3D Graphics Library), la interfaz JNI, y otras bibliotecas, como se indica en la sección de herramientas de desarrollo».

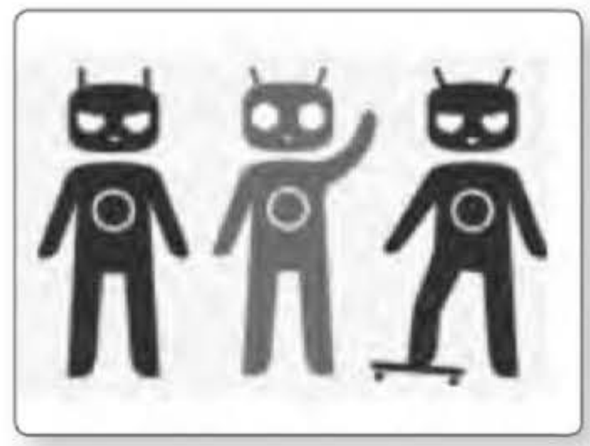
Fuente: zsoluciones.com

2.16. ROOTEAR UN DISPOSITIVO MÓVIL



En el mundo Android se reconocen conceptos como «root» de los dipositivos móviles. Rootear un teléfono significa conseguir acceso de la root (el "superusuario" puede hacer lo que quiera en el sistema, cambiar configuraciones, borrar archivos protegidos, etc.) a la misma, ya sea oficialmente durante el uso de los teléfonos de desarrollo o el uso de hacks con un teléfono del usuario final. El interés principal es actualizar el sistema antes de que el fabricante proporcione actualizaciones (si los hay) o utilizar una versión personalizada (optimizado modificados, por ejemplo, CyanogenMod). También se puede hacer manipulaciones que un administrador realiza (por ejemplo, la implementación de un kernel personalizado). El rootaer no es una operación ilegal; ya que está modificando su dispositivo, esto puede tener un efecto negativo en la pérdida de la garantía de parte de los fabricantes.

2.16.1. CyanogenMod



CyanogenMod es un firmware no oficial, disponible para más de 80 teléfonos móviles y tablets, basados en el sistema operativo de código abierto Android. Ofrece características que no se encuentran en las versiones oficiales, basadas en Android suministradas por los fabricantes, incluyendo, entre otras mejoras:

- Soporte para Free Lossless Audio Codec (FLAC)
- Multitouch
- Caché comprimida (compcache)
- Una larga lista de APN
- Menú de reinicio
- Soporte para tethering mediante Wi-Fi, Bluetooth o USB

CyanogenMod asegura que sus modificaciones mejoren el rendimiento y la fiabilidad frente a las versiones oficiales del software.



Mi primera aplicación

3.1. CREAR LA PRIMERA APLICACIÓN

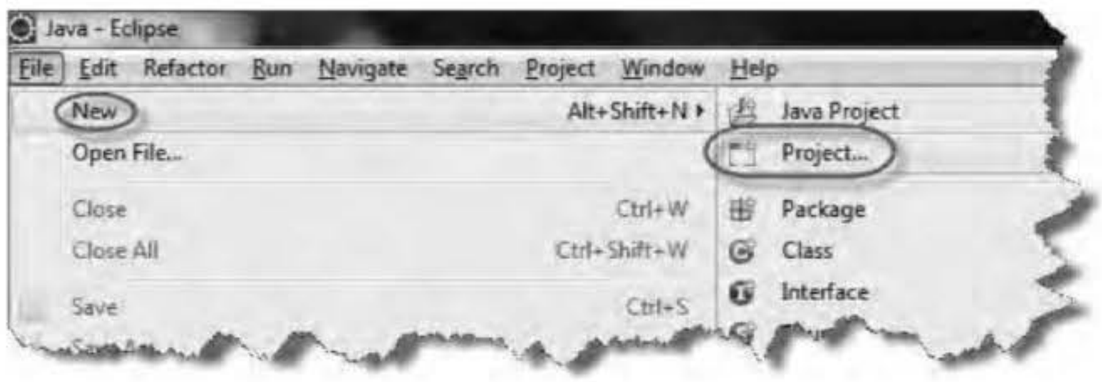


Con todas las herramientas y el SDK descargado e instalado, es el momento de empezar a crear la primera aplicación Android. Como la mayoría de libros de programación, el primer ejemplo que se utiliza es la aplicación Hola. Esto le permitirá tener una visión detallada de los diversos componentes que conforman un proyecto de Android.

Paso 1: Haga doble clic en el ícono ECLIPSE.



Paso 2: Utilizando Eclipse, cree un nuevo proyecto, seleccionando File > New > Project.



Paso 3: Seleccione el tipo de proyecto: Android Project.



Paso 4: Ingrese el nombre del proyecto: "HolaPeru" (nombre interno para Eclipse). Si desea guardar el proyecto en otro lugar, active la casilla Use default location e indique la ruta.



Nota

- *Se recomienda utilizar nombres para paquete a través de un dominio en forma inversa:*

  com.nolascov.holaperu

  El dominio del ejemplo anterior es nolascov.com.

- *Se recomienda usar un sistema de sintaxis de puntos ya referenciado ejemplo:*

  Com.inkadroid.HolaPeru

- *Se recomienda no usar mayúsculas, letras o caracteres especiales como "ñ".*
- *Se recomienda usar la convencion camelCase.*

Paso 5: Presione Siguiente (o Next).



Paso 6: Presione Siguiente (o Next).



Nota

Puede especificar una imagen distinta.

Paso 7: Haga clic en Siguiente (o Next).



Paso 8: Puede escribir un nombre para la Actividad o solo presione Finalizar (o Finish).



3.2. EXPLORAR LA APLICACIÓN



Revise el proyecto para conocer su estructura. En el Explorador de paquetes, debería ver un proyecto llamado "HolaPeru". Si se expande con todos sus nodos, esta constituye la estructura general de la mayoría de los proyectos de Android.

3.2.1. src/

Contiene todos los archivos fuente de Java. Tenga en cuenta que el paquete tiene el mismo nombre que el especificado en el asistente de proyecto Android.

3.2.2. gen/

Contiene los archivos fuente de Java generado por el sistema Android de construcción. No deberán modificarse, se regenera automáticamente, en algunos casos.

3.2.3. res/

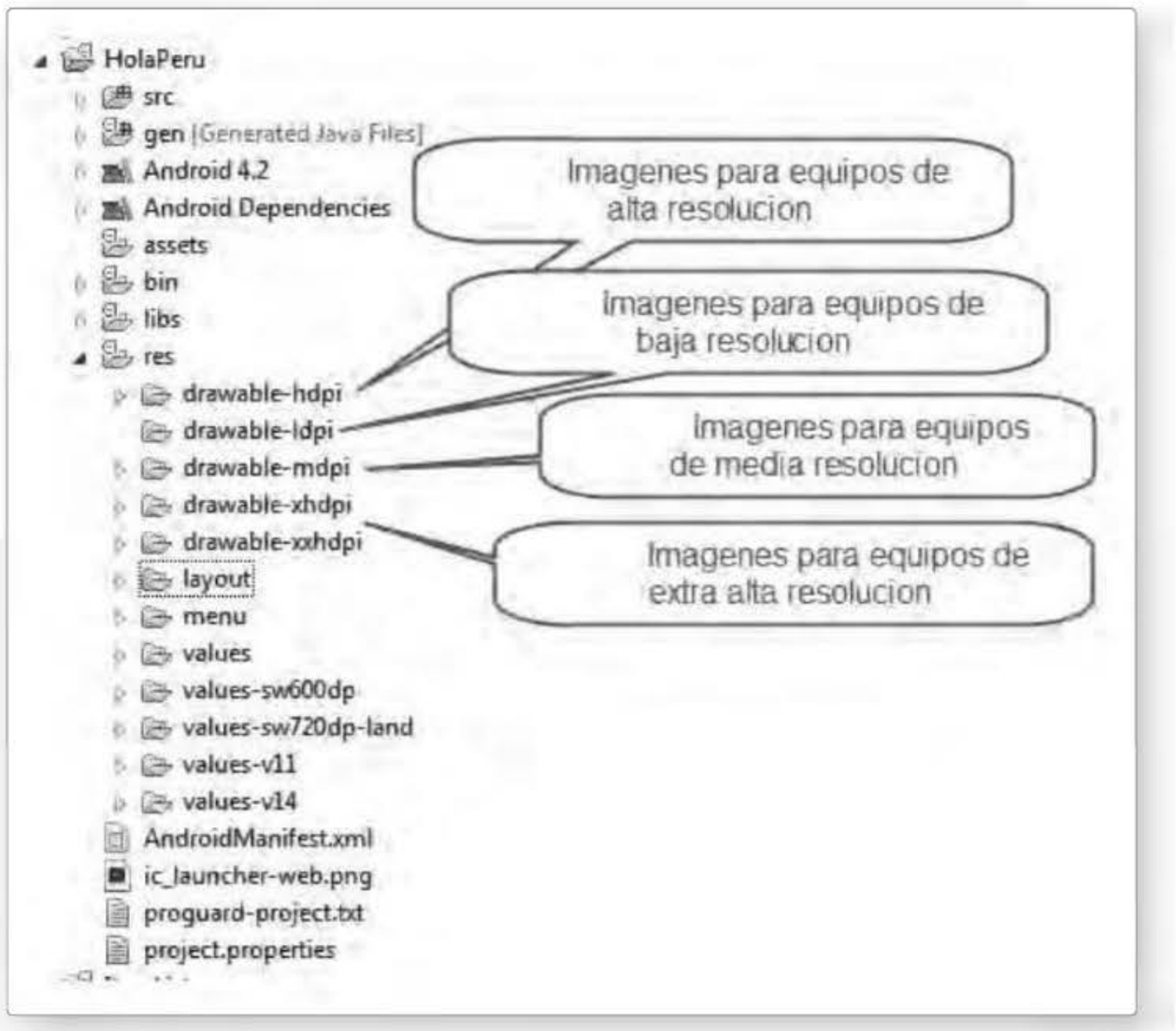
Presenta carpetas e interfaces (constituyen los recursos necesarios).

A. Interfaces

- AndroidManifest.xml: Describe su aplicación. En él se definen las actividades y servicios que la componen. La versión Android que se supone que corren y de los permisos que necesita (por ejemplo, el acceso a la tarjeta SD o en red).
- default.Properties: Tiene varios ajustes para el sistema de construcción. No se va abordar en este tema, ya que el ADT plug-in se encargará de modificarlo cuando sea necesario.

B. Carpetas

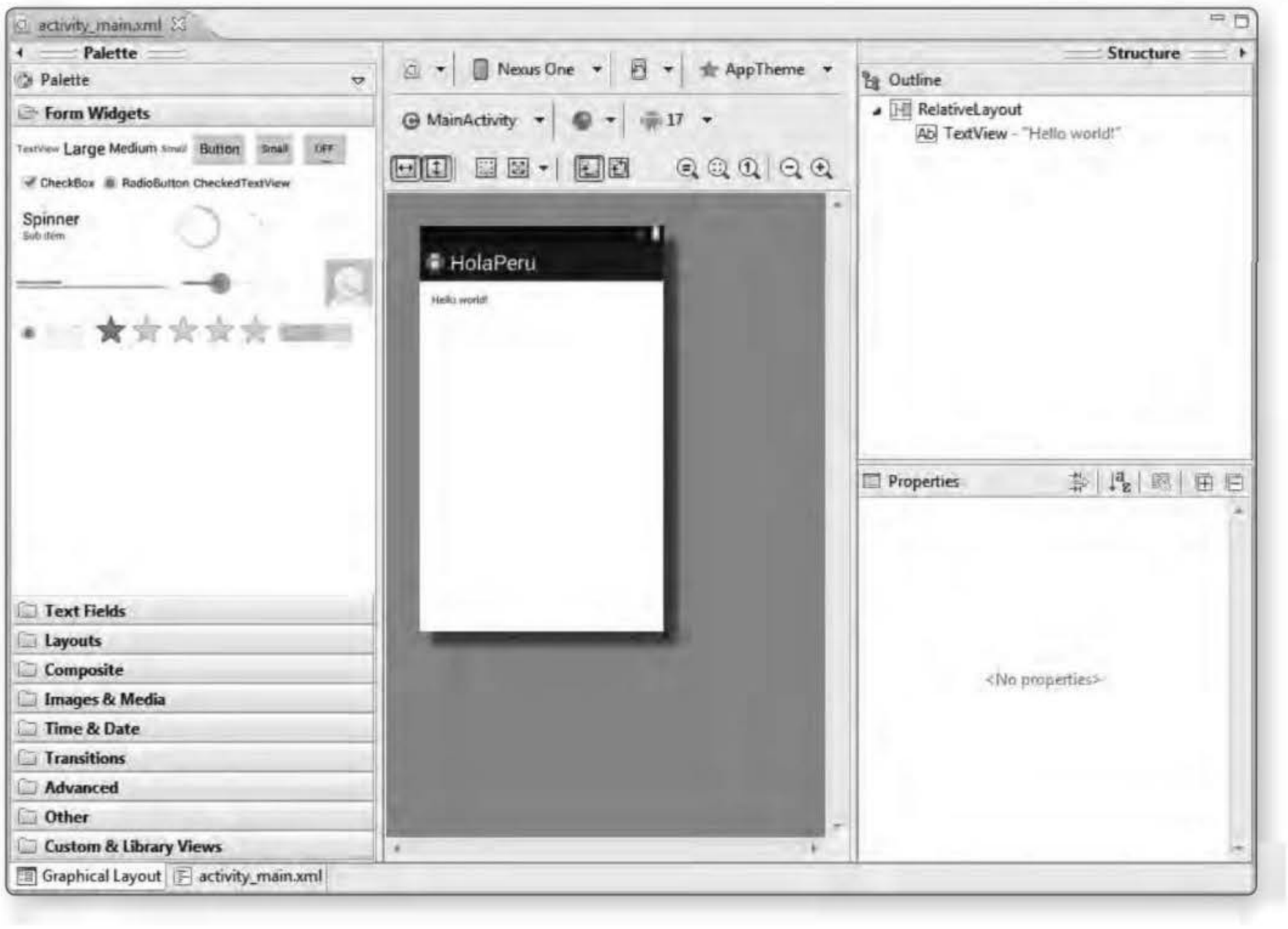
- res/drawable: Para imágenes.



- res/layout: Para interfaces.
- res/values: Para strings.

En el EXPLORADOR DE PAQUETES (que se encuentra a la izquierda del IDE de Eclipse), expanda el proyecto HolaPeru y haga clic en las flechas de diversos que a la izquierda de cada elemento aparece en el proyecto. En la carpeta res/layout, haga doble clic en el archivo main.xml.

El archivo activity_main.xml define la interfaz de usuario (UI) de su aplicación. La vista predeterminada es la vista DISEÑO, que establece la actividad gráfica. Para modificar la interfaz de usuario, haga clic en la pestaña main.xml, situado en la parte inferior y edite el archivo activity_main.xml (ver fig.).



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

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/hello_world" />

</RelativeLayout>
```

EJECUTANDO MI PRIMERA APLICACIÓN

Paso 1: *Guarde los cambios realizados, haciendo clic en el ícono*



o presionando la combinación Ctrl+S.

Paso 2: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción RUN AS > ANDROID APPLICATION y observe que el emulador se pondrá en marcha:*



Paso 3: *Haga clic y deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.



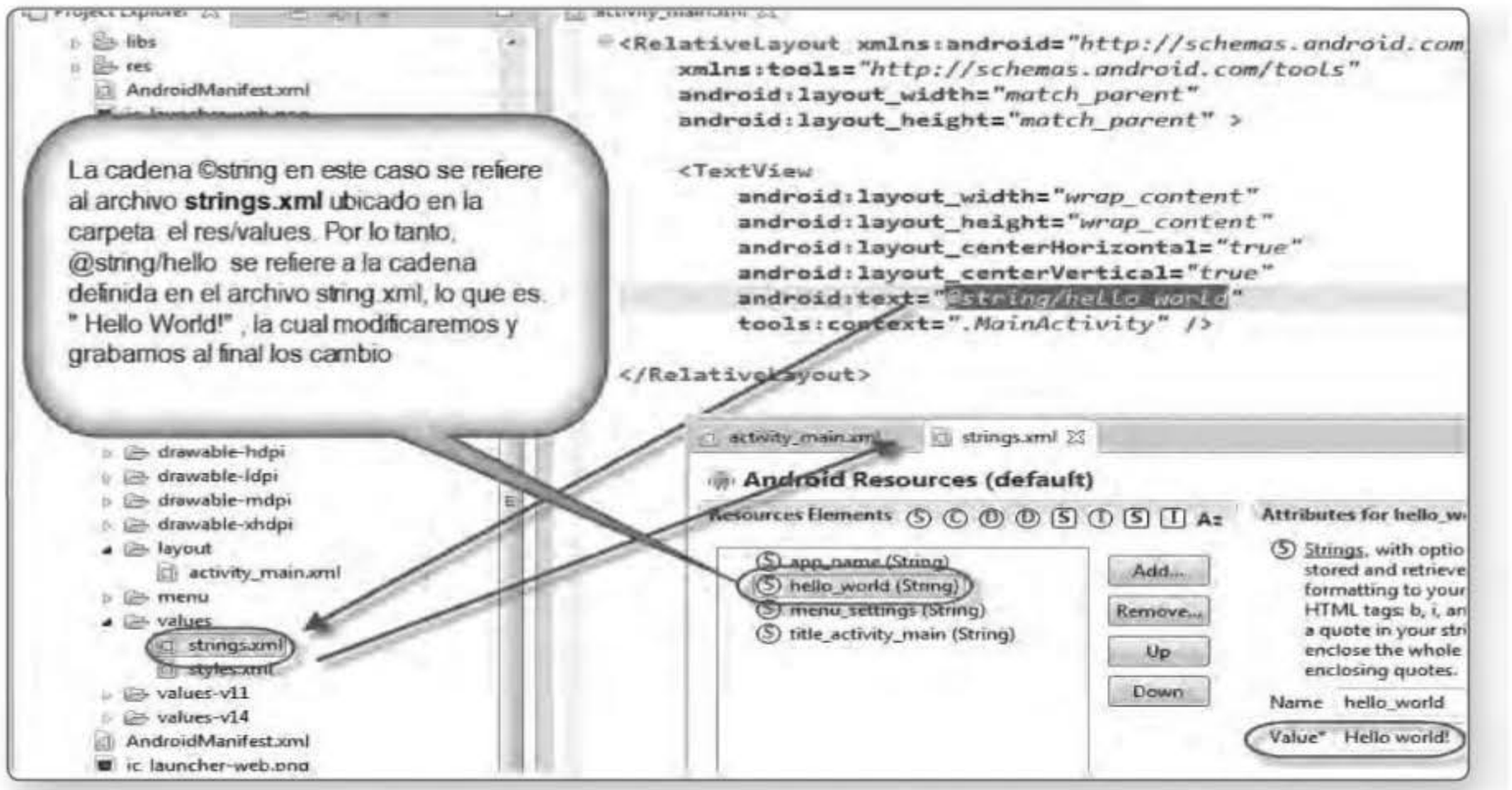
3.3. MODIFICAR TEXTO

Para mostrar el texto "Esta es mi primera aplicación" en reemplazo de "Hello Word!", existen dos formas de realizarlo.

- Modificar el archivo de Interfaz activity_main.xml (poco recomendada):



- Modificar la constante (forma recomendada):





EJECUTANDO MI PRIMERA APLICACIÓN

Paso 1: *Guarde los cambios realizados, haciendo clic en el ícono*



o presionando la combinación Ctrl+S.

Paso 2: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción RUN AS > ANDROID APPLICATION y observe que el emulador se pondrá en marcha:*

Paso 3: *Haga clic y deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.



Nota:

- *Para modificar la fuente de Texto de la interfaz XML, ingrese a:*
  Preferencias > General > Apariencia > Color y fuente > Basic > Text Font
- *Para modificar la fuente de Texto del código Java, diríjase a:*
  Preferencias > General > Apariencia > Color y fuente > Java > Java Editor Text Font

3.4. ATRIBUTOS DE DISEÑO

LAYOUT	DESCRIPCIÓN	EJEMPLO
xmlns:android	Esto define el espacio de nombres XML que se utilizará para hacer referencia al SDK de Android.	xmlns:android="http://schemas.android.com/apk/res/android
orientation=""	Esto indica a Android que la orientacion sera en forma vertical u Horizontal.	android:orientation="vertical"
android:layout_width=""	Esto indica que debe llenar el máximo espacio horizontal.	android:layout_width="fill_parent"
android:layout_height=""	Esto indica que debe llenar el espacio tanto vertical. En definitiva, se debe hacer la altura de tan alto como puede.	android:layout_height="fill_parent"

3.5. FICHEROS

3.5.1. Fichero activity_main.xml



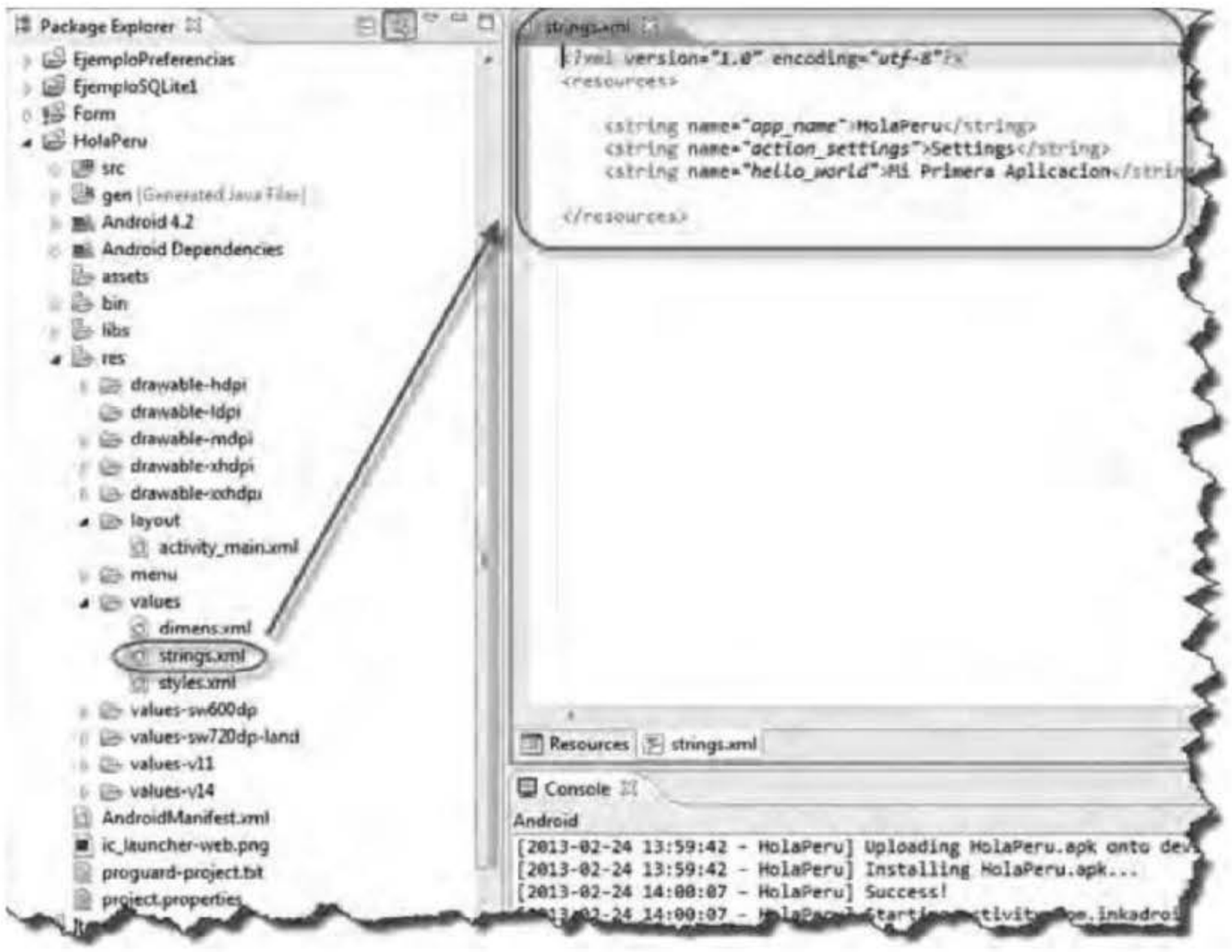
El archivo activity_main.xml define la interfaz de usuario de su actividad. Observe lo siguiente:

```
<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" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:padding="@dimen/padding_medium"
        android:text="Esta es mi Primera Aplicacion"
        tools:context=".HolaPeru" />

</RelativeLayout>
```

La **cadena ©string**, en este caso, se refiere al archivo strings.xml ubicado en la carpeta res/values. Por lo tanto, **@string/hello** se refiere a la cadena definida en el archivo string.xml lo que es "Hello World, HolaPeruActivity!":



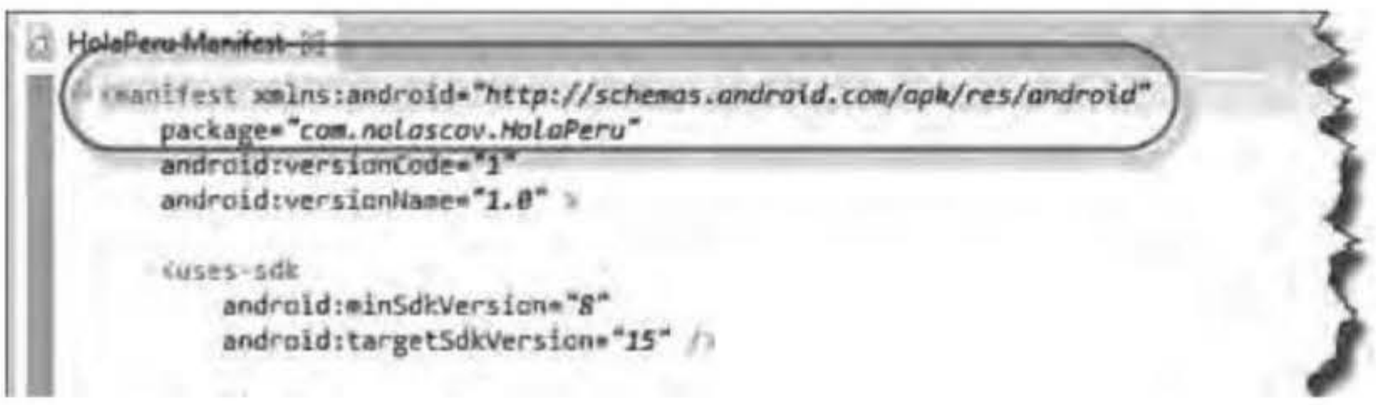
> ***Nota:***
>
> *Se recomienda el uso de strings.xml para almacenar cadenas constantes en su aplicación.*

3.5.2. Fichero AndroidManifest.xml (metadatos de la aplicación)

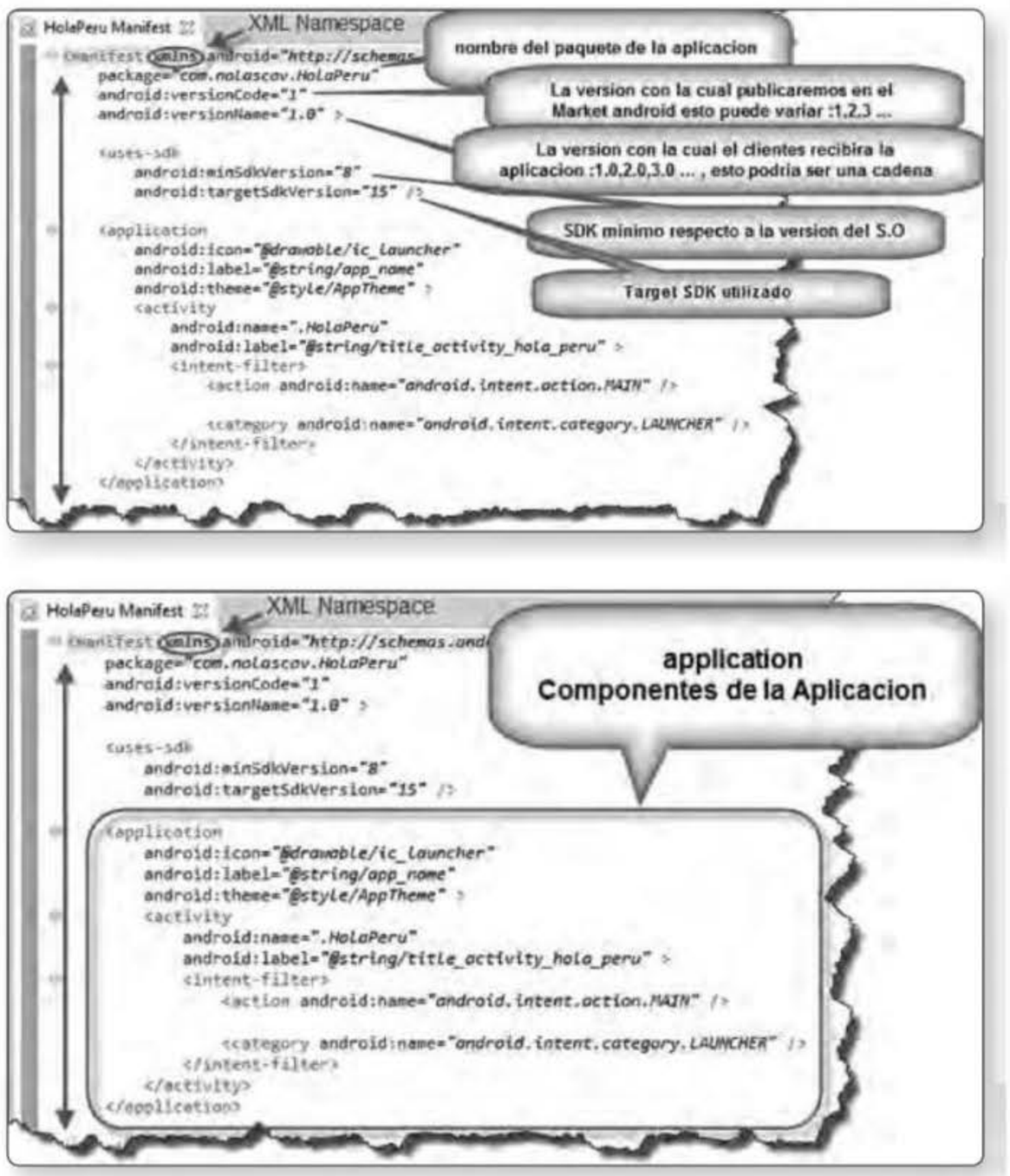
Un elemento importante e imprescindible en cualquier aplicación para Android es el archivo denominado «AndroidManifest.xml». Este archivo, generado de forma automática por el plug-in de Eclipse, representa un manifiesto escrito en XML que describe, de forma genérica, cada uno de los componentes que forman la aplicación. Esta descripción abarca aspectos como sus capacidades y requisitos, las clases que los implementan, los datos que pueden manejar o cuándo deben ser lanzados. En este fichero, también pueden concretarse permisos y políticas de seguridad que afectan a toda la aplicación. Observe el contenido del archivo AndroidManifest.xml:

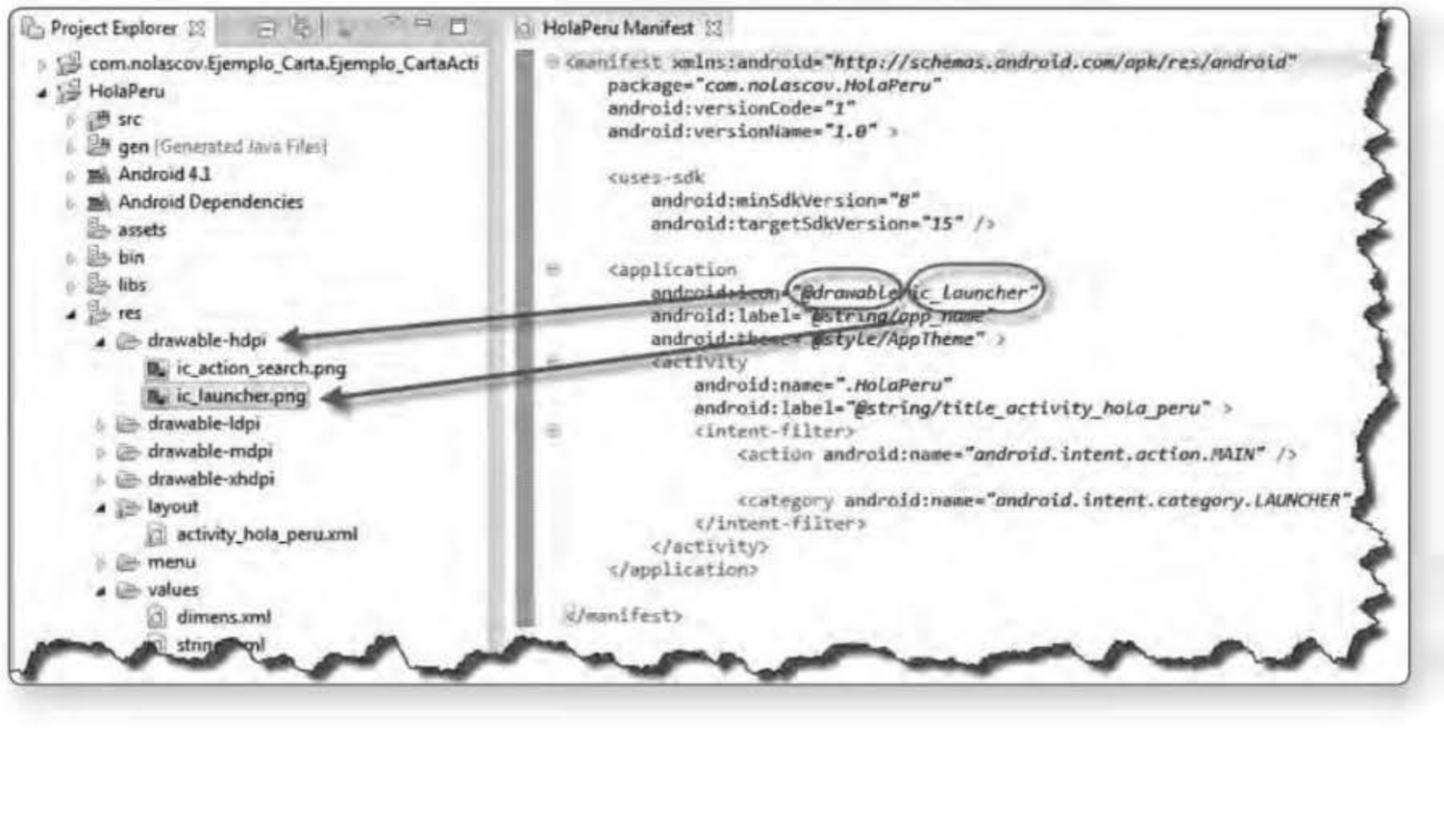


El archivoAndroidManifest.xml contiene información sobre la aplicación: En este manifiesto, primero, se declara el documento como un XML válido y, segundo, se adjunta un espacio de nombres adecuados en el elemento <manifest>, así como el paquete al que pertenece con el atributo package.

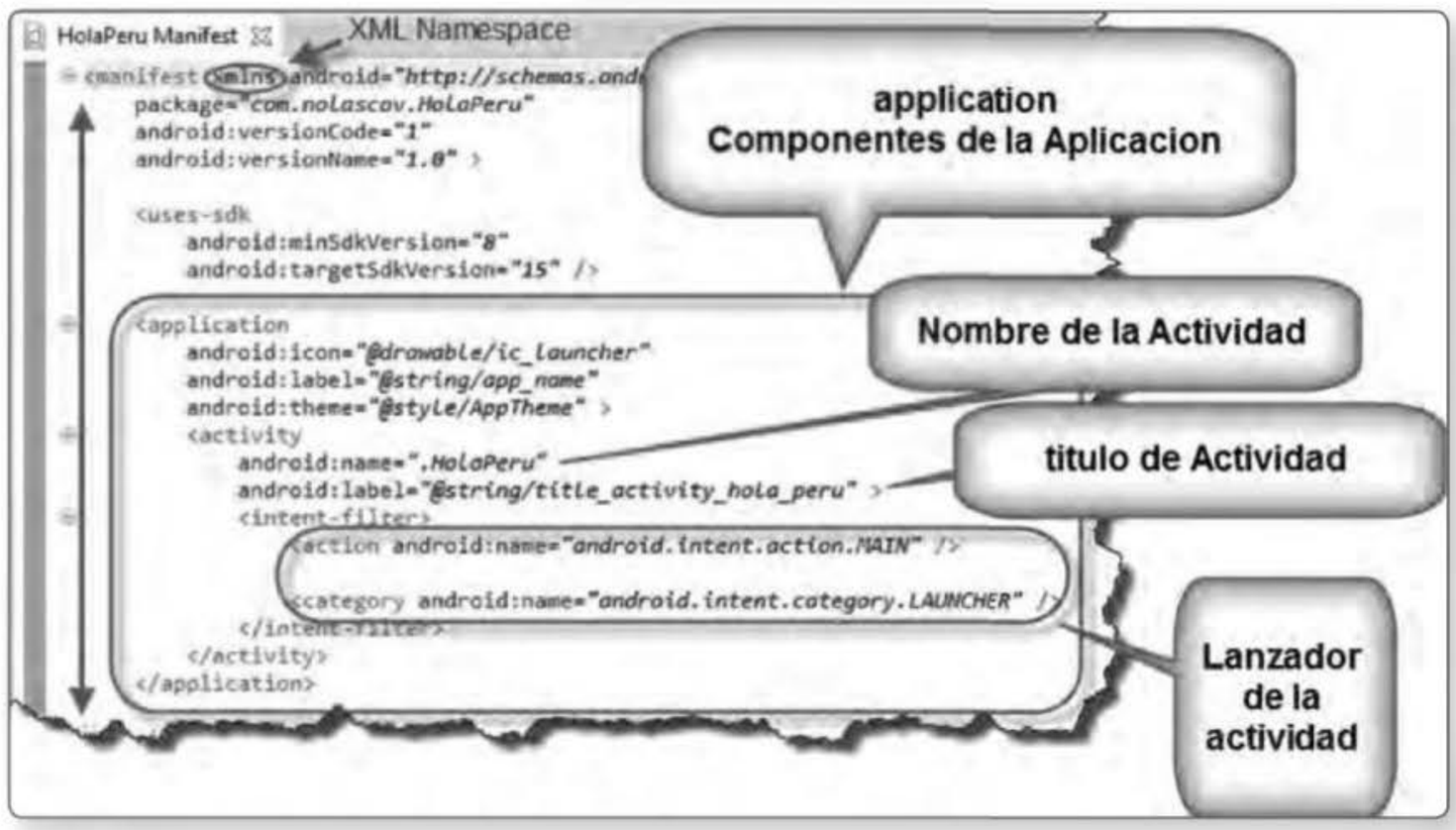


A. Definición de interfaces de usuario con XML

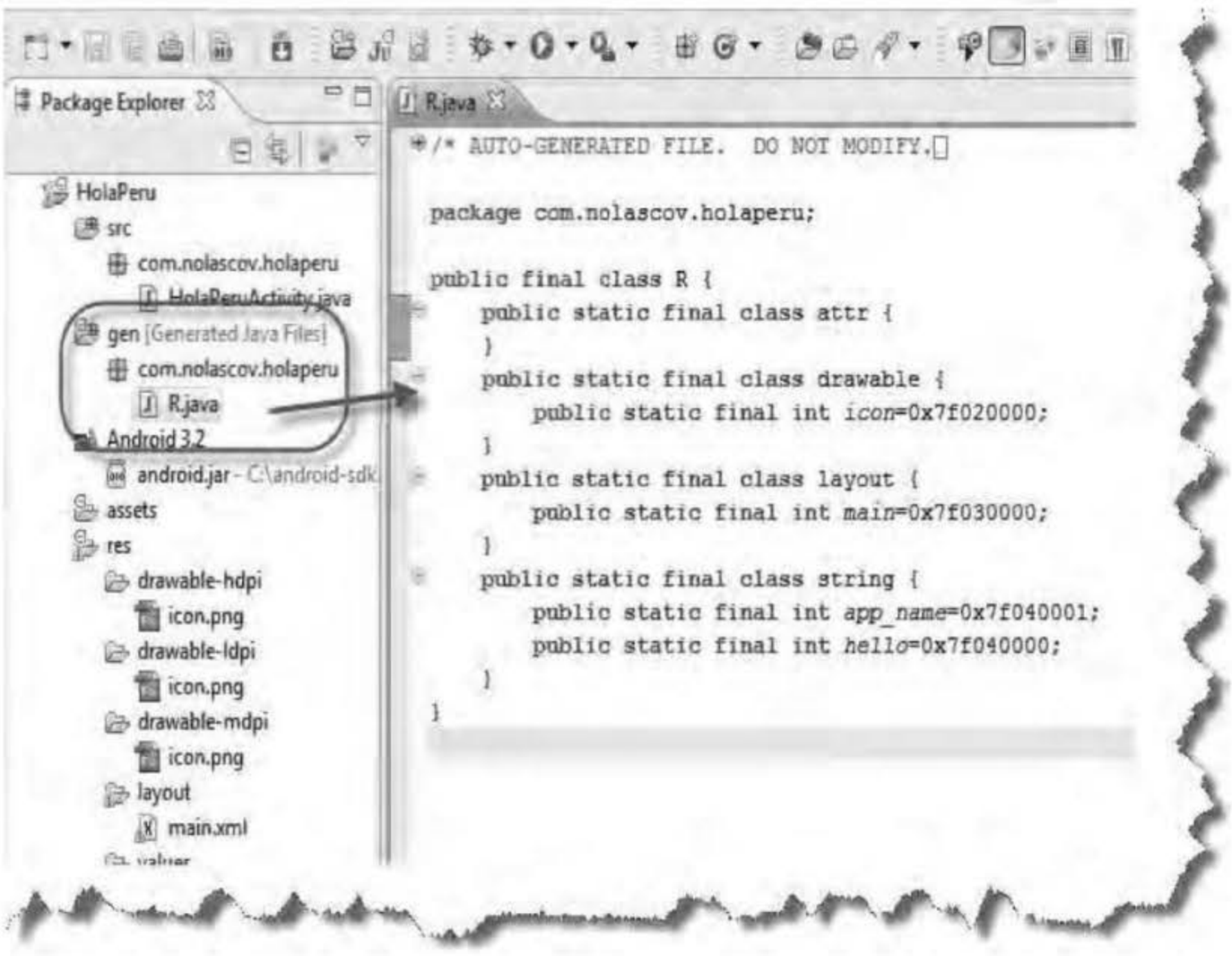




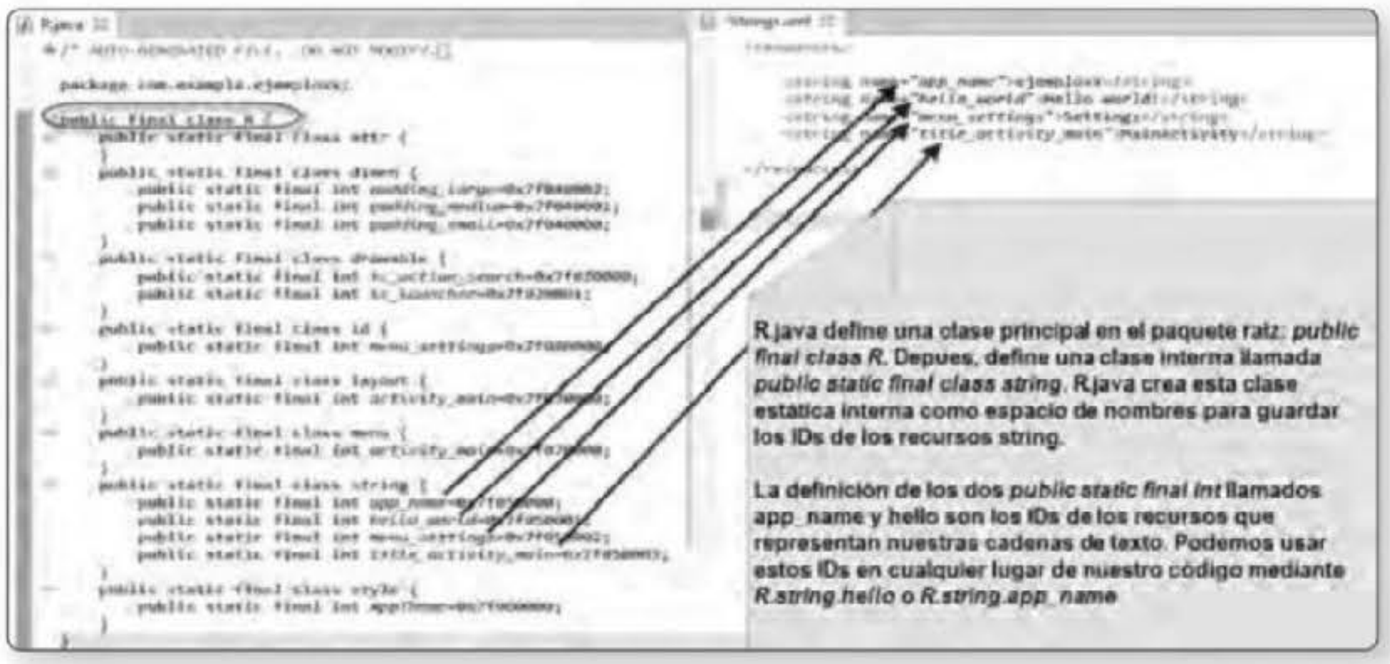




Se puede agregar más archivos y carpetas a su proyecto, ya que Eclipse genera de forma automática el contenido R.java el cual contiene lo siguiente:



ARCHIVO R.JAVA



No se puede modificar el contenido del archivo R. java, ya que Eclipse genera de forma automática el contenido de este archivo cuando se modifica el proyecto.

Nota:

Si elimina R. java manualmente, Eclipse lo regenerará de forma inmediata. Tenga en cuenta que con el fin de Eclipse, se regenera el archivo R. java y el proyecto no debe contener ningún error. Si se percata que Eclipse no regenera R. java después de haberlo eliminado, compruebe nuevamente el proyecto. El código puede contener errores de sintaxis o los archivos XML pueden no estar bien formados. Para regenerar el archivo R.java, selecione el proyecto, haga clic derecho > ANDROID UTILIDADES > FIX PROJECT PROPERTIES.

ARCHIVO BUILDCONFIG.JAVA

A partir del SDK 17, se ha añadido esta característica la cual permite ejecutar un código pero solo en modo de depuración a través de una constante que se ajusta automáticamente, de acuerdo con el tipo de construcción.

BuildConfig.java

```
/** Automatically generated file. DO NOT MODIFY */
package com.example.ejemplocontroles1;

public final class BuildConfig {
    public final static boolean DEBUG = true;
}
```

ARCHIVO MAINACTIVITY. JAVA

Finalmente, el código que conecta la actividad de la interfaz de usuario (main.xml) es el método setContentView(), que está en el archivo HolaPeruActivity. java:

*MainActivity.java

```
package com.example.ejemplocontroles1;

import android.os.Bundle;

public class MainActivity extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.activity_main, menu);
        return true;
    }
}
```

Observe lo siguiente:

```
//Como en Java cada clase deberá poseer un constructor en Android es reeemplezado por el método  onCreate
    public void onCreate(Bundle savedInstanceState) {
//esta sentencia es recomendable hacerla siempre que estemos sobrescribiendo un método de una clase padre
    super.onCreate(savedInstanceState);
// Aquí R. layout .main se refiere al archivo main.xml  que se encuentra en la carpeta res / layout
    setContentView(R.layout.main);
    }

// método  para crear menus
@Override
    public boolean onCreateOptionsMenu(Menu menu) {
            getMenuInflater().inflate(R.menu.activity_main, menu);
            return true;
    }
```

Al crear una clase que se extienda de Activity (MainActivity), se debe implementar, al menos, la función onCreate() la cual se ejecuta cuando se crea la actividad. Por defecto, esa función recibe un paquete (Bundle), que contiene el estado anterior de la actividad; en caso de que se haya suspendido. Como se puede observar en el cuadro anterior, la clase HolaPeruActivity le transfiere ese paquete a la superclase de la que extiende.

Lo siguiente que se hace en la función de creación es cargar el layout o diseño de la actividad, cargando una interfaz definida en XML. Como se explicó en una entrada anterior, los diseños se guardan en res/layout/ y para llamarlos se utiliza "R", indicando el nombre del fichero. Por ejemlo, "main.xml" pasa a ser "R.layout.main". Si se quisiera cargar otro layout solo habría que crearlo y cargarlo con "R.Otrolayout".

A medida que se agrega más archivos XML en la carpeta res/layout, los nombres de archivo se generan automáticamente en el archivo de R.java. El método onCreate() es uno de los múltiples métodos que se disparan cuando una actividad se ha cargado

3.6. LLEVAR EL APLICATIVO AL DISPOSITIVO MÓVIL PARA SER INSTALADO Y EJECUTADO

Lo que se necesita es distribuir el archivo Apk al dispositivo móvil, para luego ser instalado. Para ello realice los siguientes pasos:



Paso 1: Haga clic derecho del mouse sobre el proyecto y seleccione Export.



Paso 2: Seleccione Android > Export Android Application, luego pulse el botón Siguiente (o Next).



Paso 3: Verifique el nombre del proyecto, luego pulse el botón Siguiente (o Next).



Paso 4: Cree una clave que será importante en el Market de Android porque servirá para futuras actualizaciones en el Market de Android (Play Google), luego pulse el botón Siguiente (o Next).



Paso 5: Llene información necesaria para generar la clave.



- *Alias:* Identificador de la clave.
- *Password:* Contraseña de la clave que debe guardarse o recordarse, pues se necesitara cada vez que se quiera publicar una nueva actualización de la aplicación en Android Market.
- *Confirm:* Confirmar la contraseña anterior.
- *Validity (years):* Validez del certificado, mínimo 25 años.
- *First and Last Name:* Nombre del desarrollador o de la empresa.
- *Organization Unit:* Departamento.
- *Organization:* Nombre de la empresa.

- *City or Locality:* Ciudad.
- *State or Province:* Provincia.
- *Country Code:* Código ciudad.

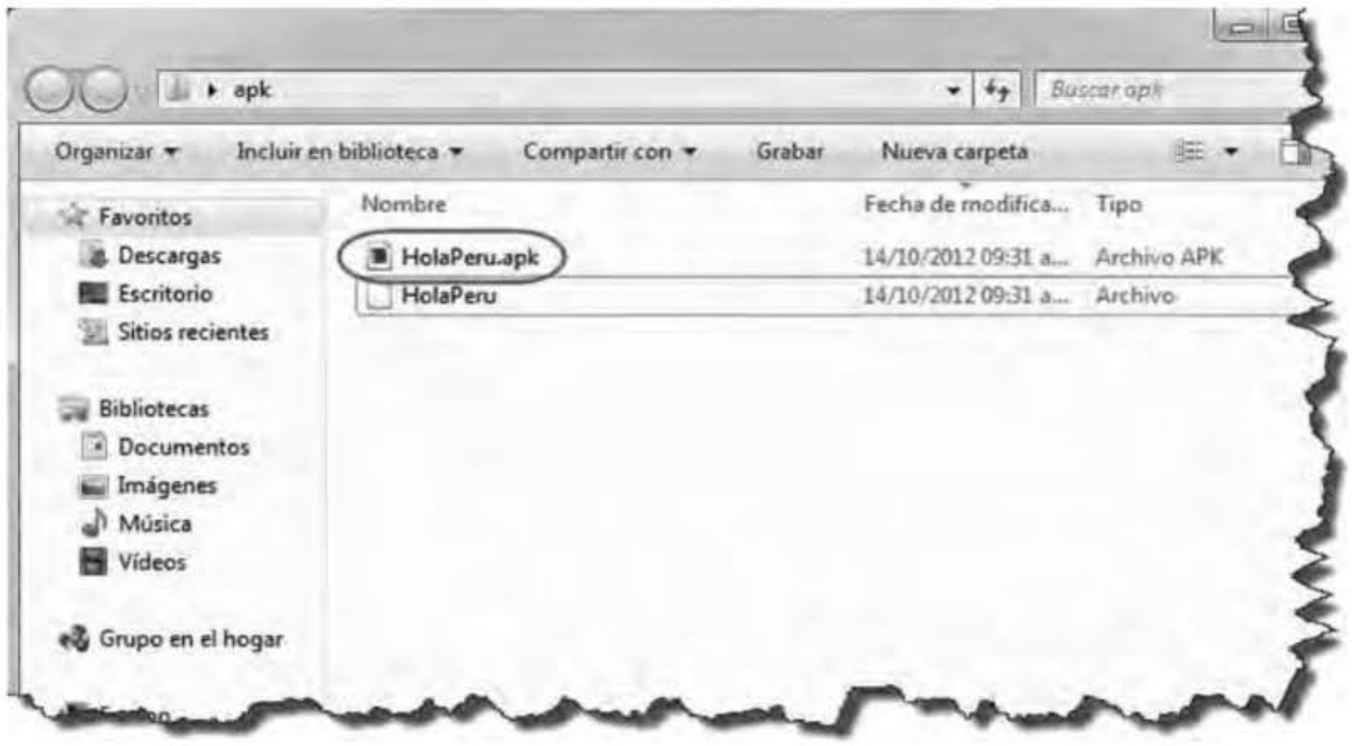
Paso 6: Una vez llenado los datos, pulse Siguiente.

> ***Nota:***
> *Se recomienda firmar todas las aplicaciones con el mismo certificado y en un periodo mayor a 25 años.*

Paso 7: Indique el Apk a generar.



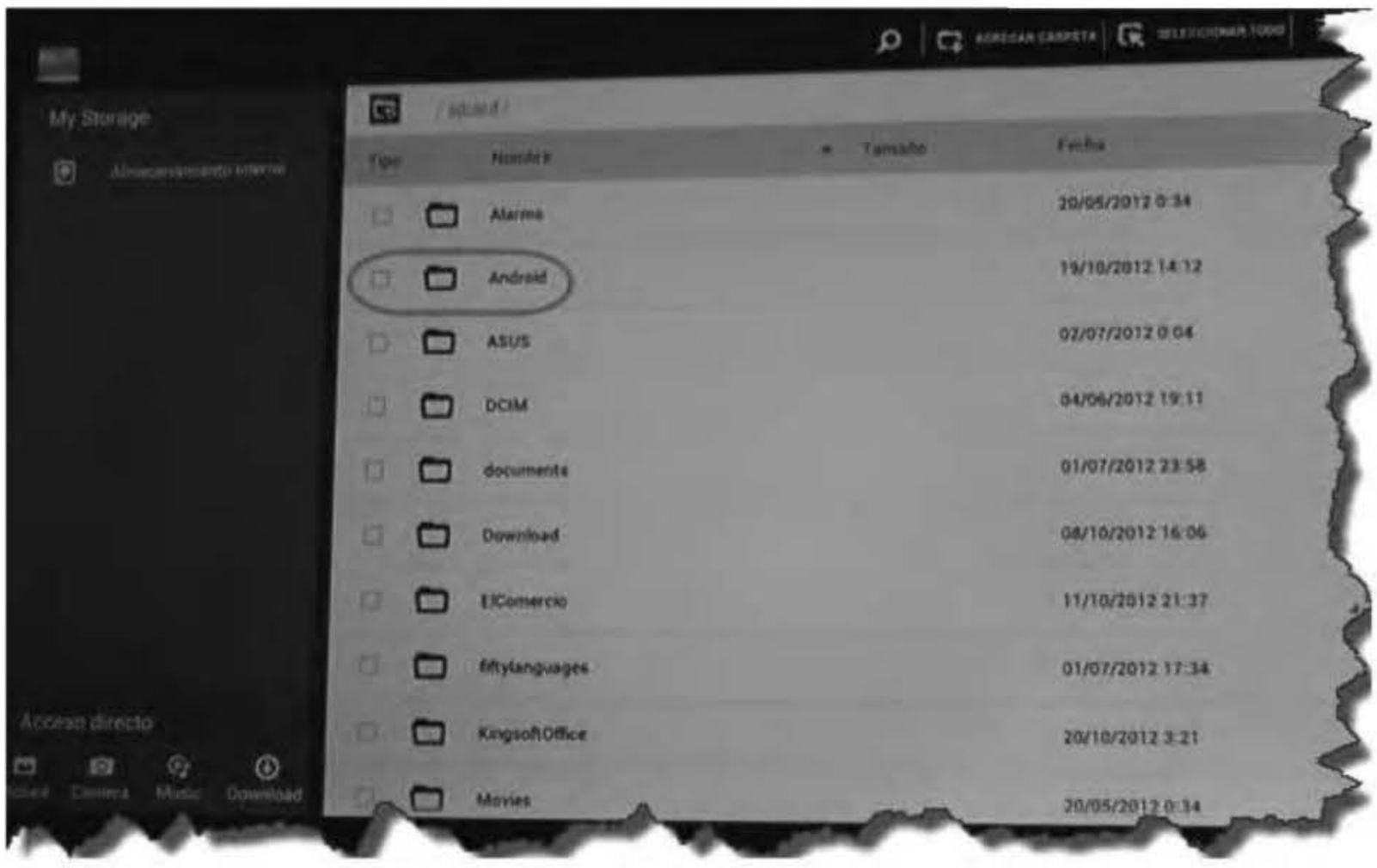
Paso 8: Ubique el fichero Apk el cual se distribuirá al dispositivo móvil, mediante cable USB o Bluetooth.



Paso 9: Conecte el cable USB al dispositivo móvil y copie el APK.



Paso 10: Ingrese al sistema de archivos de la ASUS transformer tf101.



Paso 11: Ingrese a la carpeta Android y haga doble click en el APK (HolaPeru.apk).



Paso 12: Se mostrará un mensaje "Quieres instalar esta aplicación", haga clic en el botón Instalar, luego se le indicará si desea aperturar la aplicación; para ello, presione el botón Abrir.



3.7. APP GRATUITA O PAGADA

La decisión de publicar una aplicación de pago o gratuita en Android Market es personal de cada desarrollador o empresa; pero tenga en cuenta que, por la filosofía libre de Android como sistema operativo para móviles, al contrario que iOS de Mac para iPhone, la mayor parte de las aplicaciones son gratuitas. Por lo tanto, solo debe publicar aplicaciones de pago si son necesarias, auténticas y su tipo es escasa en Android Market; de lo contrario, el usuario de un dispositivo móvil encontrará decenas o cientos de aplicaciones similares y gratuitas.

3.8. RECOMENDACIONES PARA EL DESARROLLO DE APLICACIONES ANDROID

Siga las siguientes recomendaciones:

- Siempre tenga presente que se está desarrollando aplicaciones para dispositivos con pantalla pequeña y teclado muy limitado; por lo que no han de ser aplicaciones con muchos campos de texto ni demasiadas opciones. Esto complicaría el manejo por parte del usuario.
- Hay que partir siempre del hecho de que Android es un sistema con filosofía de código abierto, por lo que los usuarios que han elegido dispositivos con este sistema tenderán a buscar aplicaciones gratuitas. Por ello, si desea obtener beneficios para su aplicación, esta debe ser interesante, genuina, divertida; si se trata de un juego, debe ser sencilla; si es una herramienta, debe incorporar funciones que no tengan otras.
- Procure, en la medida de lo posible, realizar aplicaciones que sean visibles e instalables en el mayor número posible de dispositivos; de esta forma, tendrá más difusión y con ello más posibles compradores. Por lo tanto, realice aplicaciones con API Level bajos y no con demasiados requisitos.
- Las aplicaciones deben ser rápidas. Si necesita realizar algún proceso que considere que tardará unos segundos, es recomendable que avise de esto previamente al usuario. Considere siempre que un usuario de un dispositivo móvil espera rapidez; de lo contrario, puede que cierre la aplicación y no espere a que finalice.

CAP.

4

Estructura de una aplicación Android

En este apartado, se explicará sobre los distintos componentes que forman una aplicación Android y algunas de las nuevas características de Android 4.0. En particular, se dará a conocer sobre el nuevo fragmento y las API de la barra de acciones en Android 4.0 y cómo se pueden utilizar para desarrollar atractivas aplicaciones tablets. Seguidamente se presentarán los componentes de una aplicación Android.



4.1. ACTIVIDAD (ACTIVITY)

En Android, una actividad es una ventana que contiene la interfaz de usuario (o pantalla) para su aplicación y los usuarios interactúan directamente con las actividades de sus aplicaciones. En otras palabras, la actividad es el componente más habitual de las aplicaciones para Android. Refleja una determinada actividad llevada a cabo por una aplicación y que lleva asociada típicamente una ventana o interfaz de usuario.

La mayoría de las aplicaciones permiten la ejecución de varias acciones a través de la existencia de una o más pantallas.

Ejemplo:

Una aplicación que muestra en la primera pantalla un listado de clientes el cual posee la característica de seleccionar algunos clientes y en una segunda pantalla muestra un detalle de cada cliente (datos generales, por ejemplo) y mediante el despliegue de una tercera ventana se puede visualizar el estado de cuenta de los clientes. Cada una de estas ventanas debería estar representada a través de un componente Activity, de forma que navegar de una ventana a otra implique lanzar una actividad o pausar otra. Android permite controlar por completo el ciclo de vida de los componentes Activity.

La actividad se ubica en la siguiente ruta (según el ejemplo):



Se observa que la clase HolaPeruActivity se extiende (extends) de la clase Activity (clase base).

MainActivity.java

```
package com.inkadroid.holaperu;

import android.os.Bundle;

public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }

}
```

4.1.1. Descripción del código de la actividad

package com.nolascov.HolaPeru;
import android.app.Activity;
import android.os.Bundle;

Crea una clase publica MainActivity extiende o heredea características de la clase base Activity

```
public class MainActivity extends Activity {
```

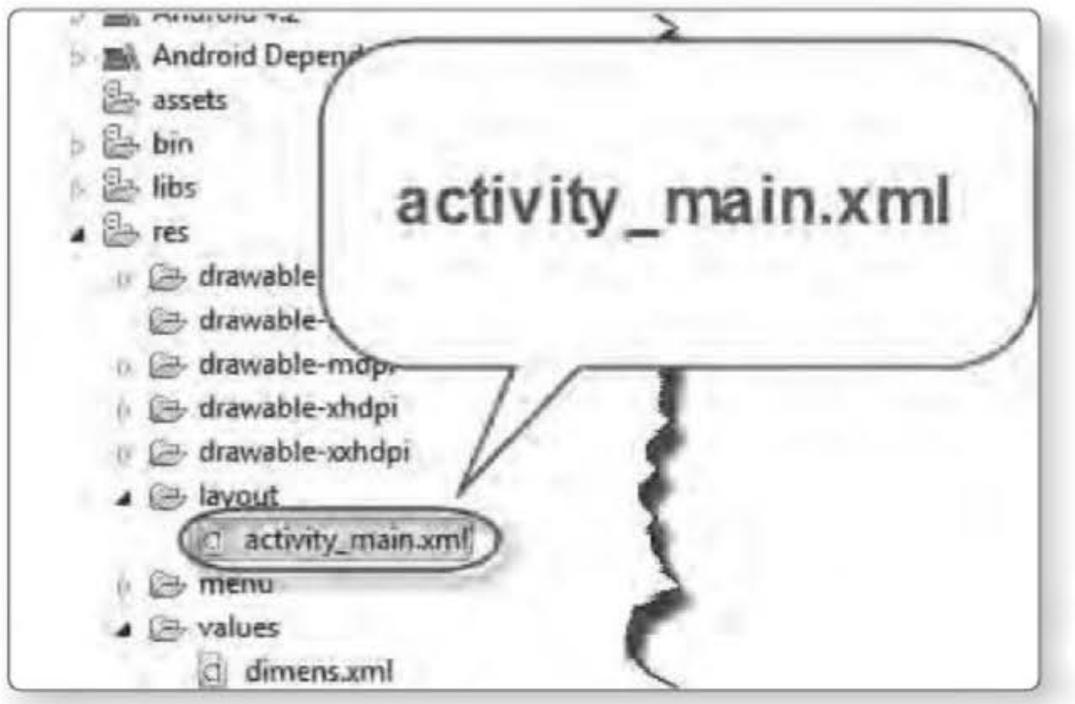
/** Called when the activity is first created. */
@Override

Este método es donde usted inicializa su actividad.

```
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
/*
Con setContentView se Carga el componente de usuario que se encuentra en la carpete
res/layout/main.xml
*/
setContentView(R.layout.main);
}
}
```

Nota:

La clase actividad carga la interfaz de usuario, utilizando el archivo XML que se encuentra en la carpeta: res/ layout



En el siguiente figura, se carga la interfaz de usuario del archivo (XML) activity_main.xml:



Cada actividad deberá ser declarada dentro del archivo de manifiesto AndroidManifest.xml.



HolaPeru Manifest

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

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="17" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.inkadroid.holaperu.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

4.1.2. Ciclo de vida de las aplicaciones Android

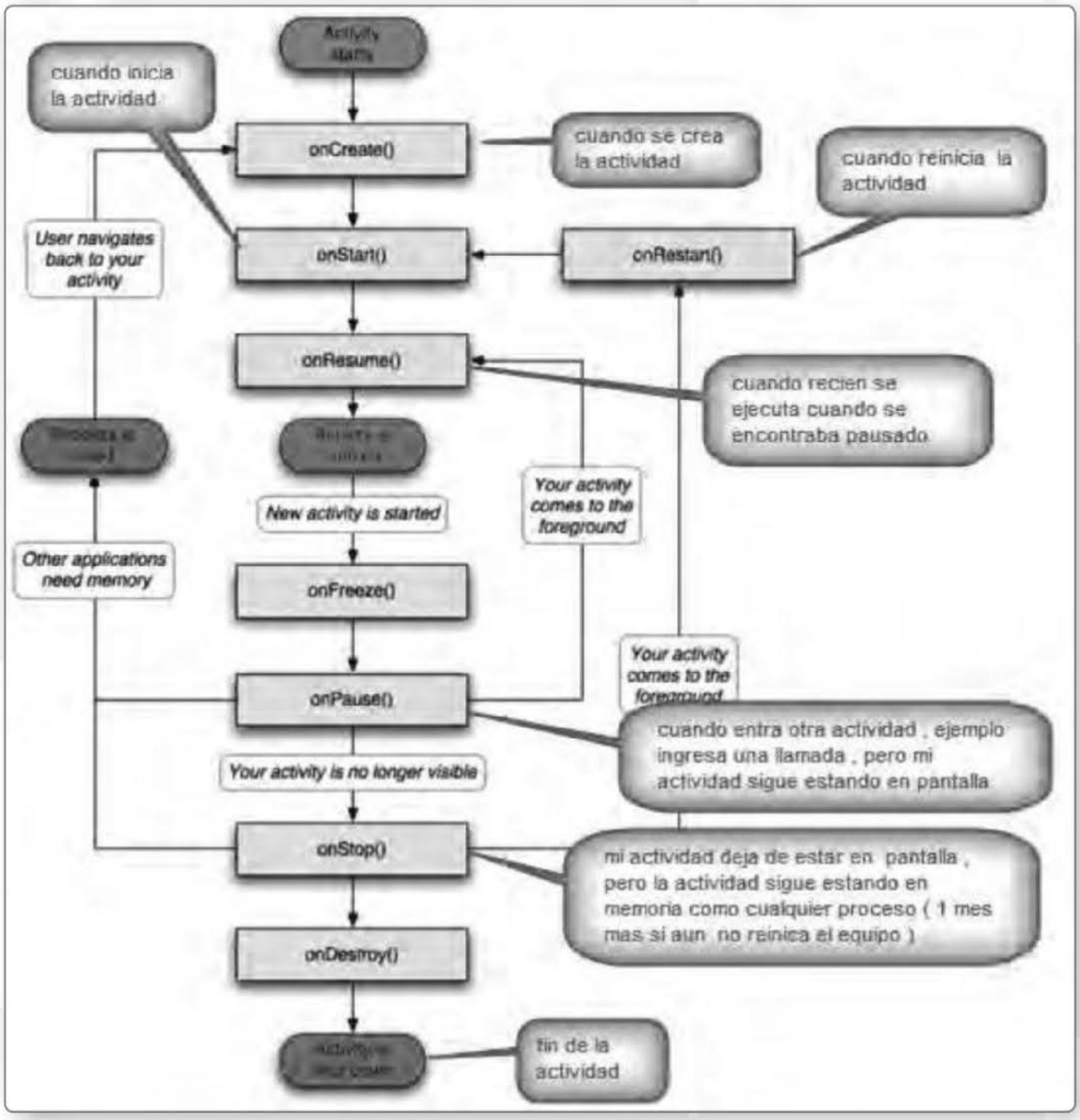


En Android, cada aplicación se ejecuta en su propio proceso. Esto aporta beneficios en cuestiones básicas como seguridad, gestión de memoria o la ocupación de la CPU del dispositivo móvil. Android se ocupa de lanzar y parar todos estos procesos, gestionar su ejecución y decidir qué hacer en función de los recursos disponibles y de las órdenes dadas por el usuario.

El usuario desconoce este comportamiento de Android. Simplemente, es consciente de que mediante un simple clic pase de una a otra aplicación y puede volver a cualquiera de ellas en el momento que lo desee. No debe preocuparse sobre cuál es la aplicación que realmente está activa, cuánta memoria está consumiendo ni si existen o no recursos suficientes para abrir una aplicación adicional. Todo eso son tareas propias del sistema operativo.

Android lanza tantos procesos como permitan los recursos del dispositivo. Cada proceso, correspondiente a una aplicación estará formado por una o varias actividades independientes (componentes Aciiviry) de esa aplicación. Cuando el usuario navega de una actividad a otra o abre una nueva aplicación, el sistema duerme dicho proceso y realiza una copia de su estado para poder recuperarlo más tarde. El proceso y la actividad siguen existiendo en el sistema, pero están dormidos y su estado ha sido guardado. Es entonces cuando se crea o despierta si ya existe el proceso para la aplicación que debe ser lanzada, asumiendo que existan recursos para ello.

Cada uno de los componentes básicos de Android tiene un ciclo de vida bien definido; esto implica que el desarrollador puede controlar en cada momento en qué estado se encuentra dicho componente, pudiendo así programar las acciones más convenientes. El componente Activity, probablemente el más importante, tiene un ciclo de vida el cual se grafica en la siguiente figura:

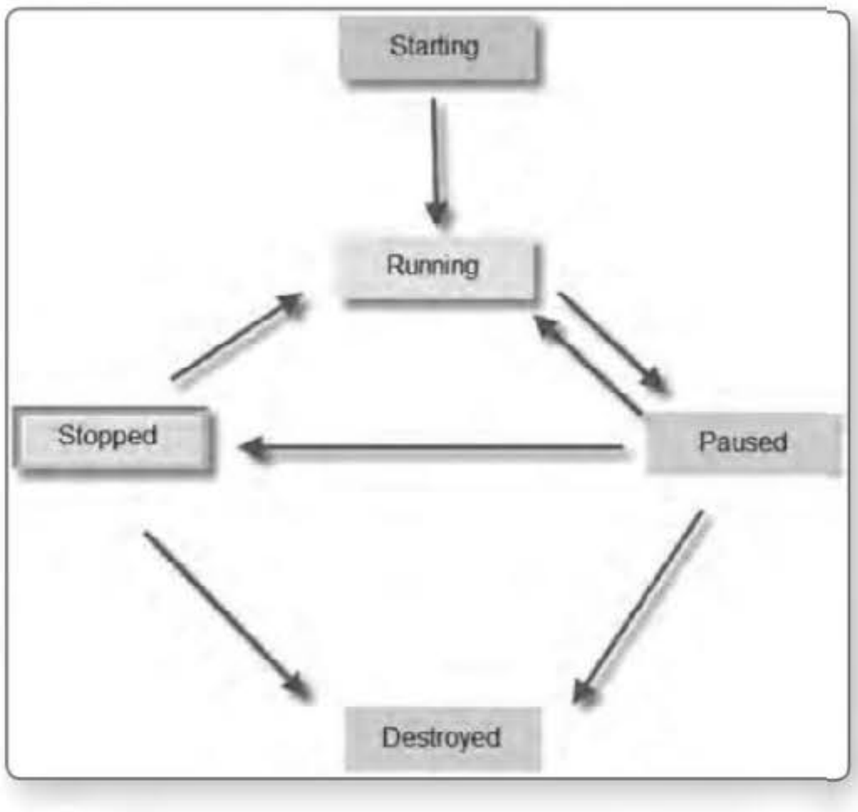


Donde:



- onCreato(), onDestroy(): Abarcan todo el ciclo de vida. Cada uno de estos métodos representa el principio y el fin de la actividad.
- onStart(), onStop( ): Representan la parte visible del ciclo de vida. Desde onStart() hasta onStop(), la actividad será visible para el usuario, aunque es posible que no tenga el foco de acción, por existir otras actividades superpuestas con las que el usuario esté interactuando y pueden ser llamados múltiples veces.
- onResume() onPause(): Delimitan la parte útil del ciclo de vida.
- Desde onResume() hasta onPause(), la actividad no solo es visible, sino que además tiene el foco de la acción y el usuario puede interactuar con ella.

4.1.3. Orden de los eventos en Android



El orden de los eventos es el siguiente:

1. Al ejecutarse por primera vez la aplicación
   - Constructor
   - Evento onStart
   - Envento onResume
   - Evento surfaceCreated
   - Evento surfaceChanged

2. En el momento que se pulsa la tecla Inicio pasa por los eventos
   - onPause
   - surfaceDestroyed
   - onStop

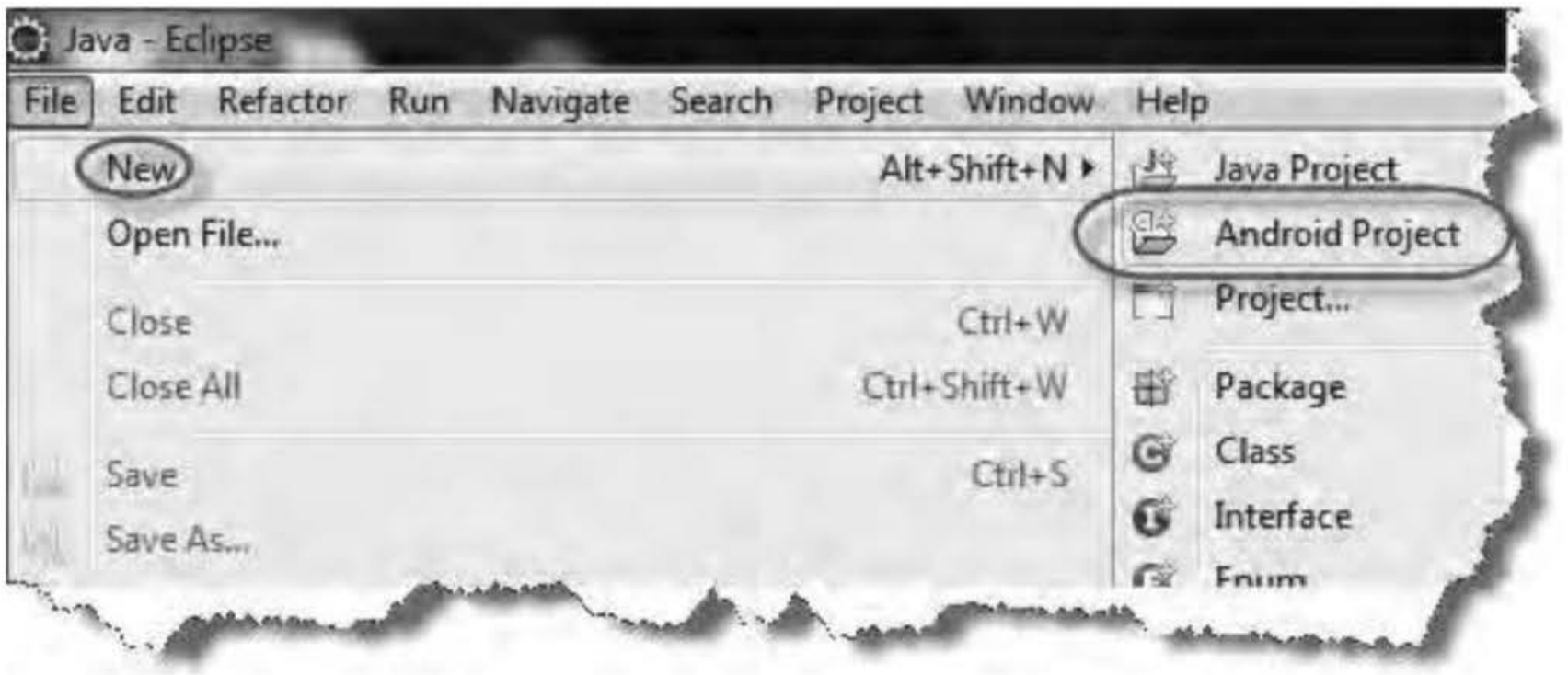
3. Al retornar a la aplicación pasa por los eventos
   - onRestart
   - onStart
   - onResume
   - surfaceCreated
   - surfaceChanged

4. En el momento de pulsar la tecla Esc pasa por lo eventos
   - OnPause
   - onStop
   - surfaceDestroyed

4.1.4. Etapas de una actividad (Activity)

Para entender el ciclo de vida de una actividad, cree un nuevo proyecto:

Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto.*



Paso 2: Ingrese el nombre del proyecto: HolaPeru.





Paso 3: Voy a compilar para la versión 4.0 y la API 14 (esto decide la disponibilidad de API que se tendrá y el soporte para los equipos, ya que solo se podrá ejecutar en equipos que soporten las API seleccionadas).



Paso 4: Colocar el nombre de la actividad y para finalizar, presione el botón *Finish*.

Nota:

Se recomienda utilizar nombres para paquete a través de un dominio en forma inversa com.nolascov.Actividad (el dominio es nolascov.com). Asimismo, opte por un sistema de sintaxis de puntos, ya referenciado; por ejemplo: com.nolascov.Actividad.

Ahora para finalizar, presione el botón *Finish*.

4.1.5. Antes de codificar el Archivo MainActivity.Java

Depuración en Android: Logging



«[...]. Una de las técnicas más útiles a la hora de depurar y/o realizar el seguimiento de aplicaciones sobre cualquier plataforma es la creación de logs de ejecución. Android por supuesto no se queda atrás y nos proporciona también su propio servicio y API de logging a través de la clase android.util. Log.

En Android, todos los mensajes de log llevarán asociada la siguiente información:

- Fecha/Hora del mensaje.
- Criticidad. Nivel de gravedad del mensaje (se detalla más adelante).
- PID. Código interno del proceso que ha introducido el mensaje.
- Tag. Etiqueta identificativa del mensaje (se detalla más adelante).
- Mensaje. El texto completo del mensaje.

De forma similar a como ocurre con otros frameworks de logging, en Android los mensajes de log se van a clasificar por su criticidad, existiendo así varias categorias (ordenadas de mayor a menor criticidad):

- Error
- Warning
- Info
- Debug
- Verbose

Para cada uno de estos tipos de mensaje existe un método estático independiente que permite añadirlo al log de la aplicación. Así, para cada una de las categorías anteriores tenemos disponibles los métodos e(), w(), i(), d() y v() respectivamente.

Cada uno de estos métodos recibe como parámetros la etiqueta (tag) y el texto en sí del mensaje. Como etiqueta de los mensajes, aunque es un campo al que podemos pasar cualquier valor, suele utilizarse el nombre de la aplicación o de la actividad concreta que genera el mensaje. Esto nos permitirá más tarde crear filtros personalizados para identificar y poder visualizar únicamente los mensajes de log que nos interesan, entre todos los generados por Android [que son muchos] durante la ejecución de la aplicación».

Fuente: www.sgoliver.net

Una de los paquetes de Android, android.util, contiene una clase llamada «Log» que permite imprimir mensajes de debug en la pestaña antes referenciada. El método:

```
Log.d(titulo, mensaje);
```

Permite escribir estos mensajes, que después Dalvik Debug Monitor puede capturar. En dicho método, title representa el título o etiqueta que se desee dar, a fin de poder filtrar después todos los mensajes de dicha categoría; por otro lado, el mensaje indica el mensaje concreto que se quiere imprimir bajo esta etiqueta.

En el archivo MainActivity.Java file, escriba lo siguiente:

*ActividadActivity.java

```
package com.nolascov.Actividad;

import android.app.Activity;

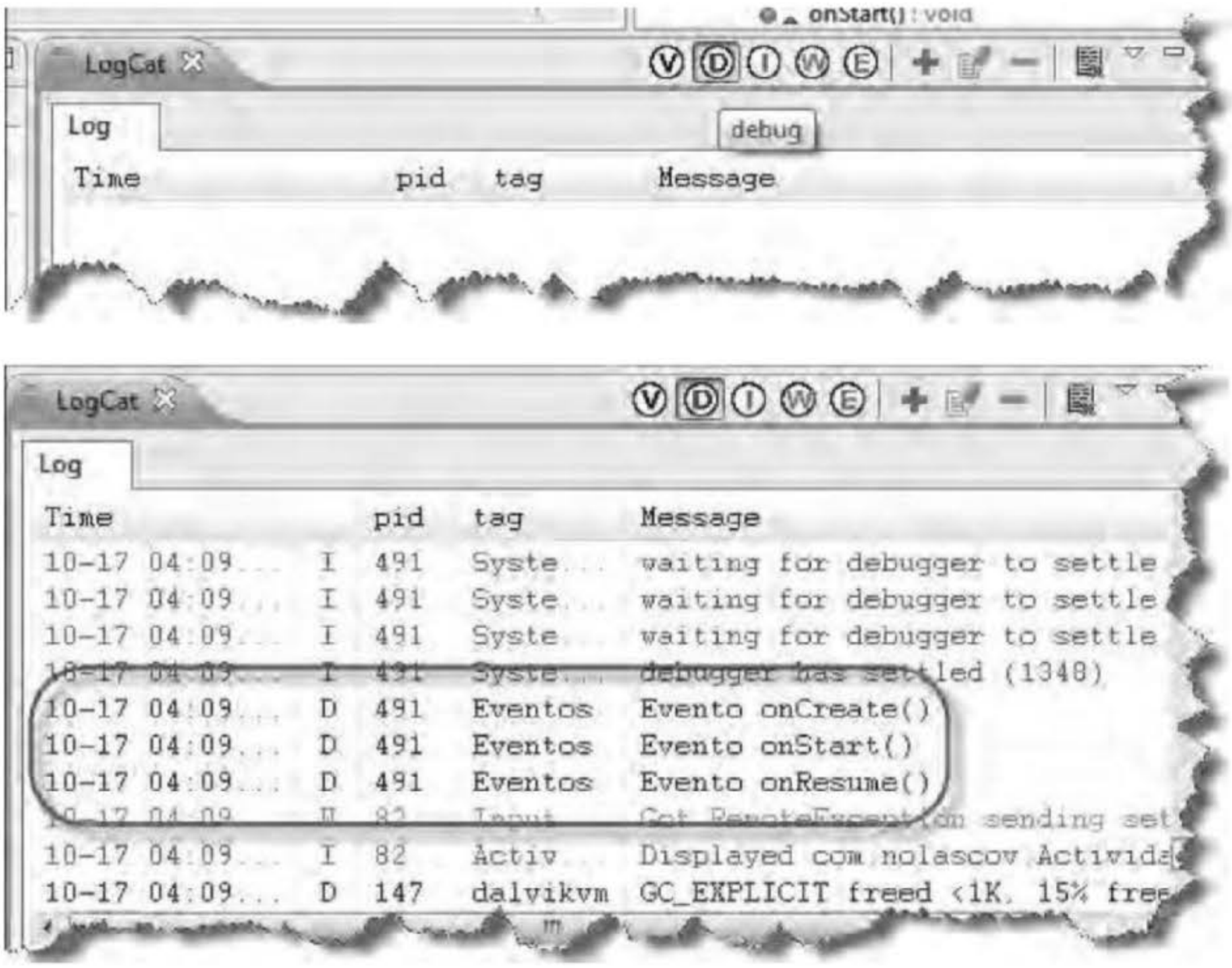
public class ActividadActivity extends Activity {
    String etiqueta="Eventos";
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        Log.d(etiqueta,  "Evento onCreate()");
    }
    public void onStart() {
        super.onStart();
        Log.d(etiqueta, "Evento onStart()");
        }
    public void onRestart() {
        super.onRestart();
        Log.d(etiqueta,  "Evento onRestart()");
        }
    public void onResume() {
        super.onResume();
        Log.d(etiqueta, "Evento onResume()");
        }
    public void onPause() {
        super.onPause();
        Log.d(etiqueta,  "Evento onPause()");
        }
    public void onStop() {
        super.onStop();
        Log.d(etiqueta,  "Evento onStop()");
        }
    public void onDestroy() {
        super.onDestroy();
        Log.d(etiqueta, "Evento onDestroy()");
        }
}
```

«Si ejecutamos la aplicación anterior en el emulador veremos cómo se abre la pantalla principal que crea Eclipse por defecto y aparentemente no ocurre nada más. ¿Dónde podemos ver los mensajes que hemos añadido al log? Pues para ver los mensajes de log nos tenemos que ir a la perspectiva de Eclipse llamada DDMS. Una vez en esta perspectiva, podemos acceder a los mensajes de log en la parte inferior de la pantalla, en una vista llamada LogCat. En esta ventana se muestran todos los mensajes de log que genera Android durante la ejecución de la aplicación, que son muchos, pero si buscamos un poco en la lista encontraremos los generados por nuestra aplicación [...].

Como se puede observar, para cada mensaje se muestra toda la información que indicamos al principio del artículo, además de estar diferenciados por un color distinto según su criticidad.

En este caso de ejemplo, los mensajes mostrados son pocos y fáciles de localizar en el log, pero para una aplicación real, el número de estos mensajes puede ser mucho mayor y aparecer intercalados caóticamente entre los demás mensajes de Android. Para estos casos, la ventada de LogCat ofrece una serie de funcionalidades para facilitar la visualización y búsqueda de determinados mensajes» (www.sgoliver.net).

Por ejemplo, se puede restringir la lista para que solo muestre mensajes con una determinada criticidad mínima. Esto se consigue pulsando uno de los 5 primeros botones que se observan en la parte superior derecha de la ventana de log. De manera que, si por ejemplo pulsa sobre el botón de la categoría debug (en azul); en la lista, solo se mostrarán los mensajes de depuración.



Otro método de filtrado es el denominado «definición de filtros personalizados» (botón "+" verde), donde se puede filtrar la lista que muestra únicamente los mensajes con un PID o Tag determinado. Si ha utilizado como etiqueta de los mensajes el nombre de su aplicación o de sus actividades, esto le proporcionará una forma sencilla de visualizar solo los mensajes generados por su aplicación (www.sgoliver.net).

4.2. INTENTS

Son mensajes que provocan notificaciones y cambios de estatus que al ser recibidos por actividades o servicios pueden levantar procesos. De esta forma, se unen componentes dentro de una misma aplicación o de diferentes aplicaciones. Por ejemplo:

```
// creamos unIntent
Intent intent = newIntent();
//llamamos a formulario o actividad : Form2
intent.setComponent(new ComponentName(this,Form2.class));
```

4.3. VISTAS (VIEWS)

Son los componentes de la interfaz de usuario. Se trata de diferentes vistas que pueden agruparse a través de grupos, logrando una jerarquía. Esto se logra a través de la disposición de los componentes por medio de un archivo XML.

4.4. SERVICIOS (SERVICES)

Son componentes que se ejecutan operaciones en segundo plano (Background) y no tienen una interfaz de usuario.

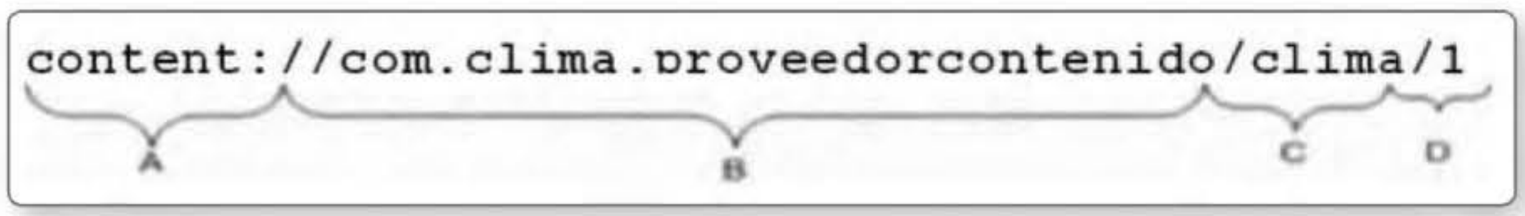
Por ejemplo, al escuchar música hay un servicio encargado de la reproducción que se ejecuta de fondo; un ejemplo muy común, es un antivirus que esta funcionando de fondo

4.5. Proveedores de contendido (Content Providers)

Permite la abstracción para almacenar y obtener datos permanentes de aplicaciones diferentes. El sistema incluye algunos proveedores de contenidos útiles (audio, video, etc.).

Se puede acceder a los datos en formato URL, de la siguiente manera:

Lista de contactos del equipo
content: //contactos/personas



Donde:

A: No se modifica.

B: Aplicación de donde tomar los datos (clima, obtener datos del clima actual). En el caso de sus aplicaciones, coloque un path completo para evitar problemas.

C: Es el path para especificar qué tipo de información de la aplicación se busca (sobre el clima).

D (opcional): Es el ID del registro que se está buscando; en este caso, el mes que se consultó fue el clima.

4.6. Receptores de difusión (Broadcast Receivers)

Son componentes que responden a avisos y a anuncios de difusión (broadcast). Estos avisos provienen del sistema (batería baja, una llamada entrante, etc.) y de aplicaciones (pasando avisos de una aplicación a otra). Aunque no muestra una interfaz de usuario; algunas veces, utiliza barras de progreso para mostrar avances. Estos se activan a través de mensajes asincrónicos llamados «intents».



Fragmentos

5.1. CONCEPTO DE FRAGMENTO



Desde el punto de vista de un desarrollador, la transición más grande de smartphones y tablets es el tamaño de la pantalla por la diversidad que existe.
Los fragmentos sirven para que las aplicaciones en smartphones y tablets puedan ser optimizadas.

5.2. USO DE FRAGMENTOS

Los fragmentos permiten que una aplicación se divida en subcomponentes para poder manejarlos de manera individual y combinarlos de distintas formas como si fuesen paneles. En otras palabras, en un dispositivo de pantalla pequeña (como un teléfono inteligente), una actividad típicamente llena toda una pantalla, mostrando los diversos puntos de vista que componen la interfaz de usuario de una aplicación. La actividad es esencialmente un contenedor. Sin embargo, cuando una actividad se muestra en un dispositivo de pantalla grande, como en una Tablet, la pantalla se vuelve mucho más grande y todos elementos en una actividad deben estar dispuestos a hacer un uso completo de la pantalla más grande lo que resulta en cambios complejos a la jerarquía de visualización. Un mejor enfoque sería contar con miniactividades donde cada una de ellas cuente con su propio conjunto de puntos de vista.

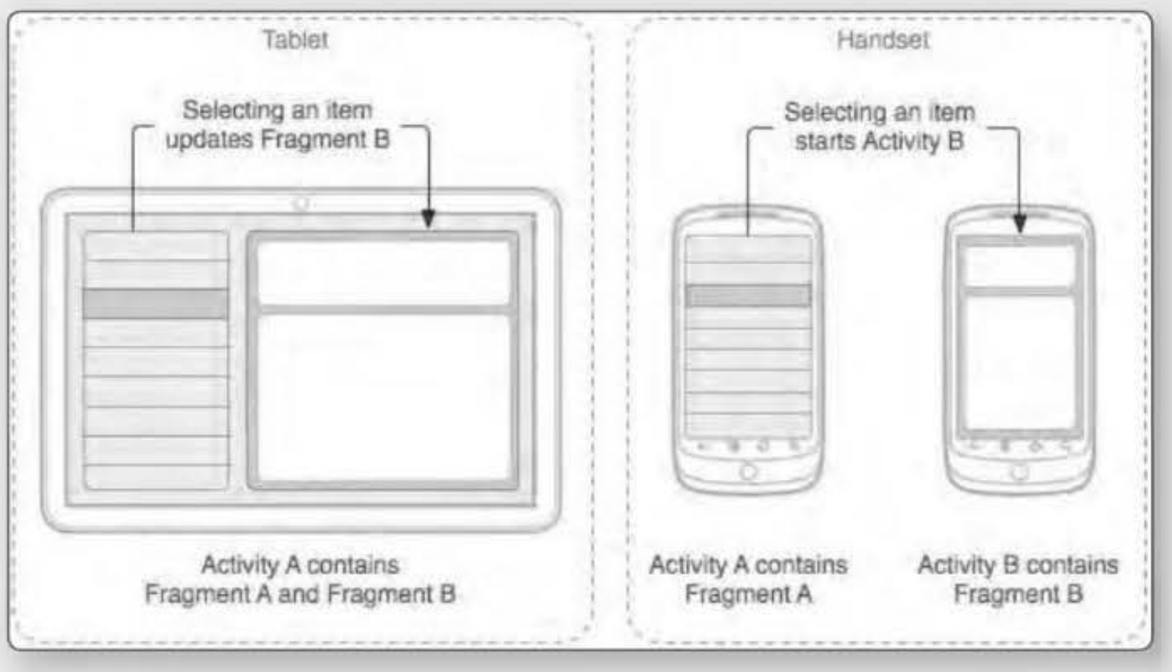
Durante la ejecución, una actividad puede contener uno o más de estas miniactividades, dependiendo de la orientación de la pantalla en la que se lleva a cabo el dispositivo. En Android 3.0, estas miniactividades se conocen como fragmentos.

Ejemplo:

Piense en un fragmento de otra forma de actividad y cree fragmentos que contengan elementos, al igual que las actividades. Considere que los fragmentos siempre deben estar insertos en una actividad.

Suponga, que la aplicación se ejecuta en una tablet de Android (o en un teléfono Android), en el modo vertical. En este caso, el fragmento 1 se pueden incrustar en una actividad; mientras que el fragmento 2, en otra actividad. Cuando los usuarios seleccionen un elemento de la lista en el Fragmento 1, Actividad 2, se pondrá en marcha.

Si la aplicación se muestra ahora en una tablet en modo horizontal; a continuación, los fragmentos pueden ser incorporados dentro de una sola actividad (ver fig.).



A partir de este ejemplo, se hace evidente que los fragmentos presentan una forma versátil en el que puede crear la interfaz de usuario de una aplicación Android. Los fragmentos forman la unidad atómica de la interfaz de usuario y pueden agregar o eliminar, de manera dinámica, a las actividades, con el fin de crear la mejor experiencia de usuario posible que el dispositivo de destino.

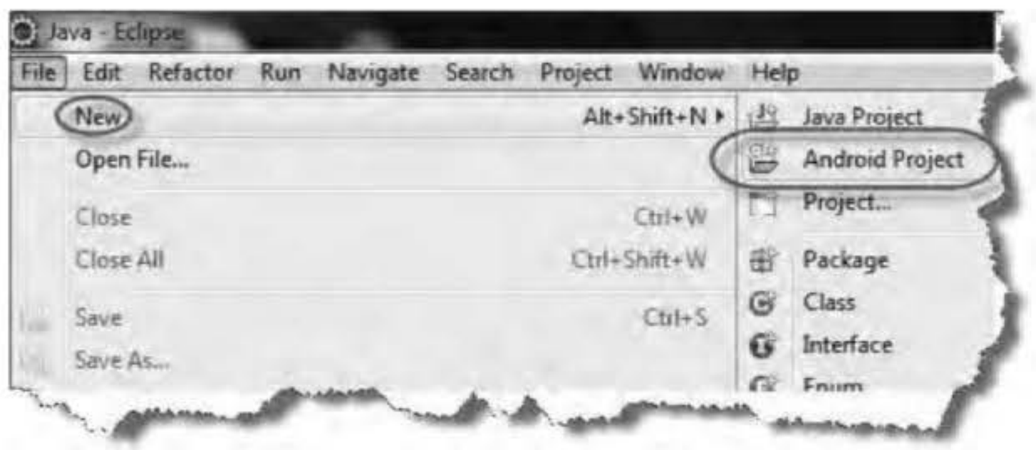
5.2.1. Recomendaciones

Se recomienda para trabajar con fragmentos lo siguiente:

- Por cada fragmento a utilizar, cree un archivo XML que representa la interfaz de usuario.
- Edite el archivo main.xml para definir los fragmentos.
- Cree una clase por cada fragmento el cual herede la clase base Fragment (representa una interfaz de usuario).

5.2.2. Ejemplo

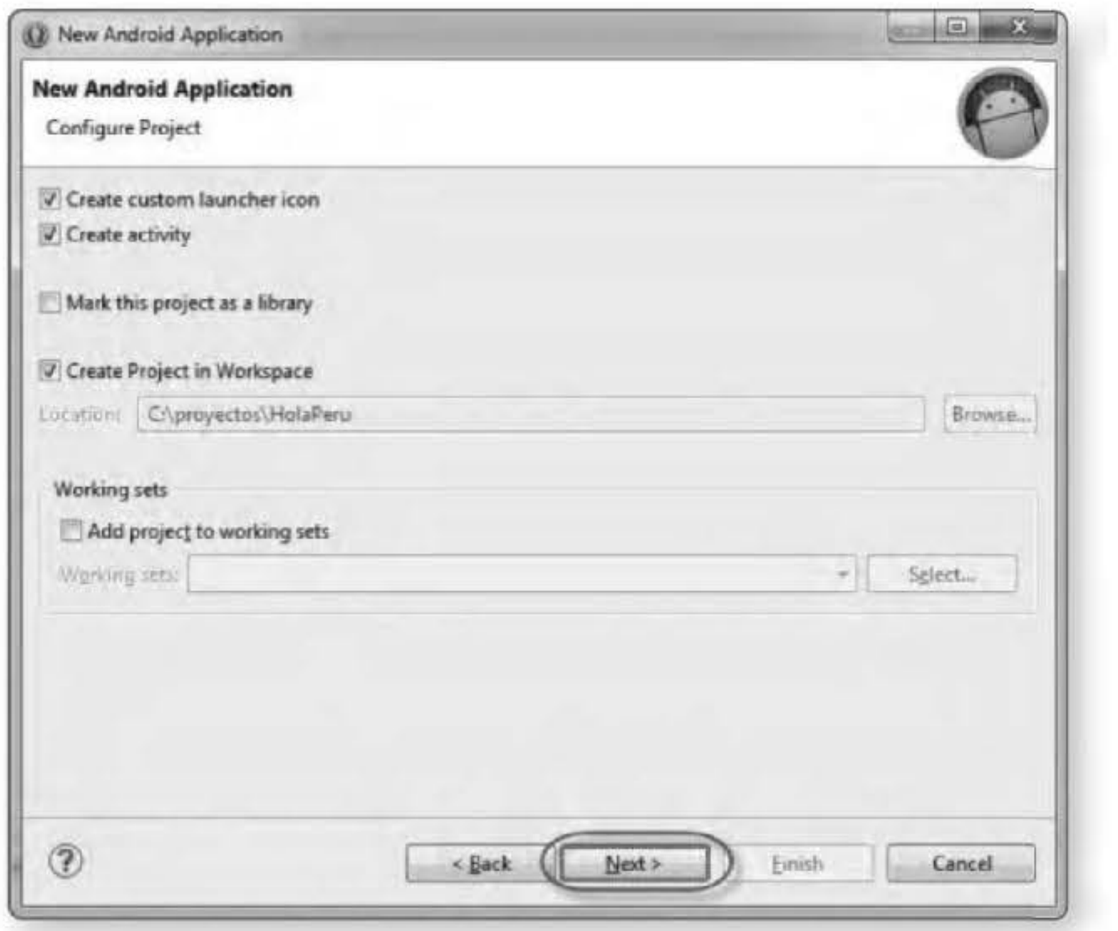
Paso 1: Una vez creada la primera aplicación para Android, cree un nuevo proyecto seleccione ARCHIVO > NUEVO > ANDROID PROYECTO.



Paso 2: Ingrese el nombre del proyecto: EjemploFragmento.



Paso 3: Haga clic en Siguiente (o Next).



Paso 4: Haga clic en Siguiente (o Next).



Nota:

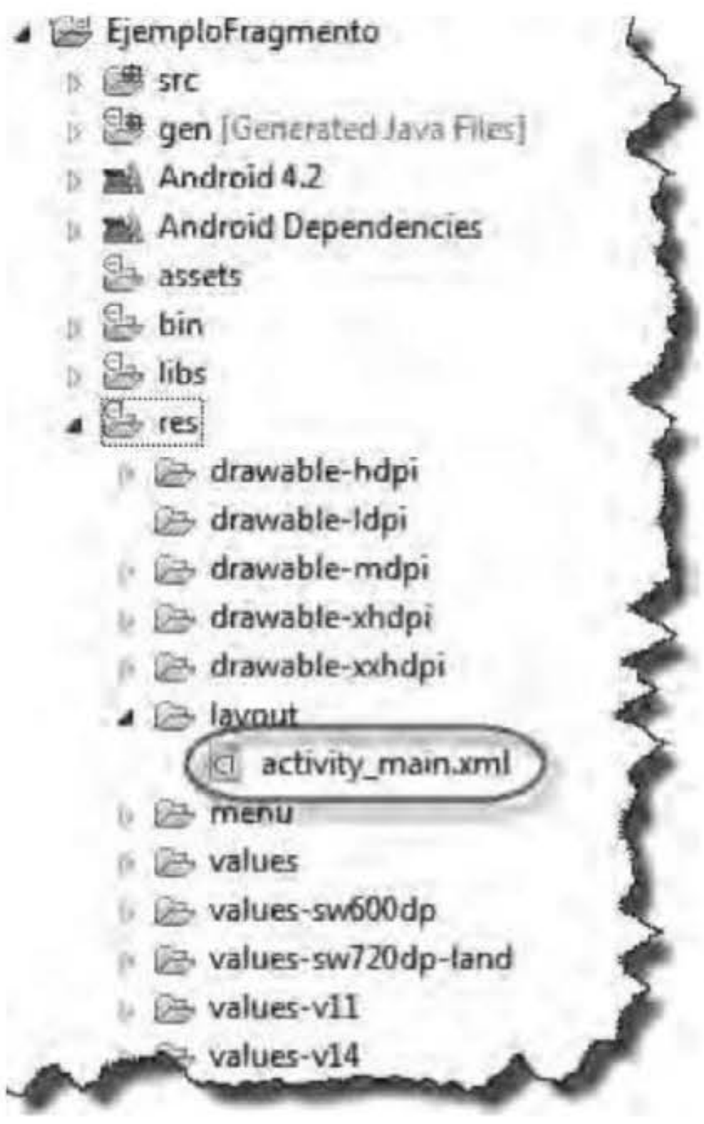
Puede especificar una imagen distinta.

Paso 5: Haga clic en Siguiente (o Next).



Paso 6: Puede escribir un nombre para la actividad o solo presione Finalizar (o Finish).





Paso 7: Ubique la carpeta res/layout.

> ***Nota:***
>
> *Por cada fragmento a utilizar, cree un archivo XML que represente la interfaz de usuario.*

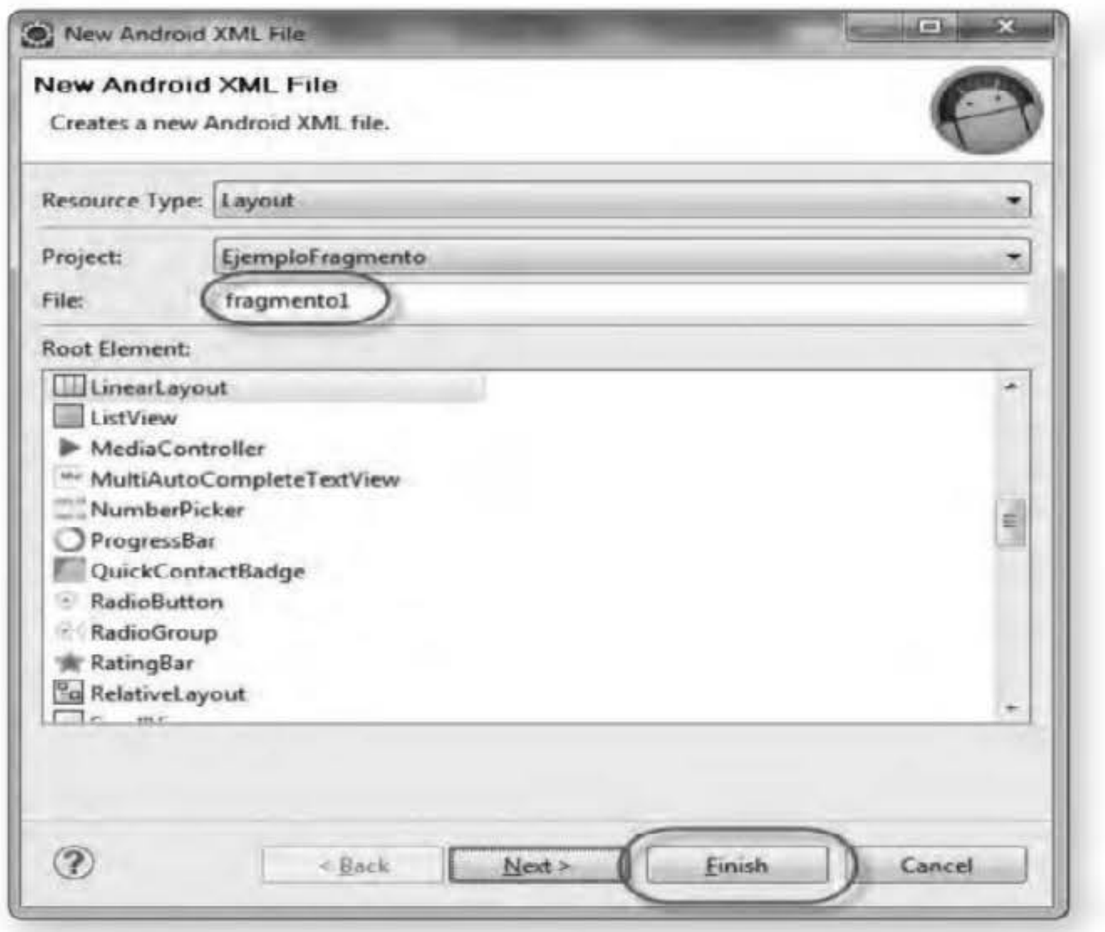
5.3. CREAR FRAGMENTOS

5.3.1. Primer fragmento

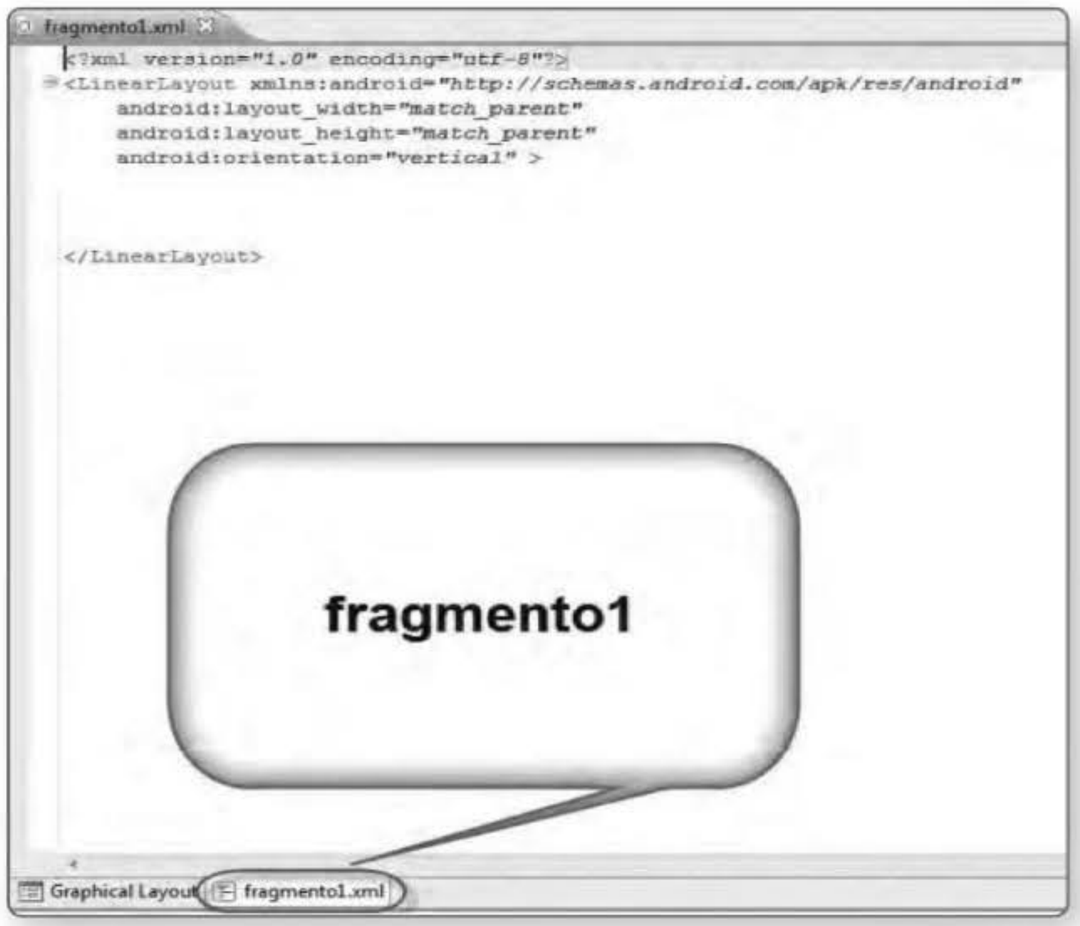
Paso 1: Ingrese a Layout y haga clic derecho; luego diríjase a New > Android XML File.



Paso 2: Ingrese el nombre del archivo (Fragmento1) y luego presione el botón Finish.



Paso 3: Seleccione fragmento1.xml y añada "UI" para cada fragmento.



Paso 4: Utilice una etiqueta para colocar un texto.

```
<TextView
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Primer Fragmento" />
```

fragmento1.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:background="#C00000">
<TextView
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Primer Fragmento" />
</LinearLayout>
```

5.3.2. Segundo fragmento

De la misma manera que se creó el fragmento1.xml, ahora cree el segundo fragmento: fragmento2.xml

fragmento2.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:background="#80FF00">
<TextView
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Segundo Fragmento" />
</LinearLayout>
```

Nota:

Edite el archivo main.xml para definir los fragmentos.

Paso 1: Modifique main.xml.



Paso 2: Defina los fragmentos.

```
*main.xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    >
<fragment
    android:name="com.nolascov.EjemploFragmento.fragmento1"
    android:id="@+id/fragmento1"
    android:layout_weight="1"
    android:layout_width="0px"
    android:layout_height="match_parent"/>
<fragment
    android:name="com.nolascov.EjemploFragmento.fragmento2"
    android:id="@+id/fragmento2"
    android:layout_weight="1"
    android:layout_width="0px"
    android:layout_height="match_parent"/>
</LinearLayout>
```

definicion de fragmentos

Paso 3: Cree una clase por cada fragmento el cual hereda la clase base Fragment (que representa una interfaz de usuario).

Paso 4: Añada dos archivos de clase Java con sus respectivos nombres: Fragmento1.java y Fragmento2. java. Para ello, seleccione la carpeta src > paquete com.nolascov.EjemploFragmento.



Paso 5: Haga clic derecho y seleccione New > Class.



Paso 6: Ingrese el nombre del archivo (fragmento1) y presione Finish.



Paso 7: Edite fragmento1.java.

```
fragmento1.java
package com.nolascov.EjemploFragmento;
import android.app.Fragment;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;


public class fragmento1 extends Fragment {
    @Override
    public View onCreateView(LayoutInflater inflater,ViewGroup container,Bundle savedInstacnceState)
    {
        // Inflate the layout for this fragment
        return inflater.inflate(
            R.layout.fragmento1, container, false);
    }
}
```

Paso 8: De la misma manera que se creó el fragmento1.java , cree fragmento2.java.

```
*fragmento2.java
umize package com.nolascov.EjemploFragmento;
import android.app.Fragment;


public class fragmento2 extends Fragment {
    @Override
    public View onCreateView(LayoutInflater inflater,ViewGroup container,Bundle savedInstacnceState)
    {
        // Inflate the layout for this fragment
        return inflater.inflate(
            R.layout.fragmento2, container, false);
    }
}
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Guarde los cambios realizados, haciendo clic en el ícono*



o presionando la combinación Ctrl+S.

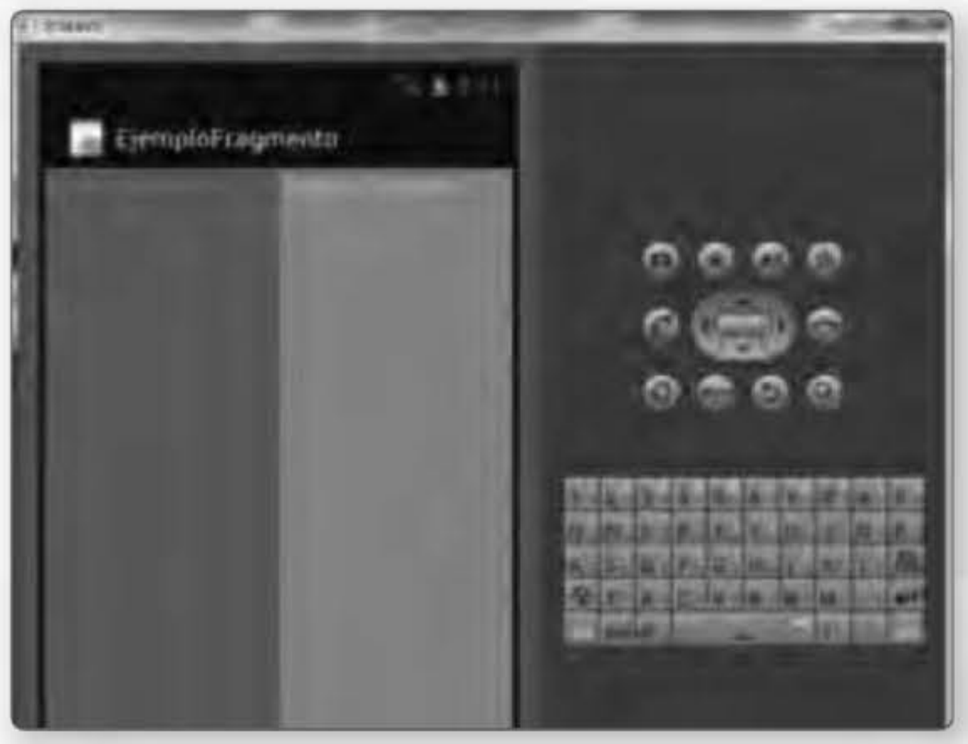
Paso 2: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:*



Paso 3: *Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.



5.4. PRESENTACIÓN DE LOS FRAGMENTOS EN FORMA VERTICAL

Modifique los siguientes archivos, en el archivo mail.xml:



```
main.xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    >
<fragment
    android:name="com.nolascov.EjemploFragmento.fragmento1"
    android:id="@+id/fragmento1"
    android:layout_height="0px"
    android:layout_weight="1"
    android:layout_width="match_parent"/>
<fragment
    android:name="com.nolascov.EjemploFragmento.fragmento2"
    android:id="@+id/fragmento2"
    android:layout_height="0px"
    android:layout_weight="1"
    android:layout_width="match_parent"/>
</LinearLayout>
```

Al ejecutar, se obtiene lo siguiente:



5.5. BARRA DE ACCIONES

La **barra de acciones** es un widget (una aplicación que está siempre activa y visible) para las actividades que sustituye a la tradicional barra de título, en la parte superior de la pantalla. Por defecto, la barra de acción incluye el logotipo de la aplicación en el lado izquierdo, seguido por el título de la actividad y cualquier otro elemento disponible en el menú de opciones, ubicado en el lado derecho.

La barra de acciones ofrece varias funciones útiles, incluyendo la capacidad de:

- *Elementos de la pantalla* del menú de opciones directamente en la barra de acciones, como "puntos de acción" que proporcione acceso instantáneo a las acciones claves de usuario.
- *Elementos de menú* que no aparecen como puntos de acción y se colocan en el menú de desbordamiento, revelada por una lista desplegable en la barra de acciones.
- Proporciona *fichas para desplazarse* entre fragmentos.
- Proporciona una *lista desplegable* para la navegación.
- Proporciona *interactiva* que considera la acción en lugar de puntos de acción (como una caja de búsqueda).

Nota:

Cabe mencionar que la barra de acción fue introducida en Android 3.0.

5.5.1. Añadir elementos a la barra de herramientas



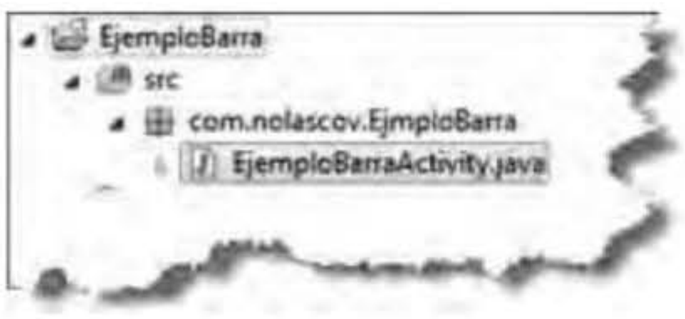
Una vez creada la primera aplicación para Android, cree un nuevo proyecto seleccionando ARCHIVO > NUEVO > ANDROID PROYECTO con el nombre: EjemploBarra (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Un elemento en la barra de acciones es simplemente un elemento de menú que se declara , debe aparecer directamente en la barra de acciones. Un elemento puede incluir un ícono o texto. Si un elemento de menú no aparece como un elemento de la barra de acción entonces el sistema lo coloca en el menú de desbordamiento, que el usuario puede abrir con el ícono del menú en la parte derecha de la barra de acciones.

Cuando la actividad se inicia, el sistema rellena la barra de acciones y un menú de desbordamiento llamando onCreateOptionsMenu () para su actividad. Como se discutió en la orientación menús desarrollador, en este método de devolución de llamada que se define el menú de opciones para la actividad.

Puede especificar un elemento de menú para que aparezca como un elemento de la acción-si no hay sitio para él-de su recurso de menú al declarar android: showAsAction = «ifRoom» para el elemento <item>. De esta manera, el elemento de menú aparece en la barra de acciones para el acceso rápido solo si hay espacio disponible para ello. Si no hay suficiente espacio, el elemento se colocará en el menú de desbordamiento (revelada por el ícono del menú en la parte derecha de la barra de acciones).
También puede declarar un elemento de menú para que aparezca como un elemento de acción a partir de su código de aplicación, llamando setShowAsAction () en el MenuItem y pasando SHOW_AS_ACTION_IF_ROOM.

Si el elemento de menú proporciona tanto un título y un ícono, entonces el elemento de acción solo muestra el ícono de defult. Si desea incluir el texto con el elemento de acción, agregue el «con el texto» la bandera: en XML, añadir withText al androide: atributo showAsAction o, en su código de aplicación, utilice la bandera cuando se llama a SHOW_AS_ACTION_WITH_TEXT setShowAsAction ().
Ubique el archivo : EjemploBarraActivity.java



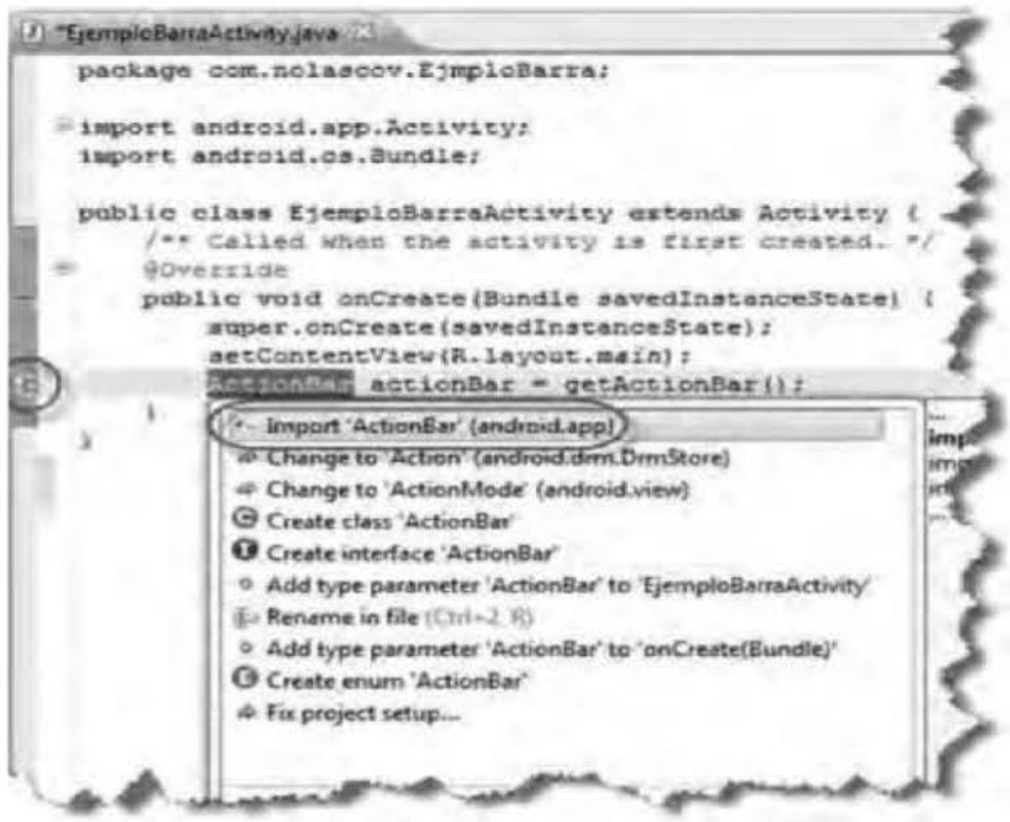
Paso 1: Empiece a modificarlo.



Paso 2: La línea siguiente puede servir para ocultar o mostrar la barra de acciones de la siguiente manera:

```
ActionBar actionBar = getActionBar();
// oculta la barra de acciones
actionBar.hide();
// muestra la barra de acciones
actionBar.show();
```

Paso 3: Añada a la librería:



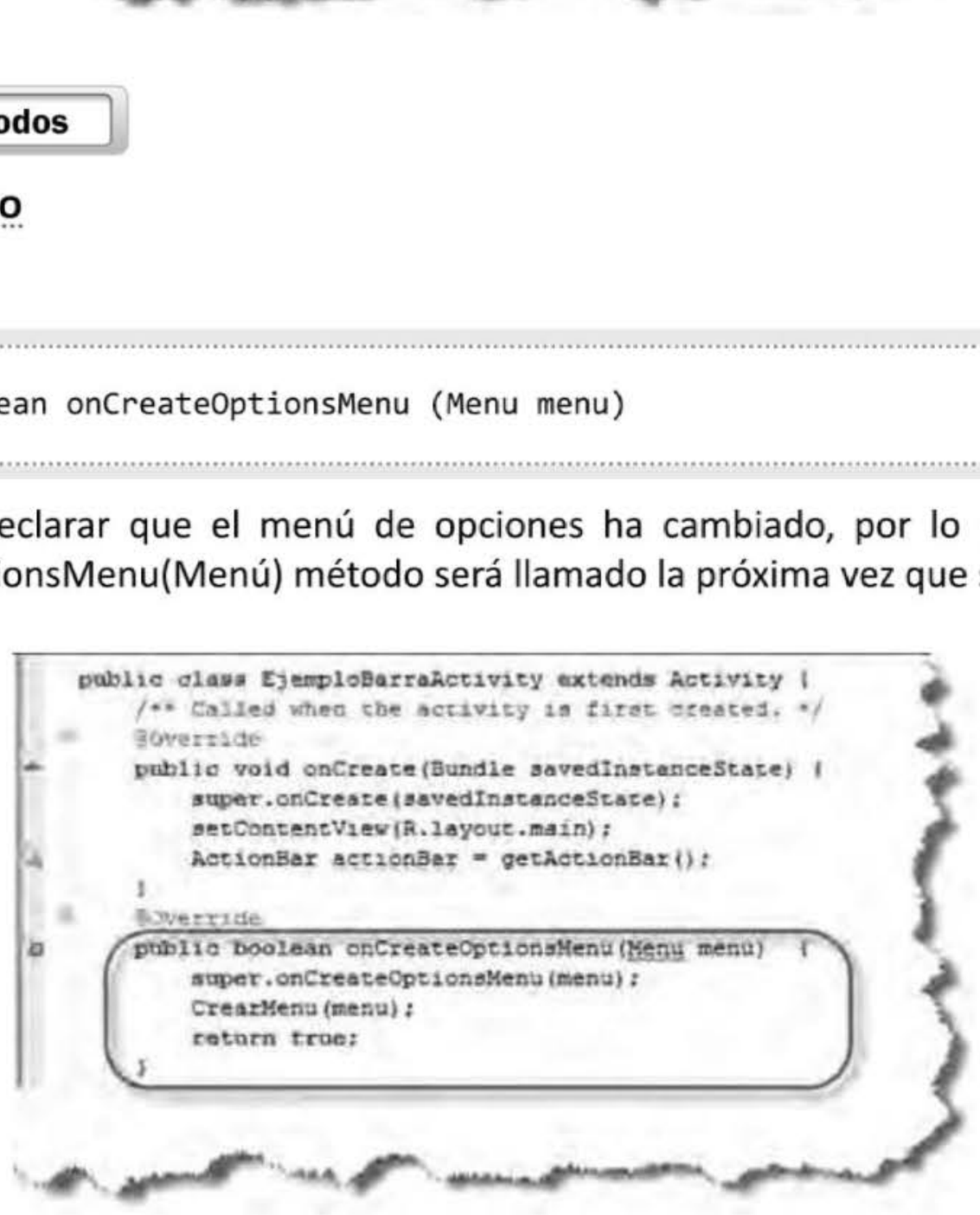
5.5.2. Añadir métodos

A. PRIMER MÉTODO

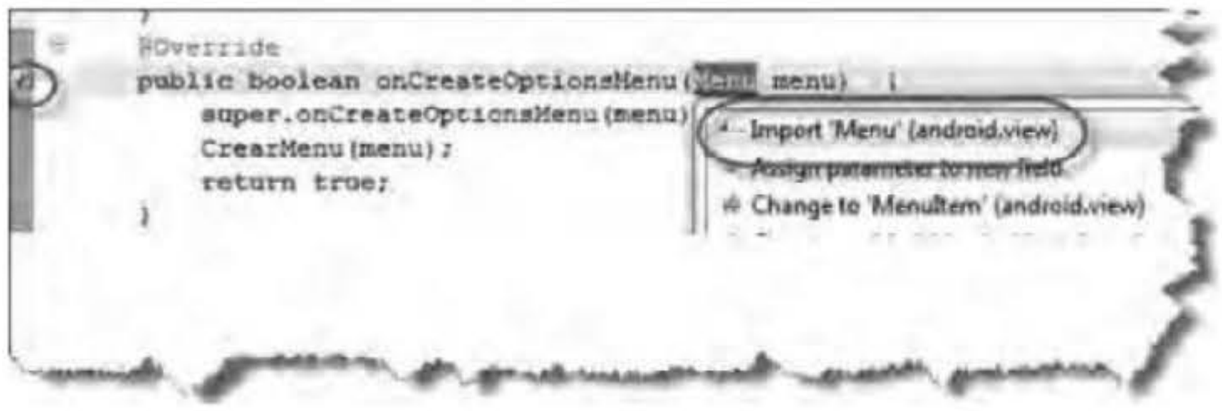
El método :

```
public boolean onCreateOptionsMenu (Menu menu)
```

Se usa para declarar que el menú de opciones ha cambiado, por lo que debe ser recreado. ElonCreateOptionsMenu(Menú) método será llamado la próxima vez que se debe mostrar.



Añada a la librería:



Nota:

Si aparece algún error es porque aún no se ha creado el método CreateMenu.

B. SEGUNDO MÉTODO

```
public boolean onOptionsItemSelected (MenuItem item)
```

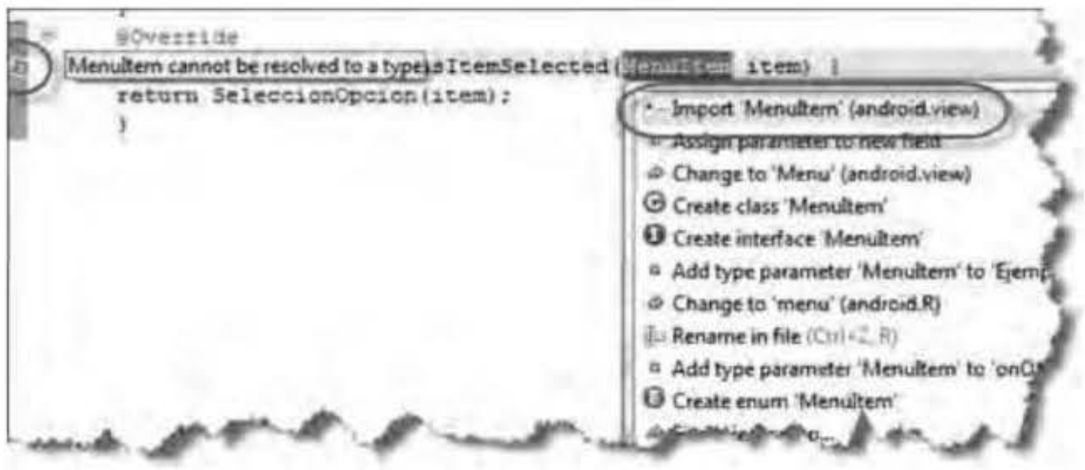
Este método es llamado cada vez que un elemento en el menú de opciones está seleccionado.

- **PARÁMETROS**

Ítem	La opción del menú seleccionado.
Retorno	*Boolean Return false* el proceso normal del menú, true para el uso del mismo.



Añada a la librería.



Nota:

Si aparece algún error es que aun no hemos credo el metodo SeleccionOpcion.

C. TERCER MÉTODO

El método Creación de menú con sus respectivas opciones:

```
private void CreateMenu(Menu menu)
{
    MenuItem Opcion1 = menu.add(0, 0, 0, "Opcion 1" );
    {
        Opcion1.setAlphabeticShortcut('a');
        Opcion1.setIcon(R.drawable.ic_launcher);
    }
    MenuItem Opcion2 = menu.add(0, 1, 1, "Opcion 2" );
    {
        Opcion2.setAlphabeticShortcut('b' );
        Opcion2.setIcon(R.drawable.ic_launcher);
    }
    MenuItem Opcion3 = menu.add(0, 2, 2, "Opcion 3" );
    {
        Opcion3.setAlphabeticShortcut('c' );
        Opcion3.setIcon(R.drawable.ic_launcher);
    }
    MenuItem Opcion4 = menu.add(0, 3, 3, "Opcion 4" );
    {
        Opcion4.setAlphabeticShortcut('d');
        Opcion4.setIcon(R.drawable.ic_launcher);
    }

}
```

- **MÉTODO ADD**

Crea una opción perteneciente a un menú y mediante el método add:

```
public abstract MenuItem add (int groupId, int itemId, int order, int titleRes)
```

✓ groupId:	Identificador un grupo de opciones por lo general es ninguno (0)
✓ itemId:	Identifica a la opción.
✓ order:	El orden que tomará la opción dentro del menú.
✓ titleRes:	El título del menú.

```
MenuItem Opcion1 = menu.add(0, 0, 0, “Opcion 1”);
```

- **METODO SETALPHABETICSHORTCUT**

Método de acceso directo a las opciones.

```
public abstract MenuItem setAlphabeticShortcut (char alphaChar)
```

```
{
      Opcion1.setAlphabeticShortcut('a');
      Opcion1.setIcon(R.drawable.icon);
      }
```

Ahora añada a la librería.



D. CUARTO MÉTODO

Seleccione las opciones del menú:

```
private boolean SeleccionOpcion(MenuItem item)
```

```
private boolean SeleccionOpcion(MenuItem item)
    {
    switch (item.getItemId())
    {
    case 0:
    Toast.makeText(this,  "Seleccionastes la opcion 1",Toast.LENGTH_LONG).show();
    return true;
    case 1:
    Toast.makeText(this,  "Seleccionastes la opcion 2",Toast.LENGTH_LONG).show();
    return true;
    case 2:
    Toast.makeText(this,  "Seleccionastes la opcion 3",Toast.LENGTH_LONG).show();
    return true;
    case 3:
    Toast.makeText(this,  "Seleccionastes la opcion 4",Toast.LENGTH_LONG).show();
    return true;
    case 4:
    Toast.makeText(this,  "Seleccionastes la opcion 5",Toast.LENGTH_LONG).show();
    return true;
    case 5:
    Toast.makeText(this,  "Seleccionastes la opcion 6",Toast.LENGTH_LONG).show();
    return true;
    }
    return false;
    }
```

5.5.3. Uso de notificaciones o toast

Una notificación o toast es un mensaje que aparece en la superficie de la pantalla. La notificación se muestra en la pantalla durante unos segundos al usuario y luego vuelve a desaparecer automáticamente sin requerir ningún tipo de actuación. Aunque son personalizables, aparecen por defecto en la parte inferior de la pantalla, sobre un rectángulo gris ligeramente translúcido. Son ideales para mostrar mensajes rápidos y sencillos al usuario los cuales no son importantes. Seguidamente, se explicará su uso en la aplicación actual:

```
private boolean SeleccionOpcion(MenuItem item)
    {
    switch (item.getItemId())
    {
    case 0:
    Toast.makeText(this,  "Seleccionastes la opcion 1",Toast.LENGTH_LONG).show();
    return true;
    case 1:
    Toast.makeText(this,  "Seleccionastes la opcion 2",Toast.LENGTH_LONG).show();
    return true;
    case 2:
    Toast.makeText(this,  "Seleccionastes la opcion 3",Toast.LENGTH_LONG).show();
    return true;
    case 3:
    Toast.makeText(this,  "Seleccionastes la opcion 4",Toast.LENGTH_LONG).show();
    return true;
    case 4:
    Toast.makeText(this,  "Seleccionastes la opcion 5",Toast.LENGTH_LONG).show();
    return true;
    case 6:
    Toast.makeText(this,  "Seleccionastes la opcion 6",Toast.LENGTH_LONG).show();
    return true;
    }
    return false;
    }
```

Contexto

Mensaje a Mostrar

duracion del mensaje

mostrar el mensaje

Añada a la librería.

```
private boolean SeleccionOpcion(MenuItem item)
    {
    switch (item.getItemId())
    {
    case 0:
    Toast.makeText(this,  "Seleccionastes la opcion 1",Toast.LENGTH_LONG).show();
    return true;
    case 1:
    Toast.makeText(this,  "Seleccionastes la opcion 2",
    return true;
    case 2:
    Toast.makeText(this,  "Seleccionastes la opcion 3",
    return true;
    case 3:
    Toast.makeText(this,  "Seleccionastes la opcion 4",
    return true;
    case 4:
    Toast.makeText(this,  "Seleccionastes la opcion 5",
    return true;
    case 6:
    Toast.makeText(this,  "Seleccionastes la opcion 6",
    return true;
```

Import 'Toast' (android.widget)
Create constant 'Toast'
Create class 'Toast'
Create interface 'Toast'
Create local variable 'Toast'
Change to 'Test' (junit.framework)
Create field 'Toast'
Create enum 'Toast'
Create parameter 'Toast'
Rename in file (Ctrl+2, R)
Fix project setup...

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción RUN AS > ANDROID APPLICATION y observe que el emulador se pondrá en marcha:*



Paso 2: *Si el emulador se bloquea, es necesario deslizar el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.



Paso 3: *Presione el botón Menú y seleccione alguna opción.*



EjemploBarraActivity.java

```
package com.nolascov.EjemploBarra;
import android.app.ActionBar;
import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Toast;
public class EjemploBarraActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
         super.onCreateOptionsMenu(menu);
        CreateMenu(menu);
         return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item)
    {
         return MenuChoice(item);
    }
        private void CreateMenu(Menu menu)
    {
        MenuItem mnu1 = menu.add(0, 0, 0, “Opcion 1” );
        {
            mnu1.setAlphabeticShortcut(‘a’);
            mnu1.setIcon(R.drawable.ic_launcher);
        }
        MenuItem mnu2 = menu.add(0, 1, 1, “Opcion 2” );
        {
            mnu2.setAlphabeticShortcut(‘b’ );
            mnu2.setIcon(R.drawable.ic_launcher);
```

```
        }
        MenuItem mnu3 = menu.add(0, 2, 2, “Opcion 3” );
        {
            mnu3.setAlphabeticShortcut(‘c’ );
            mnu3.setIcon(R.drawable.ic_launcher);
        }
        MenuItem mnu4 = menu.add(0, 3, 3, “Opcion 4” );
        {
            mnu4.setAlphabeticShortcut(‘d’);
        }
            MenuItem mnu5 = menu.add(0, 4, 4, “Opcion 5” );
            {
                mnu4.setAlphabeticShortcut(‘d’);
            }
            MenuItem mnu6 = menu.add(0, 5, 5, “Opcion 6” );
            {
                mnu4.setAlphabeticShortcut(‘d’);
            }
            MenuItem mnu7 = menu.add(0, 6, 6, “Opcion 7” );
            {
                mnu4.setAlphabeticShortcut(‘d’);
            }

    }

    private boolean MenuChoice(MenuItem item)
    {
         switch (item.getItemId()) {
         case 0:
            Toast.makeText (this, “Seleccionastes la opcion 1” ,
                Toast.LENGTH_LONG).show();
            return true;
        case 1:
            Toast.makeText (this, “Seleccionastes la opcion 2” ,
                Toast.LENGTH_LONG).show();
            return true;
        case 2:
            Toast.makeText (this, “Seleccionastes la opcion 3” ,
                Toast.LENGTH_LONG).show();
            return true;
        case 3:
            Toast.makeText (this, “Seleccionastes la opcion 4” ,
                Toast.LENGTH_LONG).show();
            return true;
        case 4:
            Toast. makeText (this, “Seleccionastes la opcion 5” ,
                Toast.LENGTH_LONG).show();
            return true;
         case 5:
            Toast.makeText (this, “Seleccionastes la opcion 6” ,
                Toast.LENGTH_LONG).show();
            return true;
         case 6:
            Toast.makeText (this, “Seleccionastes la opcion 7” ,
                Toast.LENGTH_LONG).show();
            return true;
        }
         return false;
    }
```

5.5.4. Ejemplo Meses

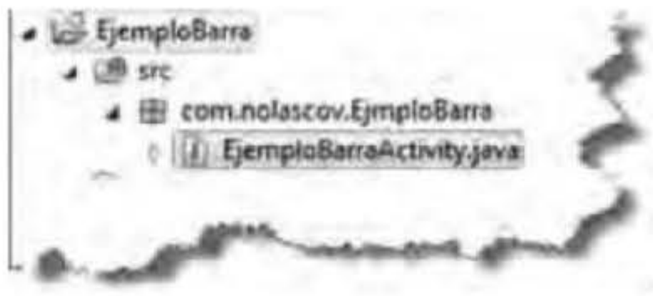


Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando ARCHIVO > NUEVO > ANDROID PROYECTO.

Paso 2: Ingrese el nombre del proyecto: EjemploMeses (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 3: Añada elementos a la barra de herramientas. Considere que un elemento en la barra de acciones es simplemente un elemento de menú que se declara y debe aparecer directamente en la barra de acciones. Un elemento puede incluir un ícono o texto.

Paso 4: Ubique el archivo: EjemploBarraActivity.java.



Paso 5: Modifíquelo.

EjemploMesesActivity.java

```
package com.nolascov.EjemploMeses;
import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Toast;
public class EjemploMesesActivity extends Activity {
  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

  }
  @Override
  public boolean onCreateOptionsMenu(Menu menu) {
    super.onCreateOptionsMenu(menu);
    CrearMenu(menu);
    return true;
  }
  @Override
  public boolean onOptionsItemSelected(MenuItem item) {
  return SeleccionOpcion(item);
  }
  private void CrearMenu(Menu menu) {
    MenuItem Opcion1 = menu.add(0, 0, 0, “Enero”);
    {
    Opcion1.setIcon(R.drawable.ic_launcher);
    }
    MenuItem Opcion2 = menu.add(0, 1, 1, “Febrero”);
    {
    Opcion2.setIcon(R.drawable.ic_launcher);
    }
    MenuItem Opcion3 = menu.add(0, 2, 2,  “Marzo”);
    {
    Opcion3.setIcon(R.drawable.ic_launcher);
    }
```

```
MenuItem Opcion4 = menu.add(0, 3, 3, “Abril”);
{
Opcion4.setIcon(R.drawable.ic_launcher);
}

MenuItem Opcion5 = menu.add(0, 4, 4, “Mayo”);
{
Opcion5.setIcon(R.drawable.ic_launcher);
}

MenuItem Opcion6 = menu.add(0, 5, 5, “Junio”);
{
Opcion6.setIcon(R.drawable.ic_launcher);
}

MenuItem Opcion7 = menu.add(0, 6, 6, “Julio”);
{
Opcion7.setIcon(R.drawable.ic_launcher);
}

MenuItem Opcion8 = menu.add(0, 7, 7, “Agosto”);
{
Opcion8.setIcon(R.drawable.ic_launcher);
}

MenuItem Opcion9 = menu.add(0, 8, 8, “Setiembre”);
{
Opcion6.setIcon(R.drawable.ic_launcher);
}
   MenuItem Opcion10 = menu.add(0, 9, 9, “Octubre”);
   {
   Opcion10.setIcon(R.drawable.ic_launcher);
   }
   MenuItem Opcion11 = menu.add(0, 10, 10, “Noviembre”);
   {
   Opcion10.setIcon(R.drawable.ic_launcher);
   }

   MenuItem Opcion12 = menu.add(0, 11, 11, “Diciembre”);
   {
   Opcion11.setIcon(R.drawable.ic_launcher);
   }
                                                              }
private boolean SeleccionOpcion(MenuItem item)
   {
   switch (item.getItemId())
   {
   case 0:
   Toast.makeText(this, “Enero”,Toast.LENGTH_LONG).show();
   return true;
   case 1:
   Toast.makeText(this, “Febrero”,Toast.LENGTH_LONG).show();
   return true;
   case 2:
   Toast.makeText(this, “Marzo”,Toast.LENGTH_LONG).show();
   return true;
   case 3:
   Toast.makeText(this, “Abril”,Toast.LENGTH_LONG).show();
   return true;
   case 4:
   Toast.makeText(this, “Mayo”,Toast.LENGTH_LONG).show();
```

```
        return true;
        case 5:
        Toast.makeText(this, “Junio”,Toast.LENGTH_LONG).show();
        return true;
        case 6:
        Toast.makeText(this, “Julio”,Toast.LENGTH_LONG).show();
    return true;
        case 7:
        Toast.makeText(this, “Agosto”,Toast.LENGTH_LONG).show();
        return true;
        case 8:
        Toast.makeText(this, “Setiembre”,Toast.LENGTH_LONG).show();
        return true;
        case 9:
        Toast.makeText(this, “Octubre”,Toast.LENGTH_LONG).show();
        return true;
        case 10:
        Toast.makeText(this, “Noviembre”,Toast.LENGTH_LONG).show();
        return true;
        case 11:
        Toast.makeText(this, “Diciembre”,Toast.LENGTH_LONG).show();
    return true;
        }
        return false;
        }
    }
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción RUN AS > ANDROID APPLICATION y observe que el emulador se pondrá en marcha:*



Paso 2: *Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.



Paso 3: *Presione el botón Menú y seleccione alguna opción.*



5.5.5. Ejemplo Alimentos

Diseñe una aplicación donde muestre un listado de algunos alimentos aparezca y se mostrará el ícono de uno de ellos con su descripción.

Paso 1: Una vez creada la primera aplicación para Android, cree un nuevo proyecto seleccione ARCHIVO > NUEVO > ANDROID PROYECTO.

Paso 2: Ingrese el nombre del proyecto: EjemploAlimentos.

EjemploAlimentosActivity.java

```
package com.nolascov.EjemploAlimentos;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Toast;

public class EjemploAlimentosActivity extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu)  {
     super.onCreateOptionsMenu(menu);
     CrearMenu(menu);
     return true;
     }
     @Override
     public boolean onOptionsItemSelected(MenuItem item) {
     return SeleccionOpcion(item);
     }
     private void CrearMenu(Menu menu) {
     MenuItem Opcion1 = menu.add(0, 0, 0, “Arroz”);
     {
     Opcion1.setAlphabeticShortcut(‘a’);
     Opcion1.setIcon(R.drawable.ic_launcher);
     Opcion1.setShowAsAction(MenuItem.SHOW_AS_ACTION_IF_ROOM     |     MenuItem.SHOW_AS_
ACTION_WITH_TEXT);
     }

     MenuItem Opcion2 = menu.add(0, 1, 1, “Brocoli”);
     {
     Opcion2.setAlphabeticShortcut(‘b’);
     Opcion2.setIcon(R.drawable.ic_launcher);
     }

     MenuItem Opcion3 = menu.add(0, 2, 2,   “Limon”);
     {
     Opcion3.setAlphabeticShortcut(‘c’);
     //Opcion3.setIcon(R.drawable.limon);
     }
```

```
MenuItem Opcion4 = menu.add(0, 3, 3, “Manzana”);
{
Opcion4.setAlphabeticShortcut(‘d’);
Opcion4.setIcon(R.drawable.ic_launcher);
}

                                                                    }
private boolean SeleccionOpcion(MenuItem item)
{
switch (item.getItemId())
    {
    case android.R.id.home:
    Toast.makeText(this,  “Presione Clic en el icono”,Toast.LENGTH_LONG).show();
    Intent i = new Intent(this, EjemploAlimentosActivity.class);
        i.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
startActivity(i);
    return true;
    case 0:
        Toast.makeText(this,  “Seleccionastes Arroz”,Toast.LENGTH_LONG).show();
        return true;
    case 1:
    Toast.makeText(this,  “Seleccionastes Brocoli”,Toast.LENGTH_LONG).show();
    return true;
    case 2:
    Toast.makeText(this,  “Seleccionastes Limon”,Toast.LENGTH_LONG).show();
    return true;
    case 3:
    Toast.makeText(this,  “Seleccionastes Manzana”,Toast.LENGTH_LONG).show();
    return true;
        }
    return false;
    }
}
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:*



Paso 2: *Si el emulador se bloquea, es necesario deslizar el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.



Paso 3: *Presione el botón Menú y seleccione alguna opción.*

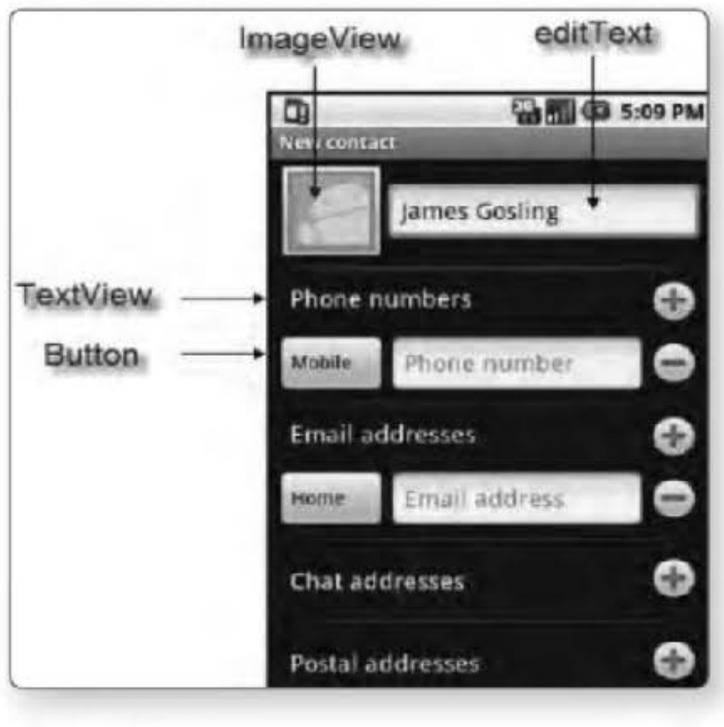


CAP.

6

Interfaz de usuario

6.1. VISTA (O VIEW)



Una vista es un área rectangular en la pantalla que gestiona el tamaño, el dibujado, el cambio de foco y los gestos del área que representan.
La clase android.view. View sirve de clase base para todos los widgets como: TextView, EditText, Button, RadioButton, Checkbox, DatePicker TimePicker, Spinner.

6.1.1. ViewGroup



Una vista especial que contiene otras vistas hijas. ViewGroup es la clase base los layouts y vistas contenedoras.

Esta clase también define la clase ViewGroup.LayoutParams.

Layouts	AbsoluteLayout, TableLayout, LinearLayout, RelativeLayout.
ViewGroups	DatePicker, Gallery, GridView ListView, ScrollView, Spinner, TabWidget.

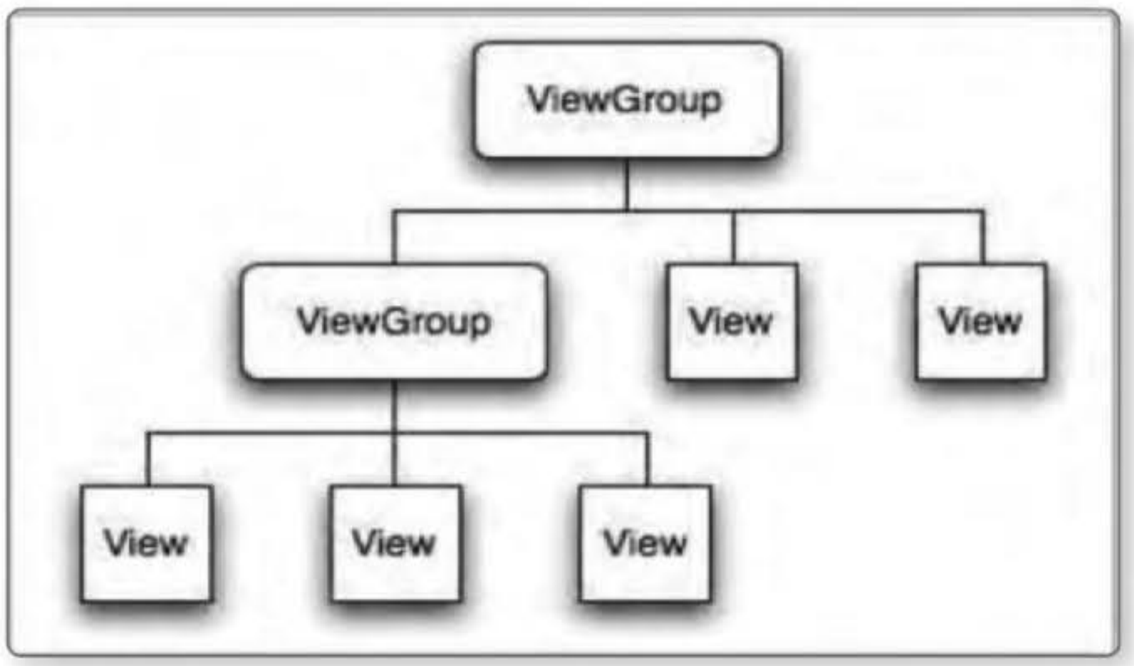
6.1.2. Organización de las vistas

Las vistas se organizan en estructuras de árbol cuya raíz es un ViewGroup.

setContentView() permite añadir una vista a una actividad.

La plataforma Android ofrece dos métodos para diseñar la interfaz:

- procedural (código)
- declarativa (XML)




```
public void setContentView(View v) {}
public void setConentView(View v, LayoutParams p) {}
public void setConentView(int layoutResID)
```

6.2. LAYOUT



Un layout es un contenedor invisible que determina la disposición de las vistas en la pantalla. Todos los layouts heredan de ViewGroup.

Android recomienda definir el layout en XML mediante archivos de layout que se encuentran en res/layout/ main_layout.xml. Cuando se compila la aplicación, se compila también cada archivo XML de layout y queda accesible desde la clase R "View Resource", generada por Android.

6.2.1. Tipos de layout

Los layouts son objetos derivados de la clase ViewGroup y determinan la posición que tendrán los Views en la pantalla. A continuación, se detallará los tipos de layouts que proporciona Android y su uso:

A. FRAMELAYOUT

Permite cambiar dinamicamente los controles en el layout (está pensado para ubicar un solo elemento). Este es el layout más simple que consiste en un marco que ocupa toda la pantalla. Los Views hijos que se quieran agregar se posicionarán en la esquina superior izquierda por lo que si añade varios de ellos, algunos quedarán ocultos sobre otros. Este layout puede servir por ejemplo para mostrar una simple imagen.



B. LINEARLAYOUT

Alinea los hijos horizontal o verticalmente. Los Views hijos serán alineados de forma lineal uno trás de otro, ya sea horizontal o verticalmente, según se especifique en el atributo la orientación en el fichero XML. Además, se puede modificar otros atributos para organizar los Views como por ejemplo sus márgenes, gravedad, etc.



C. TABLELAYOUT

Dispone los elementos en filas y columnas. En este layout, los hijos son posicionados en filas y columnas exactamente de la misma forma que una tabla. Algunas de las celdas pueden estar vacías, contener Views o incluso ViewGroups.



D. RELATIVELAYOUT



Dispone de unos elementos relativos a los otros. En este tipo de layouts, se cuenta con varias propiedades para posicionar cada control en la pantalla, con respecto a su View padre o a otro de los Views que se tenga en la interfaz. Los elementos serán visualizados en el orden dado; de esta forma, si el primero de ellos está centrado en la pantalla, los elementos siguientes estarán alineados con respecto al centro de la pantalla.

E. ABSOLUTELAYOUT



Dispone los elementos en coordenadas exactas (X , Y).

La clase Layout contiene una clase LayoutParams que es especializada por cada tipo de Layout: LinearLayoutParams, RelativeLayoutParams, etc.

6.3. TIPOS DE BOTONES

En este apartado, se va a tratar sobre los diferentes tipos de botones y cómo poder personalizarlos. El SDK de Android proporciona tres tipos de botones: el clásico (Button), el de tipo on/off (ToggleButton) y el que puede contener una imagen (ImageButton).

6.3.1. Button



Un control de tipo Button es el botón más básico que se puede utilizar. En el ejemplo siguiente., se definirá un botón con el texto “Púlsame”, asignando su propiedad android:text. Además de esta propiedad, se podría utilizar muchas otras como el color de fondo (android:background), estilo de fuente (android:typeface), color de fuente (android:textcolor), tamaño de fuente (android:textSize), etc.

A.

A. PROPIEDADES

```
<Button android:id=”@+id/Button1”
 android:layout_width=”wrap_content”
 android:layout_height=”wrap_content”
 android:textColor=”#ff0000”
 android:text=”Button”/>
```

B. EJEMPLOCONTROLES

Ahora diseñe una aplicación donde se muestre el uso de los diferente controles.

Paso 1: Una vez creada la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android* proyecto con el nombre de proyecto: EjemploControles (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 2: Edite el archivo main.xml y añada el botón.

*main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    />
<Button android:id="@+id/BtnBoton1"
    android:text="Presionar"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
</LinearLayout>
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.*

Paso 2: *Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.



6.3.2. ToggleButton (on/off)

Un control de tipo ToggleButton es un tipo de botón que puede permanecer en dos estados, pulsando "/no pulsado". En este caso, en lugar de definir un solo texto para el control se definirá dos textos, dependiendo de su estado. Así, se puede asignar las propiedades android:textOn y android:textoOff para definir ambos textos. Vea el siguiente ejemplo:

```
<ToggleButton android:id="@+id/BtnBoton2"
  android:textOn="ON"
  android:textOff="OFF"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content" />
```

Edite el archivo main.xml y añada el botón.



EJECUTANDO LA APLICACIÓN

Paso 1: Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.

Paso 2: Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.



La figura muestra la aplicación que se ejecuta en el emulador de Android.



6.3.3. ImageButton

En un control de tipo ImageButton, se puede definir una imagen a mostrar; en lugar de un texto. Para ello, debe asignar la propiedad android:src. Normalmente, se asigna esta propiedad con el descriptor de algún recurso que se haya incluido en la carpeta /res/drawable. Así, por ejemplo, en este caso, se incluirá una imagen llamada "ok.png" que hará referencia, entonces, al recurso "@drawable/ok".



EJECUTANDO LA APLICACIÓN

Paso 1: Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.

Paso 2: Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.



La figura muestra la aplicación que se ejecuta en el emulador de Android.



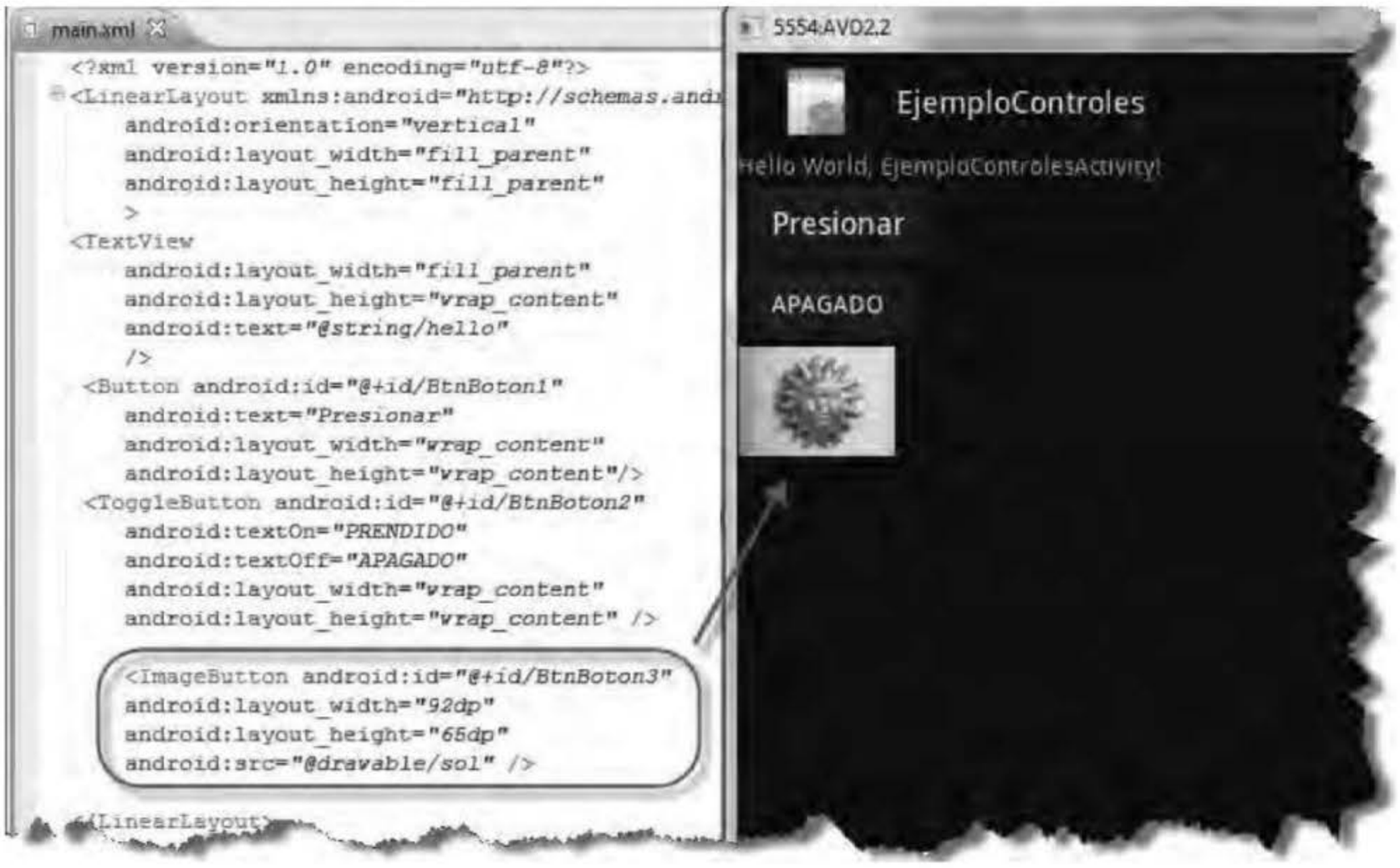
6.3.3. ImageButton

En un control de tipo ImageButton, se puede definir una imagen a mostrar; en lugar de un texto. Para ello, debe asignar la propiedad android:src. Normalmente, se asigna esta propiedad con el descriptor de algún recurso que se haya incluido en la carpeta /res/drawable. Así, por ejemplo, en este caso, se incluirá una imagen llamada "ok.png" que hará referencia, entonces, al recurso "@drawable/ok".



```
<ImageButton android:id="@+id/BtnBoton3"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:src="@drawable/ok" />
```

Edite el archivo main.xml y añada el botón.



EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.*

Paso 2: *Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.



6.4. VISTAS DE TEXTO

6.4.1. TextView

El control TextView es otro de los clásicos en la programación de GUI. Las etiquetas de texto se utilizan para mostrar un determinado texto al usuario. Al igual que en el caso de los botones, el texto del control se establece mediante la propiedad android:text. Aparte de esta propiedad, la naturaleza del control hace que las más interesantes sean las que establecen el formato del texto mostrado. Igual que en el caso de los botones presenta las siguientes propiedades:

android:background	Color de fondo
android:textColor	Color del texto
android:textSize	Tamaño de la fuente
android:typeface	Estilo del texto (negrita, cursiva, etc.)

```
TextView android:id="@+id/LblEtiqueta"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Este es unTexto"
android:background="#AA44FF"
android:typeface="monospace" />
```

De igual forma, también se puede manipular estas propiedades desde su código. Como ejemplo, en el siguiente fragmento, se recupera el texto de una etiqueta con getText() y luego se concatena unos números. Actualice su contenido mediante setText() y cámbie su color de fondo con setBackgroundColor().

```
final TextView lblEtiqueta = (TextView)findViewById(R.id.LblEtiqueta);
String texto = lblEtiqueta.getText().toString();
texto += "123";
lblEtiqueta.setText(texto);
```

6.4.2. EditText



El control EditText es el componente de edición de texto que proporciona la plataforma Android. Permite la introducción y edición de texto por parte del usuario.

A. EJEMPLO

```
<EditText android:id="@+id/TxtTexto"
 android:layout_width="fill_parent"
 android:layout_height="wrap_content"
 android:layout_below="@id/LblEtiqueta"/>
```

main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.and
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    />
 <Button android:id="@+id/BtnBoton1"
    android:text="Presionar"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
 <ToggleButton android:id="@+id/BtnBoton2"
    android:textOn="PRENDIDO"
    android:textOff="APAGADO"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />

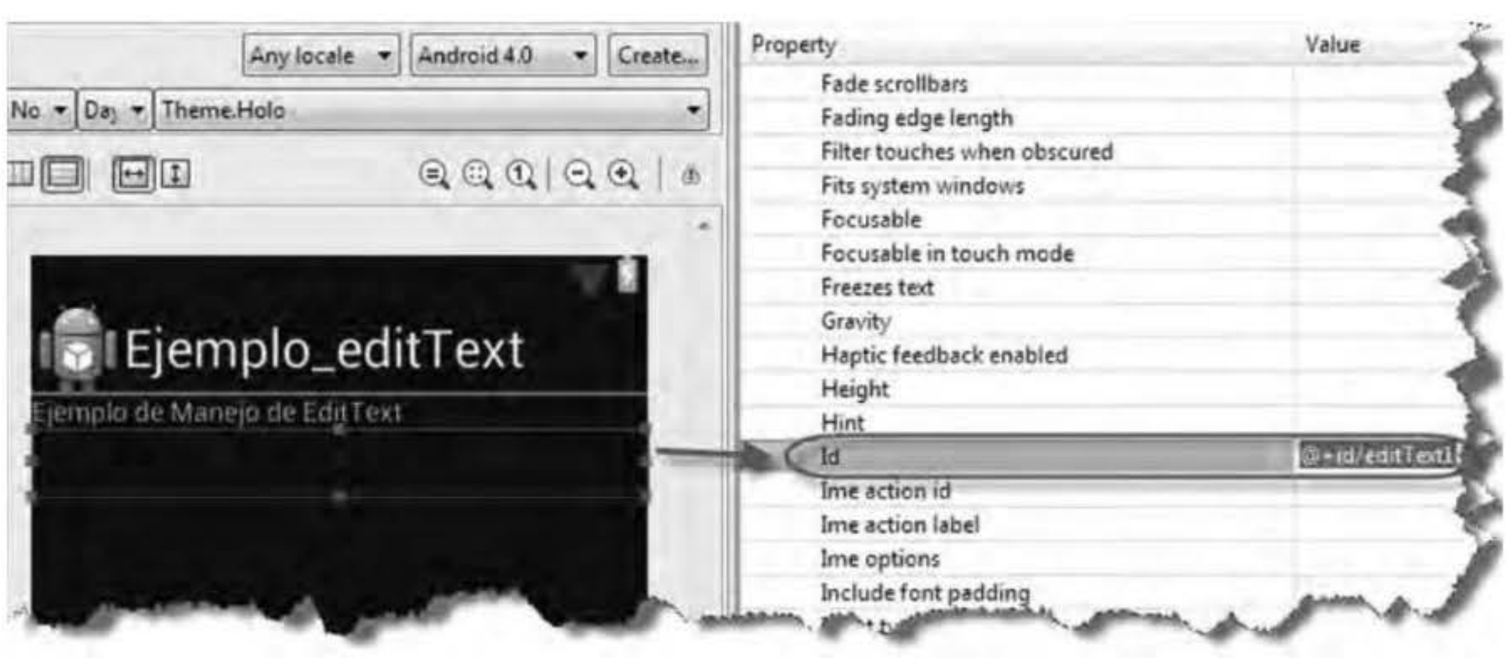
 <ImageButton android:id="@+id/BtnBoton3"
    android:layout_width="92dp"
    android:layout_height="65dp"
    android:src="@drawable/sol" />

<EditText android:id="@+id/TxtTexto"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"/>
</LinearLayout>
```



B. PROPIEDADES DEL EDITTEXT

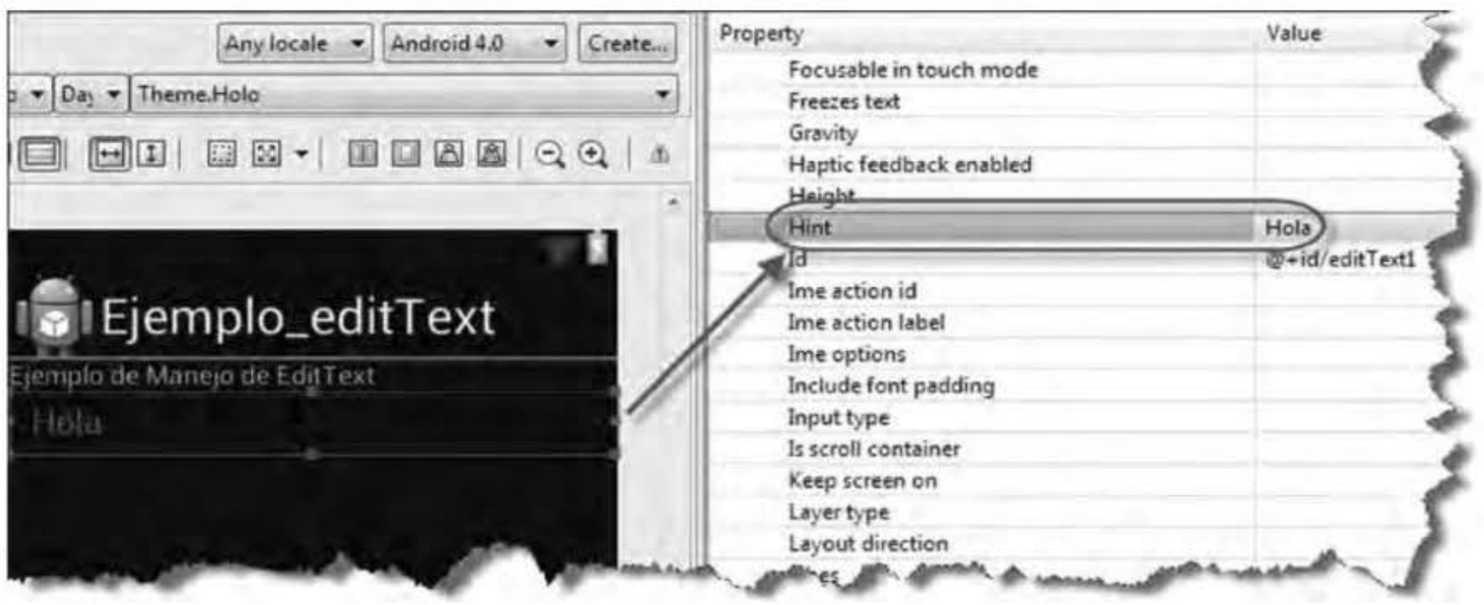
- *Id:* Al igual que los demás controles, esta es la principal propiedad al momento de programar y lo cual permitirá tener una referencia fácilmente del control desde código.



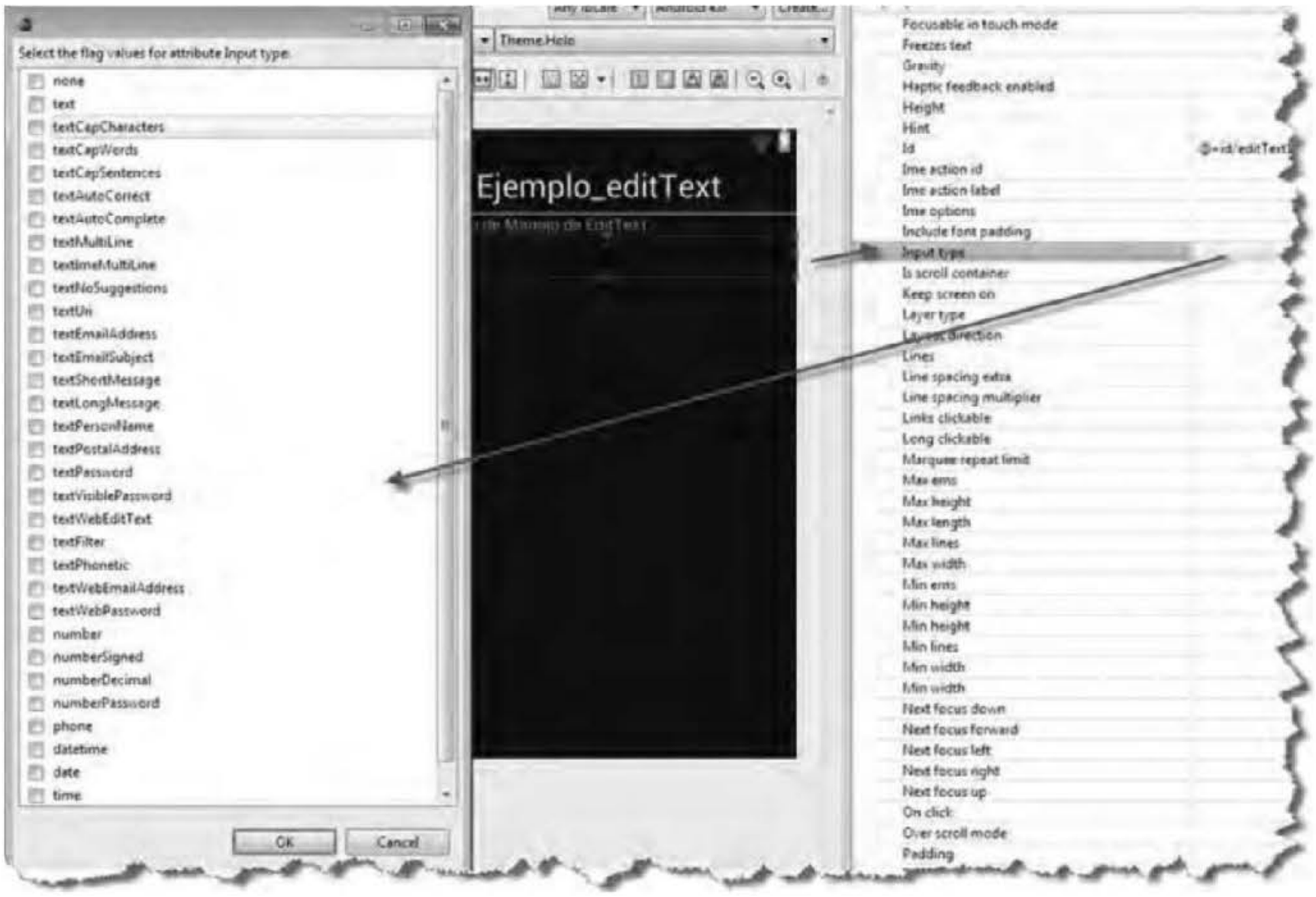
- *Text:* Esta propiedad permite especificar el texto visible. Dentro del control funciona como un valor por defecto del control que resulta algunas veces en este tipo de control, pues cuando el usuario quiera digitar un valor, tendría que borrar primero todo lo que tiene antes de escribir el valor deseado.



- *Hint:* Esta propiedad permite colocar un texto al control como se hace con la propiedad text. La diferencia radica en que esta funciona como marca de agua y desaparece apenas el usuario entre al cursor. Es muy recomendable para optimizar espacio.



- *inputType:* Esta propiedad permite formatear el texto de entrada y el tipo de teclado que se mostrará cuando se está escribiendo sobre este control.



6.5. EJEMPLOBOTONES

Diseñe una aplicación que permita elegir un sistema operativo de preferencia.

Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto*.

Paso 2: Ingrese el nombre del proyecto: EjemploBotones (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 3: Edite el archivo activity_main.xml , añada el botón y el EditText :



```
<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" >
    <EditText
        android:id="@+id/editElejido"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentTop="true"
        android:ems="10"
        android:hint="Seleccione su Sistema Operativo "
        android:inputType="textPersonName" >
        <requestFocus />
    </EditText>
    <Button
        android:id="@+id/btnAndroid"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/editElejido"
        android:layout_marginTop="20dp"
        android:text="Sistema Operativo Android" />
    <Button
        android:id="@+id/btnIphone"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/btnAndroid"
        android:layout_below="@+id/btnAndroid"
        android:layout_marginTop="19dp"
        android:text="Sistema Operativo Iphone" />
    <Button
        android:id="@+id/btnSymbian"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/btnIphone"
        android:layout_marginTop="16dp"
        android:text="Sistema Operativo Symbian" />
    <Button
        android:id="@+id/btnRim"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/btnSymbian"
        android:layout_marginTop="24dp"
        android:text="Sistema Operativo RIM" />
    <Button
        android:id="@+id/btnWindow"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/btnRim"
        android:layout_marginTop="26dp"
        android:text="Sistema Operativo Window Mobile" />
</RelativeLayout>
```

6.6. EVENTOS DE LOS BOTONES

Los controles pueden lanzar muchos otros eventos. Entre ellos, el más común y que se quiere capturar, en la mayoría de las ocasiones, es el evento onClick. Para definir la lógica de este evento, implemente definiendo un nuevo objeto View.OnClickListener() y asócielo al botón mediante el método setOnClickListener(). Para ello, añada esta funcionalidad a su aplicación:

```
package com.nolascov.ejemplobotones
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
public class MainActivity extends Activity implements OnClickListener {
    EditText editElejido;
    Button btnAndroid,btnIphone,btnRim,btnSymbian,btnWindow;
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        editElejido = (EditText) findViewById(R.id.editElejido);
        btnAndroid  =  (Button) findViewById(R.id.btnAndroid);
        btnAndroid.setOnClickListener(this);
        btnIphone  =  (Button) findViewById(R.id.btnIphone);
        btnIphone.setOnClickListener(this);
        btnRim  =  (Button) findViewById(R.id.btnRim);
        btnRim.setOnClickListener(this);
        btnSymbian  =  (Button) findViewById(R.id.btnSymbian);
        btnSymbian.setOnClickListener(this);
        btnWindow  =  (Button) findViewById(R.id.btnWindow);
        btnWindow.setOnClickListener(this);
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.activity_main, menu);
        return true;
    }
    public void onClick(final View clicked)  {
            switch (clicked.getId()) {
            case R.id.btnAndroid:
                    editElejido.setText("Sistema Operativo Android Google");
                    break;
            case R.id.btnIphone:
                    editElejido.setText("Sistema Operativo Iphone");
                    break;
            case R.id.btnRim:
                    editElejido.setText("Sistema Operativo Rim");
                    break;
            case R.id.btnSymbian:
                    editElejido.setText("Sistema Operativo Symbian");
                    break;
            case R.id.btnWindow:
                    editElejido.setText("Sistema Operativo Windows Mobile");
                    break;
            }
    }
}
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:*

Paso 2: *Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.



6.7. EJEMPLOMAIL



En este apartado, se explicará la manera cómo diseñar una aplicación que permita el envío de e-mails.

Paso 1: Una vez creada la primera aplicación para Android, cree un nuevo proyecto seleccione *Archivo > Nuevo > Android proyecto*.

Paso 2: Ingrese el nombre del proyecto: EjemploControles2 (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 3: Edite el archivo activity_main.xml y añada el botón.

Paso 4: Añada etiquetas (TextView) y textos editables (EditText):

activity_main.xml

```
<RelativeLayout xmlns:tools="http://schemas.android.com/tools"
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >
    <EditText
        android:id="@+id/destino"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_alignParentTop="true"
        android:ems="10"
        android:inputType="textEmailAddress" >
        <requestFocus />
    </EditText>
    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/destino"
        android:layout_alignBottom="@+id/destino"
        android:text="Destino" />
    <Button
        android:id="@+id/envio"
        style="?android:attr/buttonStyleSmall"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:text="Enviar Correo" />

    <EditText
        android:id="@+id/mensaje"
        android:layout_width="wrap_content"
        android:layout_height="120dp"
        android:layout_above="@+id/envio"
        android:layout_alignParentRight="true"
        android:layout_marginBottom="41dp"
        android:ems="10"
        android:inputType="textMultiLine" />
```

```
    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignTop="@+id/mensaje"
        android:text="Mensaje" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignRight="@+id/textView3"
        android:layout_below="@+id/destino"
        android:layout_marginTop="52dp"
        android:text="Asunto" />

    <EditText
        android:id="@+id/asunto"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/textView2"
        android:layout_alignBottom="@+id/textView2"
        android:layout_alignParentRight="true"
        android:ems="10"
        android:inputType="textPersonName" />

</RelativeLayout>
```

EJECUTANDO LA APLICACIÓN

Paso 1: Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción *Run As > Android Application* y observe que el emulador se pondrá en marcha:



Paso 2: Eventos del botón.

```
package com.inkadroid.ejemplomail;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
public class MainActivity extends Activity  implements OnClickListener  {
    private Button envio;
    private EditText destino;
    private EditText asunto;
    private EditText mensaje;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);
            this.envio = (Button) findViewById(R.id.envio);
            this.destino = (EditText) findViewById(R.id.destino);
            this.asunto = (EditText) findViewById(R.id.asunto);
            this.mensaje = (EditText) findViewById(R.id.mensaje);
            envio.setOnClickListener(this);
     }
             public void onClick(final View clicked)  {

                    Intent i = new Intent(android.content.Intent.ACTION_SEND);
                    i.setType("plain/text");
                    i.putExtra(android.content.Intent.EXTRA_EMAIL,       new
String[]{destino.getText().toString()});
                    i.putExtra(android.content.Intent.EXTRA_SUBJECT,  asunto.
getText().toString());
                    i.putExtra(android.content.Intent.EXTRA_TEXT,    mensaje.
getText().toString());
                    startActivity(i);
                }
}}
```

6.8. EJEMPLO_OPERACIONES



En este apartado, se diseñará una aplicación que permita realizar cálculos básicos de dos números.

Paso 1: Una vez creada la primera aplicación para Android, cree un nuevo proyecto seleccionando Archivo > Nuevo > Android proyecto con el nombre de proyecto: Ejemplo_Operaciones (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 2: Edite el archivo activity_main.xml y añada el botón.

Paso 3: Añada etiquetas (TextView), textos editables (EditText) y algunos Button:

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" >

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginLeft="50dp"
        android:layout_marginTop="33dp"
        android:text="Numero1" />

    <EditText
        android:id="@+id/txtNumero1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textView1"
        android:layout_below="@+id/textView1"
        android:layout_marginTop="24dp"
        android:ems="10"
        android:hint="Numero1"
        android:inputType="number" >

        <requestFocus />
    </EditText>

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignRight="@+id/textView1"
        android:layout_below="@+id/txtNumero1"
        android:layout_marginTop="41dp"
        android:text="Numero2" />

    <EditText
        android:id="@+id/txtNumero2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textView2"
        android:layout_centerVertical="true"
        android:ems="10"
        android:hint="Numero2"
        android:inputType="number" />

    <TextView
        android:id="@+id/txtResultado"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignRight="@+id/textView2"
        android:layout_below="@+id/txtNumero2"
        android:layout_marginTop="41dp"
        android:text="Resultado" />
```

```
    <Button
        android:id="@+id/Suma"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/txtResultado"
        android:layout_below="@+id/txtResultado"
        android:layout_marginTop="33dp"
        android:text="+"
        android:textSize="30dp" />

    <Button
        android:id="@+id/Resta"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/Suma"
        android:layout_alignBottom="@+id/Suma"
        android:layout_toRightOf="@+id/Suma"
        android:text="-"
        android:textSize="30dp" />

    <Button
        android:id="@+id/multiplicacion"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/Resta"
        android:layout_alignBottom="@+id/Resta"
        android:layout_toRightOf="@+id/Resta"
        android:text="*"
        android:textSize="30dp" />

    <Button
        android:id="@+id/division"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/multiplicacion"
        android:layout_alignBottom="@+id/multiplicacion"
        android:layout_toRightOf="@+id/multiplicacion"
        android:text="/"
        android:textSize="30dp" />

</RelativeLayout>
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:*

Paso 2: *Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.



Paso 3: *Eventos del botón.*

```
package com.nolascov.ejemplo_operaciones;
import android.os.Bundle;
import android.widget.*;
import android.app.Activity;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.view.View.OnClickListener;

public class MainActivity extends Activity implements OnClickListener {

    int resultado;
    int n1;
    int n2;
    EditText txtNumero1;
    EditText txtNumero2;
    Button Suma;
    Button Resta;
    Button Multiplicacion;
    Button Division;
    TextView txtResultado;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        txtNumero1 =(EditText) findViewById(R.id.txtNumero1);
        txtNumero2 =(EditText) findViewById(R.id.txtNumero2);

        Suma =(Button) findViewById(R.id.Suma);
        Resta =(Button) findViewById(R.id.Resta);
        Multiplicacion =(Button) findViewById(R.id.multiplicacion);
        Division =(Button) findViewById(R.id.division);

        txtResultado =(TextView) findViewById(R.id.txtResultado);

        Suma.setOnClickListener(this);
        Resta.setOnClickListener(this);
        Multiplicacion.setOnClickListener(this);
        Division.setOnClickListener(this);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.activity_main, menu);
        return true;
    }
    public void onClick(final View clicked) {
            switch(clicked.getId()) {

            case R.id.Suma:
              resultado=  Integer.parseInt(txtNumero1.getText().toString())+Integer.
parseInt(txtNumero2.getText().toString());
            txtResultado.setText(String.valueOf(resultado));
          break;
```

```
                case R.id.Resta:
                  resultado=  Integer.parseInt(txtNumero1.getText().toString())-Integer.
parseInt(txtNumero2.getText().toString());
                txtResultado.setText(String.valueOf(resultado));
            break;
                case R.id.multiplicacion:
                  resultado=  Integer.parseInt(txtNumero1.getText().toString())*Integer.
parseInt(txtNumero2.getText().toString());
                txtResultado.setText(String.valueOf(resultado));
            break;
                case R.id.division:
                  resultado=  Integer.parseInt(txtNumero1.getText().toString())/Integer.
parseInt(txtNumero2.getText().toString());
                txtResultado.setText(String.valueOf(resultado));
            break;

                                }
                        }

}}
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.*

Paso 2: *Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android



6.9. EJEMPLOCONTROLES2



Ahora diseñe una aplicación que permita el ingreso de las credenciales de un usuario (usuario y contraseña).

Paso 1: Una vez creada la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto* con el nombre de proyecto: EjemploControles2 (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

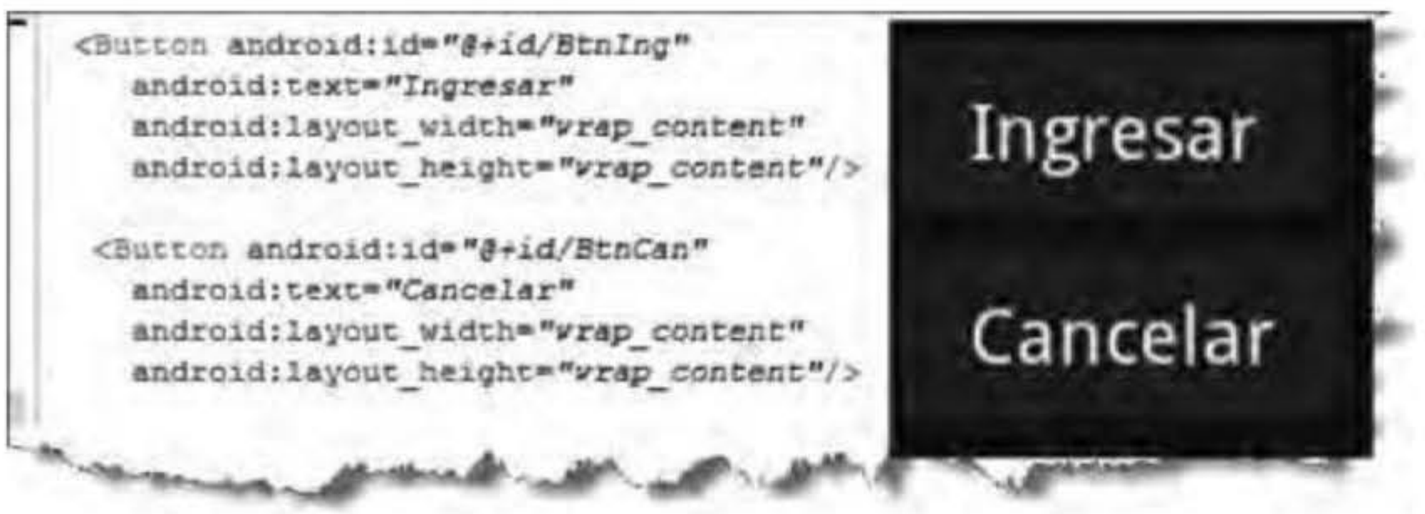
Paso 2: Edite el archivo main.xml y añada el botón.

Paso 3: Añada etiquetas (TextView) y textos editables (EditText):





Paso 4: Modifique los botones.



EJECUTANDO LA APLICACIÓN

Paso 1: Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.

Paso 2: Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.



La figura muestra la aplicación que se ejecuta en el emulador de Android.



6.9.1. AbsoluteLayout

Algunas veces para desarrollar una aplicación es más fácil trabajar con AbsoluteLayouts, donde los elementos que van en su interior ocupan posiciones absolutas en la pantalla, especificando sus posiciones mediante algunos atributos:

```
android:layout_x="Xpx"
android:layout_y="Ypx"
```

Se puede colocar una imagen o un botón en unas coordenadas en concreto, teniendo en cuenta que el punto de referencia, el (0,0), es la esquina superior izquierda. Probablemente, esta funcionalidad desaparecerá para las últimas versiones de Android porque se está poblando el mercado de la telefonía móvil con Android y cada dispositivo cuenta con su tamaño de pantalla.

No puede trabajar con posiciones absolutas de pantalla cuando no conoce qué teléfono va a usar su aplicación.

Nota:

Se recomienda no usar AbsoluteLayouts. Es mejor usar, en su lugar, RelativeLayouts.

Seguidamente, se expondrá el uso de AbsoluteLayout en la aplicación EjemploControles2:

```
<?xml version=”1.0” encoding=”utf-8”?>
<AbsoluteLayout xmlns:android=”http://schemas.android.com/apk/res/android”
android:orientation=”vertical”
android:layout_width=”fill_parent”
android:layout_height=”fill_parent”>
<TextView
    android:id=”@+id/lbl1”
    android:layout_width=”fill_parent”
android:layout_height=”wrap_content”
android:text=”User Name:”
android:layout_x=”10px”
android:layout_y=”10px”/>
<EditText
    android:id=”@+id/txtUser”
android:layout_width=”250dp”
android:layout_height=”wrap_content”
android:textSize=”18dp”
    android:layout_below=”@+id/lbl1”
android:layout_centerHorizontal=”true”
    android:layout_x=”110px”
android:layout_y=”10px”/>
<TextView
    android:id=”@+id/lbl2”
    android:layout_width=”fill_parent”
android:layout_height=”wrap_content”
android:text=”Contraseña:”
android:layout_x=”10px”
android:layout_y=”80px”/>
<EditText
    android:id=”@+id/txtClave”
android:layout_width=”250dp”
android:layout_height=”wrap_content”
android:textSize=”18dp”
    android:layout_below=”@+id/lbl2”
android:layout_centerHorizontal=”true”
android:layout_x=”110px”
android:layout_y=”80px”/>
<Button android:id=”@+id/BtnIng”
android:text=”Ingresar”
android:layout_width=”wrap_content”
android:layout_height=”wrap_content”
android:layout_x=”10px”
android:layout_y=”130px”/>
<Button android:id=”@+id/BtnCan”
android:text=”Cancelar”
android:layout_width=”wrap_content”
android:layout_height=”wrap_content”
android:layout_x=”140px”
android:layout_y=”130px”/>
</AbsoluteLayout>
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.*

Paso 2: *Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.





Nota:

Las posiciones indican dónde se motrarán los controles, según su posicionamiento absoluto.

Paso 3: *Eventos de un botón.*

*EjemploControles2Activity.java

```
package como.nolascov.EjemploControles2;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

public class EjemploControles2Activity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        //---Boton Ingresar---
        Button BtnIng = (Button) findViewById(R.id.BtnIng);
        BtnIng.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                Toast.makeText (getBaseContext(),"Comprobando Credenciales",Toast.LENGTH_SHORT).show();
            }
        });

        //---Boton Cancelar---
        Button btnCan = (Button) findViewById(R.id.BtnCan);
        btnCan.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                Toast.makeText (getBaseContext(),"Cancelando",Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.*

Paso 2: *Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.



6.10. ListView



Muestra una lista de ítems verticalmente. Se puede diseñar la apariencia de los elementos de la lista. Normalmente, se usa con una ListActivity y un ListAdapter.

6.10.1. Crear un archivo XML

Se va a crear una aplicación que muestre una lista de los incas y que al momento de seleccionar un mensaje de notificación aparezca dicho mensaje.

Paso 1: Una vez creada la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto* con el nombre de proyecto: EjemploListas (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

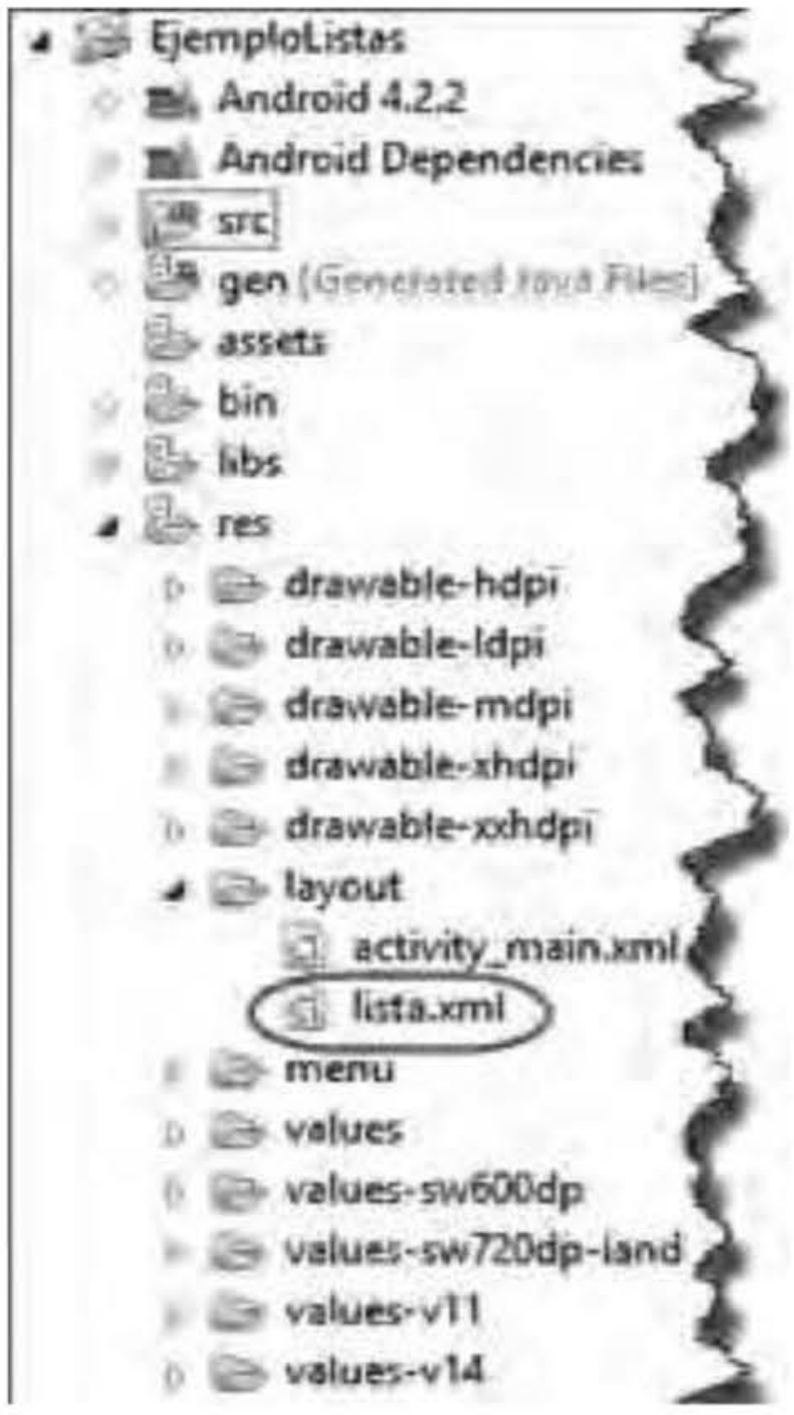
Paso 2: Añada un archivo XML para crear la lista.

Paso 3: Seleccione la carpeta res/layout y presione clic derecho:



Paso 4: Ingrese el nombre del archivo:lista y presione *Finalizar.*





6.10.2. Diseñar una ListView simple

Para diseñar la vista de los elementos de la lista en res/layout/ lista.xml, realice lo siguiente:

Paso 1: Edite el archivo lista.xml y añada lo siguiente:

lista.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/androi
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

        <TextView
            android:id="@+id/seleccion"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:textColor="#A4C639"
            android:textSize="20dip" />

     <ListView
         android:id="@id/android:list"
         android:layout_width="match_parent"
         android:layout_height="0dp"
         android:layout_weight="1"
         android:drawSelectorOnTop="false" />

</LinearLayout>
```

Paso 2: Modifique el archivo MainActivity.java.

```
package com.inkadroid.ejemplolistas;

import android.app.Activity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.view.View;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.AdapterView;

public class MainActivity extends Activity {

    /*
     Definimos un objeto de tipo TextView y otro de tipo ListView
    donde almacenaremos las referencias a los objetos que definimos
    en el archivo XML:lista
    */
    ListView lv;
    TextView seleccionado;
    //definimos un arreglo donde almacenamos los valores a mostrar
```

```
    String[] incas = {
                    "Manco capac",
                    "Sinchi Roca",
                    "Lloque Yupanqui",
                    "Mayta Cápac",
                    "Cápac Yupanqui",
                    "Inca Roca",
                    "Yáhuar Huaca",
                    "Wiracocha",
                    "Pachacútec",
                    "Amaru Inca Yupanqui",
                    "Túpac Inca Yupanqui",
                    "Huayna Cápac",
                    "Huáscar",
                    "Atahualpa"};

    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.lista);
        //hacemos referencia a los dos objetos:
        lv = (ListView)findViewById(android.R.id.list);
        seleccionado = (TextView)findViewById(R.id.seleccion);

        /*
        Paso siguiente es llenar la lista con las opciones,
        para lo cuál llamamos al método setAdapter()
        que recibe como parámetros el contexto en el cuál
        se está utilizando la lista (this), el nombre del recurso
         que define el aspecto visual de la lista
       (en este caso hacemos uso de un layout predefinido por Android: android.R.layout.
simple_list_item_1)
         y por último, el arreglo de datos que en
         este caso definimos en la variable datos.
         */

          lv.setAdapter(new ArrayAdapter<String>(this,android.R.layout.simple_list_
item_1,incas));

        /*
        ahora asignamos variable lv un listener que nos permitirá
        definir una acción que responda al evento de seleccionar
        un ítem de la lista. En este caso, lo que hacemos es indicar
        que se nos muestre el texto de la opción en el TextView.
         */
        lv.setOnItemClickListener(new OnItemClickListener() {
            public void onItemClick(AdapterView<?> parent, View view,
                int posicion, long id) {
              seleccionado.setText("Elejistes: " + incas[posicion]);
            }
          });
    }
}
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.*

Paso 2: *Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.



6.10.3. Diseñar una ListView múltiple



Paso 1: Se presenta la ListView.

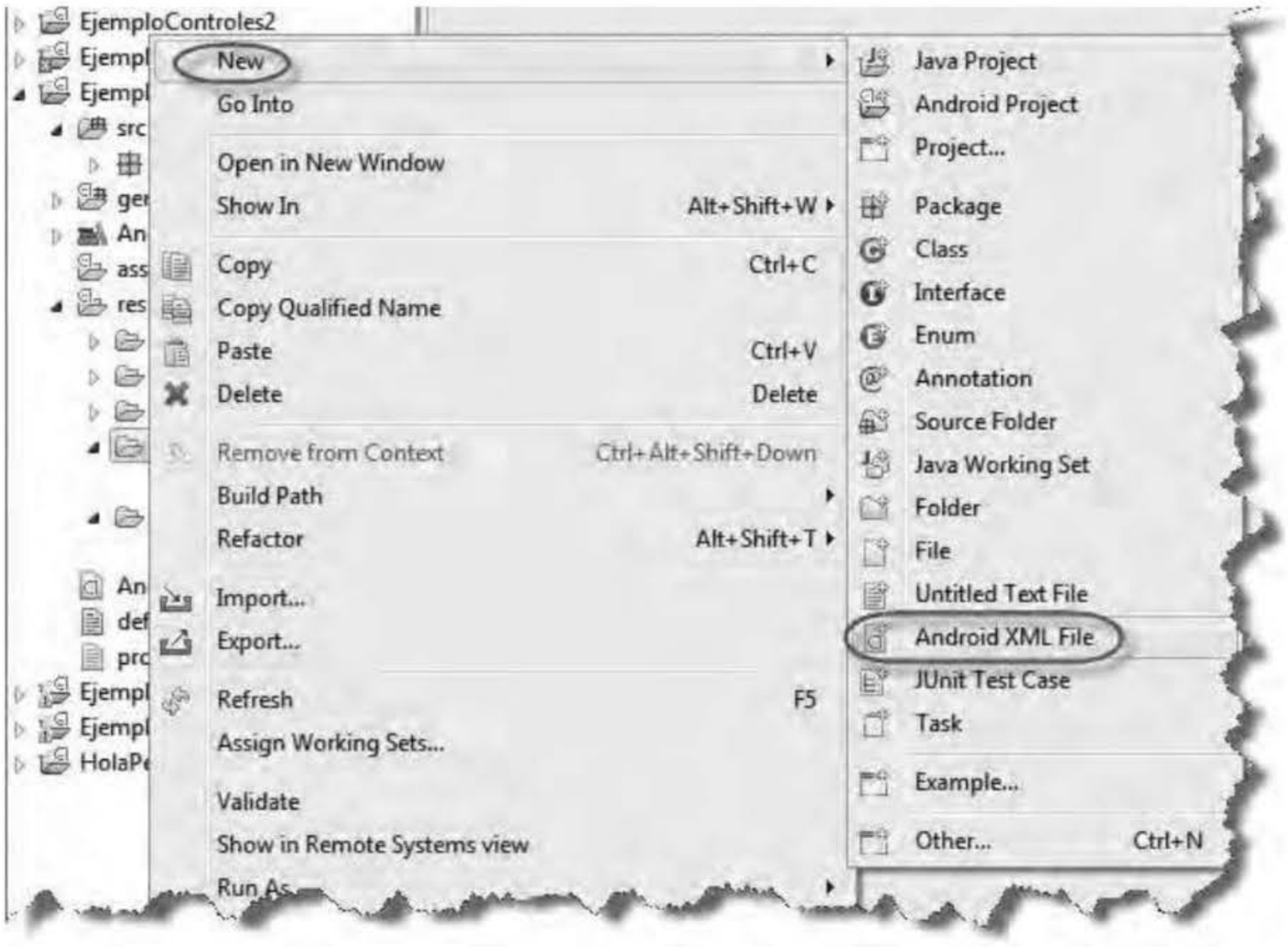
Paso 2: Cree una aplicación que muestre una lista con dos elementos:

- Título del libro
- ISBN
- Autor
- Páginas

Paso 3: Una vez creada la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto* con el nombre de proyecto: EjemploListas2 (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

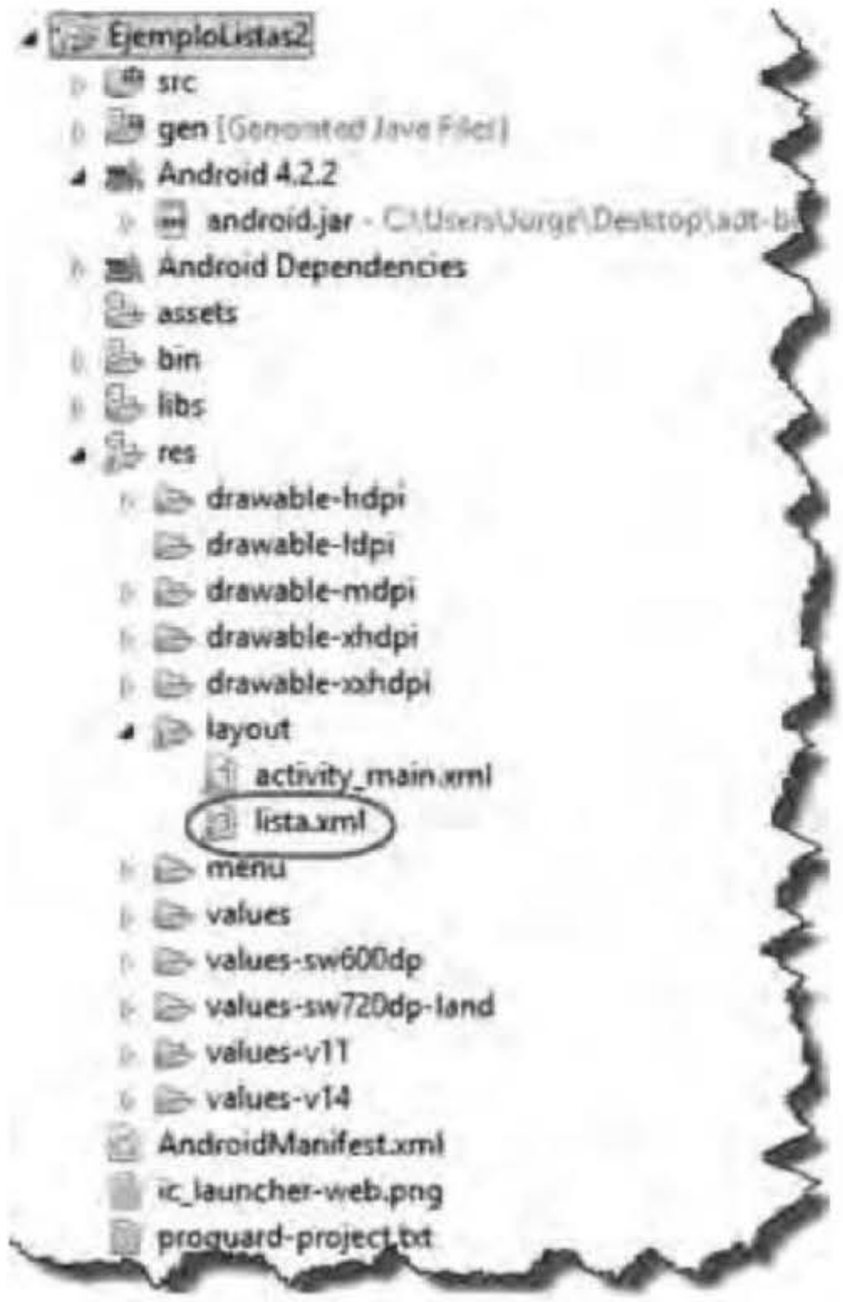
Paso 4: Añada un archivo XML para crear la lista.

Paso 5: Seleccione la carpeta res/layout y presione clic derecho:



Paso 6: Ingrese el nombre del archivo:lista y presione Finalizar.





6.10.4. Diseñar la vista de los elementos de la lista en res/layout/ lista.xml

Paso 1: Edite el archivo lista. xml y añada lo siguiente:



activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">

    <TextView
        android:id="@+id/seleccion"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textColor="#A4C639"
        android:textSize="20dip" />

    <ListView
        android:id="@id/android:list"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:background="#ffeee8aa"
        android:drawSelectorOnTop="false" >
    </ListView>

</LinearLayout>
```

Paso 2: Modifique el archivo activity_main.xml.

Paso 3: Modifique el archivo MainActivity.java.

```
package com.inkadroid.ejemplolistas2;
import java.util.ArrayList;
import java.util.HashMap;
import android.app.ListActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ListView;
import android.widget.SimpleAdapter;
import android.widget.TextView;
import android.widget.AdapterView.OnItemClickListener;

public class MainActivity extends ListActivity {
    ListView lv;
    TextView seleccionado;
    static final ArrayList<HashMap<String,String>> list =
            new ArrayList<HashMap<String,String>>();

    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        SimpleAdapter adapter = new SimpleAdapter(
            this,
            list,
            R.layout.lista,
            new String[] {“titulo”,”isbn”,”autor”,”paginas”},
            new int[] {R.id.titulo,R.id.isbn, R.id.autor, R.id.paginas}
        );
        populateList();
        setListAdapter(adapter);
        seleccionado = (TextView)findViewById(R.id.seleccion);
        lv = (ListView)findViewById(android.R.id.list);
```

```
        lv.setOnItemClickListener(new OnItemClickListener() {
            public void onItemClick(AdapterView<?> parent, View view,
                int posicion, long id) {
             seleccionado.setText("Elemento: " +posicion);
            }
          });
    }

    private void populateList() {
    HashMap<String,String> map = new HashMap<String,String>();

    map.put("titulo",
    "Programación Transact con SQL SERVER 2012 ");
    map.put("isbn", " 978-612-304-084-0");
    map.put("autor", "Manuel Torres Remon");
    map.put("paginas", "328 Paginas");
    list.add(map);

    map = new HashMap<String,String>();

    map.put("titulo",
    "Program. orientada a objetos con Visual Basic 2012 ");
    map.put("isbn", "978-612-304-085-7");
    map.put("autor", "Manuel Torres Remon");
    map.put("paginas", "416 Paginas");
    list.add(map);

    map = new HashMap<String,String>();

    map.put("titulo",
    "Program. orientada a objetos con Visual Basic 2012 ");
    map.put("isbn", "978-612-304-085-7");
    map.put("autor", "Manuel Torres Remon");
    map.put("paginas", "416 Paginas");
    list.add(map);

    map = new HashMap<String,String>();

    map.put("titulo",
    "Desarrollo de aplicaciones con Android");
    map.put("isbn", "978-612-304-047-5");
    map.put("autor", "Jorge Nolasco Valenzuela");
    map.put("paginas", "264 Paginas");
    list.add(map);

    map = new HashMap<String,String>();

    map.put("titulo",
    "Una nueva forma de desarrollo Visual Studio 2010");
    map.put("isbn", "978-612-304-004-8");
    map.put("autor", "Cristian Sánchez F.");
    map.put("paginas", "384 Paginas");
    list.add(map);

            }
}
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.*

Paso 2: *Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.



6.10.5. Aplicacion Pisco

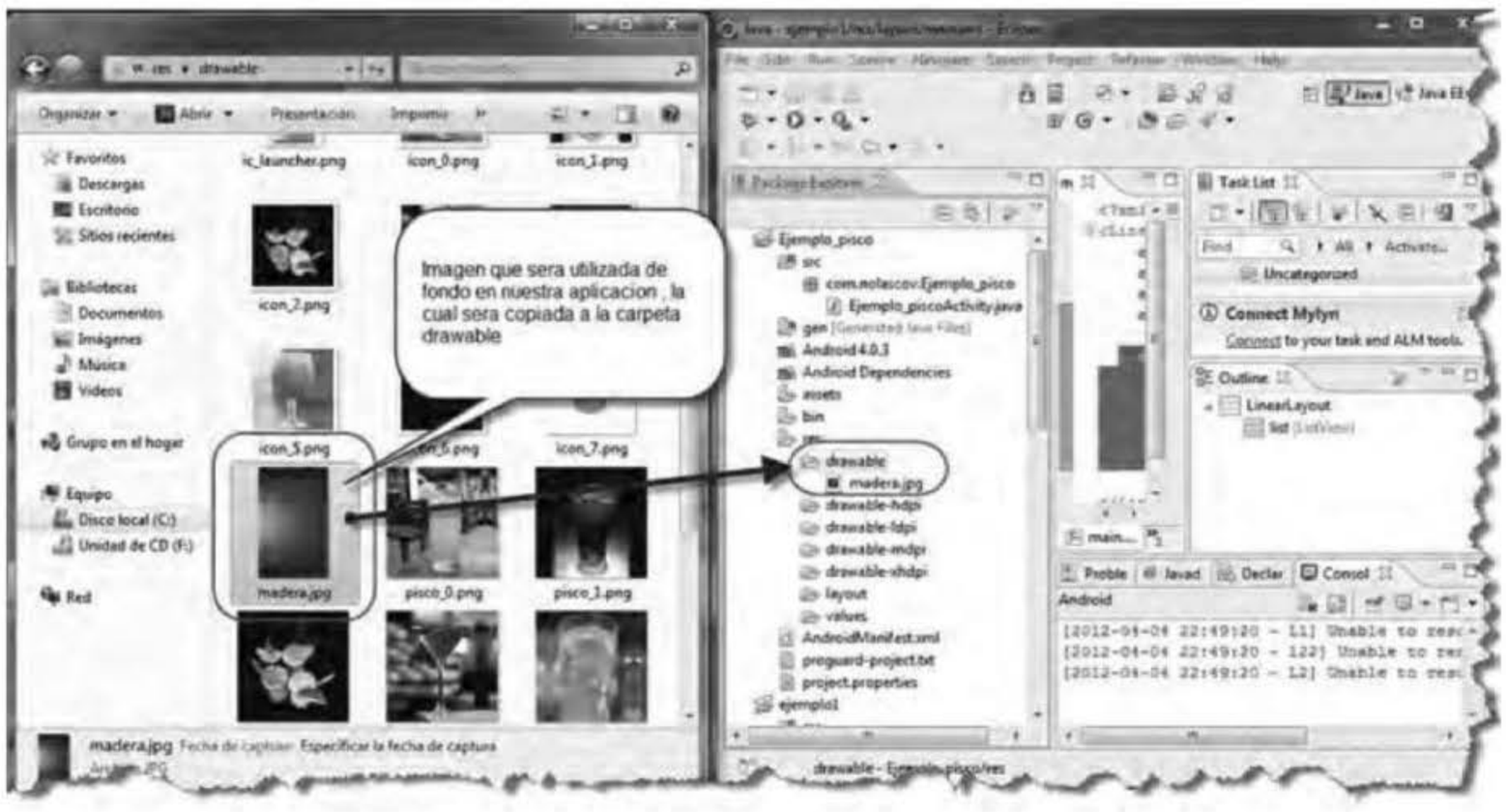


Crear una aplicación que muestre una lista de los principales tragos preparados con el pisco peruano.

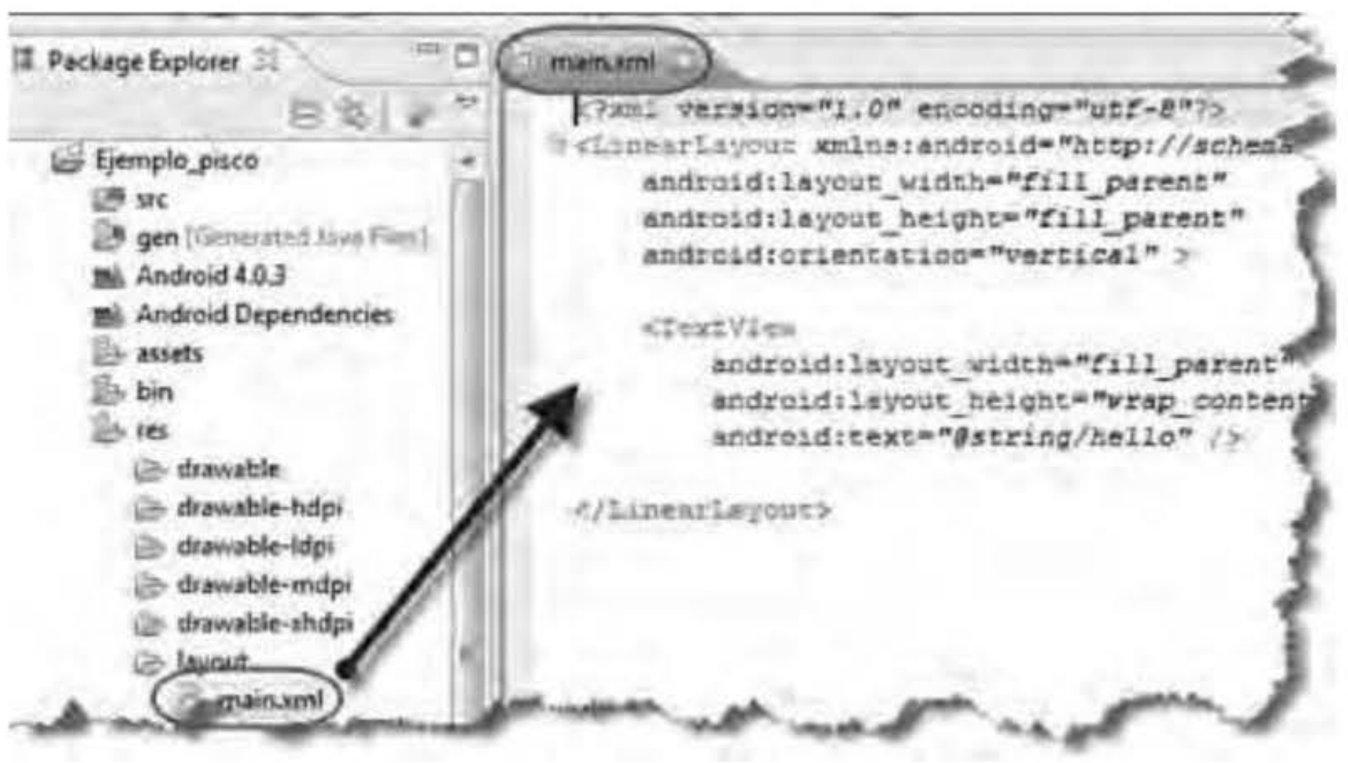
A. CREAR EL ARCHIVO XML

Paso 1: Una vez creada la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto* con el nombre de proyecto: Ejemplo_Pisco_ (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

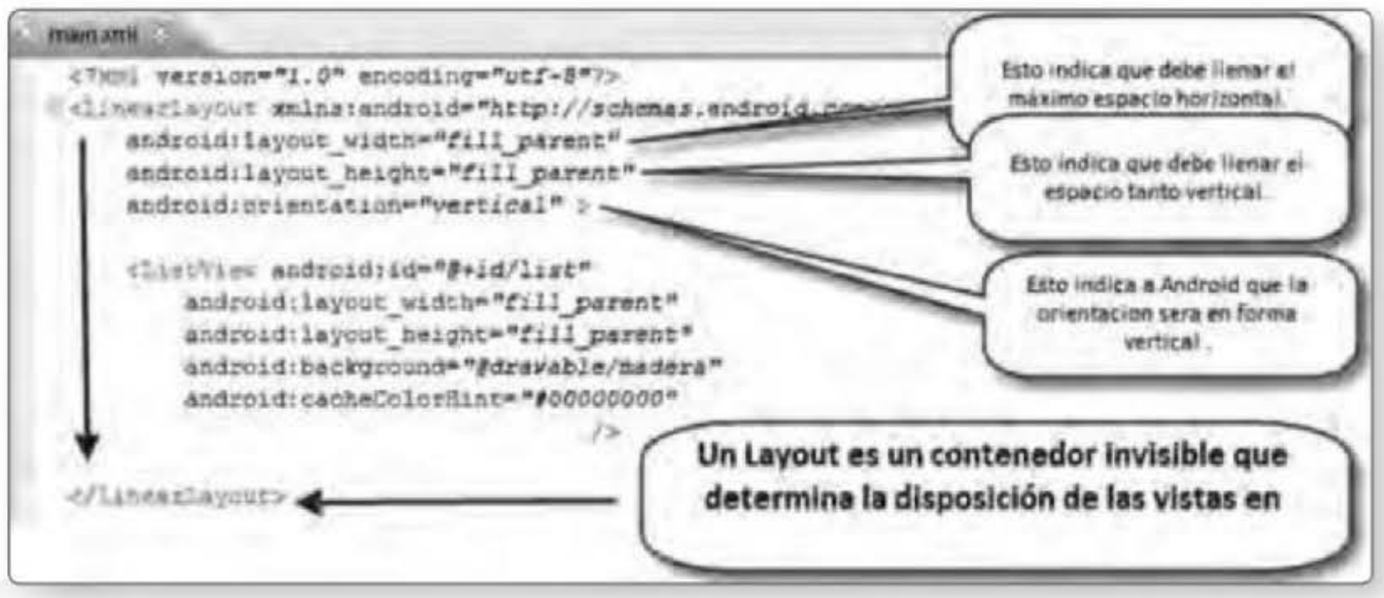
Paso 2: Copie la imagen que será utilizada como imagen de fondo.



Paso 3: El archivo main.xml define la interfaz de usuario (UI) de su aplicación. La vista predeterminada es la vista *Diseño*, que establece la actividad gráfica. Para modificar la interfaz de usuario, haga clic en la pestaña main.xml, situado en la parte inferior. Se puede ubicar este archivo y expanda el proyecto Ejemplo_pisco. Haga clic en las flechas de diversos que aparece a la izquierda de cada elemento en el proyecto, en la carpeta res / layout.



Paso 4: Para añadir un control ListView a su interfaz de usuario, es necesario que conozca primero los siguientes puntos:



Paso 5: Añada un archivo XML para crear la lista. Seleccione la carpeta res/layout y presione clic derecho:



Paso 6: Ingrese el nombre del archivo:pisco_entry y presione Finalizar.





B. DISEÑAR LA VISTA DE LOS ELEMENTOS DE LA LISTA EN RES/LAYOUT/ PISCO_ENTRY.XML

Paso 1: Edite el archivo pisco_entry.xml y añada lo siguiente:

pisco_entry.xml

```
<?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="fill_parent">
  <!-- icono -->
    <ImageView
        android:id="@+id/icono"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:padding="8dp" />
  <!-- titulo -->
  <TextView
      android:id="@+id/titulo"
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:layout_toRightOf="@id/icono"
      android:layout_alignTop="@id/icono"
  android:layout_margin="1dp"
      android:textColor="#FFFFFF"
      android:textSize="14sp"
      android:textStyle="bold" />
  <!-- subtitulo -->
  <TextView
      android:id="@+id/subtitulo"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_alignLeft="@id/titulo"
      android:layout_below="@id/titulo"
  android:layout_margin="1dp"
```

Android soporta 3 tipos de imagenes PNG, JPG, GIF , El componente de diseño ImageView es el elemento base que se utiliza para la visualización de imágenes en Android.

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.*



Paso 2: *Copie las siguientes imágenes:*



Paso 3: *Añada la clase PiscoHolder.java; para ello, seleccione la carpeta /src y presione clic derecho:*

- **Clase PiscoHolder:** Es un encapsulamiento del ítem que, a su vez, está formado por más de un view.

PiscoHolder.java

```
package com.nolascov.Ejemplo_pisco;
// esta clase contiene los widgets de cada item en la lista
import android.widget.ImageView;

public class PiscoHolder {

    public ImageView imageView;
    public TextView titleView;
    public TextView subTitleView;

}
```

Creamos una clase donde indicamos los widgets que necesitamos : un ImageView y dos TextView

- **Clase PiscoEntry:** Añada la clase PiscoEntry.java donde se registran los campos que poseerán los datos a mostrar:

PiscoEntry.java

```
package com.nolascov.Ejemplo_pisco_;
// esta clase es un registro que contiene los campos
public class PiscoEntry {

    private final String nombre;
    private final String descripcion;
    private final String ingredientes;
    private final String preparacion;
    private final int    picture;
    private final int    icon;

    public PiscoEntry( final String nom, final String desc,
                       final String ing, final String prep,
                       final int pict, final int ico ) {
        nombre = nom;
        descripcion = desc;
        ingredientes = ing;
        preparacion = prep;
        picture = pict;
        icon = ico;
    }
    public String getTitle() {
        return nombre;
    }
    public String getAuthor() {
        return descripcion;
    }
    public int getIcon() {
        return icon;
    }
}
```

- **Clase PiscoAdapter:** Sirve para actualizar el contenido de cada uno de los ítems mostrados en la lista.

Paso 4: *Añada la clase PiscoAdapter.java:*

```
package com.nolascov.Ejemplo_pisco_;
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;

public final class PiscoAdapter extends ArrayAdapter<PiscoEntry> {
    private final int piscoEntryLayout;
    private static PiscoHolder viewHolder;

    public PiscoAdapter( final Context context, final int idLayout ) {
        super(context, 0);
        piscoEntryLayout = idLayout;
    }
    // devolver la instancia View con datos, para ser mostrada en la lista
    @Override
     public View getView(final int position, final View convertView, final ViewGroup
parent) {
    //recuperar su correspondiente ViewHolder , lo que optimiza la eficiencia de
búsqueda
        final View view = getWorkingView(convertView);
        viewHolder = getViewHolder(view);
        final PiscoEntry entry = getItem(position);
       //Configuración de la vista del título es sencillo
        viewHolder.titleView.setText(entry.getTitle());
        //Configuración de la vista de subtítulos requiere un poco de formato
        viewHolder.subTitleView.setText( entry.getAuthor() );
        //Configuración de vista de la imagen también es sencillo
        viewHolder.imageView.setImageResource(entry.getIcon());
        return view;
    }
    // obtener la vista de trabajo
    public final View getWorkingView(final View convertView) {
    //El workingView es básicamente la convertView volver a utilizar si es posible
    //o inflado nueva si no es posible
        View workingView = null;

        if( convertView instanceof View ) {
            workingView = convertView;
        } else {
                        final LayoutInflater inflater = (LayoutInflater)getContext().
getSystemService(Context.LAYOUT_INFLATER_SERVICE);
                workingView = inflater.inflate(piscoEntryLayout, null);
            }
            return workingView;
        }
        // obtener el objeto guardado en este View
        public final PiscoHolder getViewHolder(final View workingView) {
        // El viewHolder nos permite evitar el repitando hace referencia a que miran para
arriba
           // A partir de puntos de vista son reciclados, estas referencias no va a cambiar
            final Object tag = workingView.getTag();
            viewHolder = null;
```

```
        if( !(tag instanceof PiscoHolder) ) {
            viewHolder = new PiscoHolder();
            viewHolder.titleView = (TextView) workingView.findViewById(R.id.titulo);
                    viewHolder.subTitleView = (TextView) workingView.findViewById(R.
id.subtitulo);
            viewHolder.imageView = (ImageView) workingView.findViewById(R.id.icono);
            workingView.setTag(viewHolder);
        } else {
            viewHolder = (PiscoHolder)tag;
        }
            return viewHolder;
    }
}
```

Paso 5: *Edite la actividad Ejemplo_piscoActivity.java.*



Paso 6: *Modifique Ejemplo_piscoActivity.java.*

```
//crear una clase publica Ejemplo_piscoActivity heredando caracteristicas de la
clase base Activity
package com.nolascov.Ejemplo_pisco_;
import com.nolascov.Ejemplo_pisco_.PiscoAdapter;
import com.nolascov.Ejemplo_pisco_.PiscoHolder;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ListView;
import android.widget.Toast;
import android.widget.AdapterView.OnItemClickListener;
public class Ejemplo_pisco_Activity extends Activity implements OnItemClickListener{
//implementar el escucha de evento OnItemClickListener
        private static PiscoHolder viewHolder;
         private PiscoAdapter newsEntryAdapter;
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        // configuramos la lista
        final ListView listView = (ListView) findViewById(R.id.list);
```

```
        //evento de escucha para el listView
        listView.setOnItemClickListener(this);
        // instanciamos la clase PiscoAdapter
        newsEntryAdapter = new PiscoAdapter(this, R.layout.pisco_entry);
        // indicamos el adapater que utilizara el listView
        listView.setAdapter(newsEntryAdapter);
        // llenamos la lista a travez del adapter
        addNewsEntries( newsEntryAdapter );
    }
    private void addNewsEntries( PiscoAdapter adapter ) {
       adapter.add( new PiscoEntry( “Chilcano”, “Pisco, jugo de limÃ³n, ginger ale,
amargo de angostura”, “”, “”, R.drawable.pisco_0, R.drawable.icon_0 ) );
      adapter.add( new PiscoEntry( “PerÃº Libre”, “Pisco, rodaja de limÃ³n, gaseosa
oscura, hielo”, “”, “”, R.drawable.pisco_1, R.drawable.icon_1 ) );
        adapter.add( new PiscoEntry( “Cocktel De Fruta”, “Pisco, pulpa de fresa,
jarabe de goma, leche, hielo”, “”, “”, R.drawable.pisco_2, R.drawable.icon_2 ) );
        adapter.add( new PiscoEntry( “CapitÃ¡n”, “Pisco, Vermut rojo, hielo”, “”,
“”, R.drawable.pisco_3, R.drawable.icon_3 ) );
        adapter.add( new PiscoEntry( “Pisco Tonic”, “Pisco, agua tÃ³nica, hielo”,
“”, “”, R.drawable.pisco_4,R.drawable.icon_4 ) );
      adapter.add( new PiscoEntry( “Pisco Sunrise”, “Pisco, jugo de naranja, jarabe
de granadina, hielo”, “”, “”, R.drawable.pisco_5,R.drawable.icon_5 ) );
       adapter.add( new PiscoEntry( “Biblia”, “Pisco, vino Oporto, jarabe de goma,
leche, huevo entero, hielo”, “”, “”, R.drawable.pisco_6,R.drawable.icon_6 ) );
        adapter.add( new PiscoEntry( “Pisco Sour”, “Pisco, clara de huevo, jugo de
limÃ³n, jarabe de goma, hielo”, “”, “”, R.drawable.pisco_7,R.drawable.icon_7 ) );
    }
    public void onItemClick( final AdapterView<?> av, final View view, final int pos,
final long id ) {
                viewHolder  =  newsEntryAdapter.getViewHolder(  newsEntryAdapter.
getWorkingView(view) );
          Toast.makeText(getApplicationContext(), viewHolder.titleView.getText(),
Toast.LENGTH_SHORT).show();
    }
}
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.*

Paso 2: *Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.





Paso 3: *Seleccione algún ítem.*

Puede descargar la aplicación desde el siguiente link:

https://play.google.com/store/search?q=Inkadroid&c=apps

6.11. SlidingDrawer (cajón deslizante)

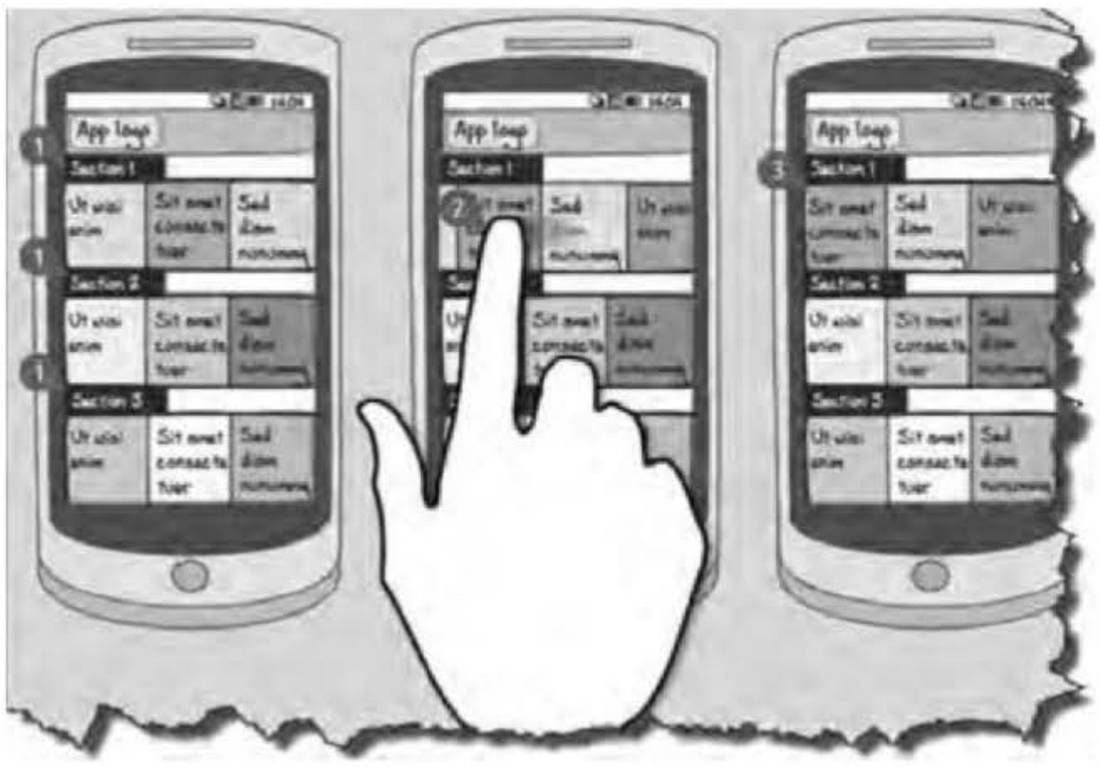


SlidingDrawer es un widgets que se utiliza para mostrar y ocultar un panel con diversos elementos en su interior.

La idea es usarlo para gestionar su aparición en pantalla desde alguna parte de la vista (izquierda-derecha, derecha-izquierda, arriba-abajo, abajo-arriba). Dentro de la vista del SlidingDrawable, se puede añadir un layout personalizado. A continuación, se desarrollará un ejemplo.

6.11.1. ScrollView y HorizontalScrollView

ScrollView y HorizontalScrollView son contenedores de diseño para una jerarquía de vista que se puede desplazar vertical u horizontalmente por el usuario, permitiendo que sea más grande que la pantalla física. Un ScrollView/HorizontalScrollView es un FrameLayout.



6.11.2. Aplicacion Wallpaper



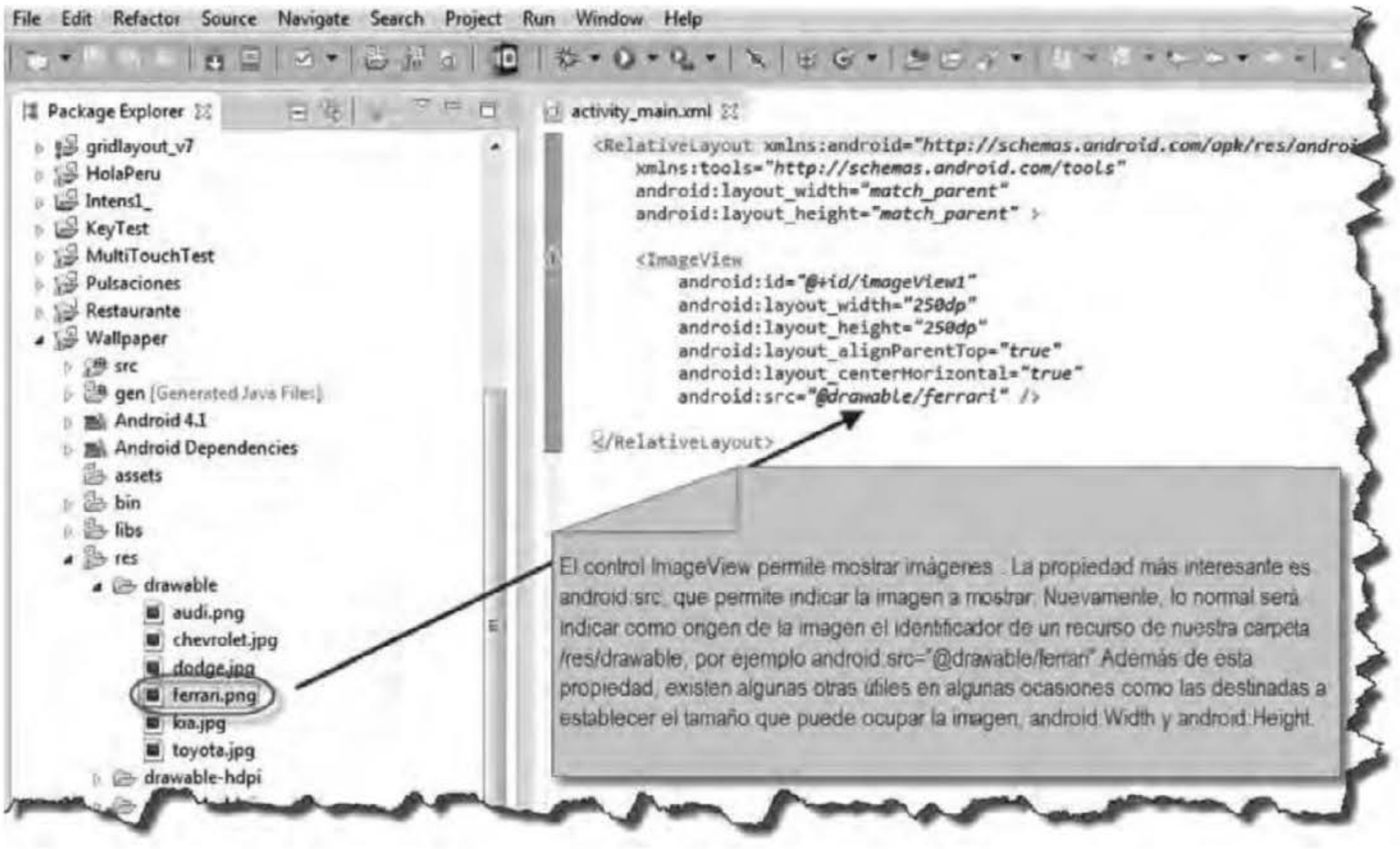
Crear una aplicación que muestre una lista de los principales tragos preparados con el pisco peruano.

Paso 1: Una vez creada la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto* con el nombre de proyecto: Wallpaper (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 2: Copie las siguientes imágenes:



Paso 3: Modifique el archivo activity_main.xml.



Paso 4: Añada un HorizontalScrollView para mostrar las imágenes observadas.

```
<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” >

    <ImageView
        android:id=”@+id/imageView0”
        android:layout_width=”250dp”
        android:layout_height=”250dp”
        android:layout_alignParentTop=”true”
        android:layout_centerHorizontal=”true”
        android:src=”@drawable/ferrari” />

    <HorizontalScrollView
        android:id=”@+id/horizontalScrollView1”
        android:layout_width=”250dp”
        android:layout_height=”wrap_content”
        android:layout_centerHorizontal=”true”
        android:layout_below=”@+id/imageView0”
        android:scrollbars=”horizontal|none” >

        <LinearLayout
            android:layout_width=”fill_parent”
            android:layout_height=”wrap_content”
            android:orientation=”horizontal” >
```

```
        <ImageView
        android:id="@+id/imageView1"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:src="@drawable/ferrari" />

        <ImageView
        android:id="@+id/imageView2"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:src="@drawable/audi" />

        <ImageView
        android:id="@+id/imageView3"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:src="@drawable/chevrolet" />

        <ImageView
        android:id="@+id/imageView4"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:src="@drawable/dodge" />

        <ImageView
        android:id="@+id/imageView5"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:src="@drawable/kia" />

        <ImageView
        android:id="@+id/imageView6"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:src="@drawable/toyota" />
            </LinearLayout>
        </HorizontalScrollView>


    </RelativeLayout>
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.*



Paso 2: *Separe las imágenes que se visualizan en el HorizontalScrollView.*

```
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal" >

  <ImageView
  android:id="@+id/imageView1"
  android:layout_width="150dp"
  android:layout_height="150dp"
  android:layout_alignParentTop="true"
  android:layout_centerHorizontal="true"
  android:paddingLeft="20dp"
  android:src="@drawable/ferrari" />

  <ImageView
  android:id="@+id/imageView2"
  android:layout_width="150dp"
  android:layout_height="150dp"
  android:layout_alignParentTop="true"
  android:layout_centerHorizontal="true"
  android:paddingLeft="20dp"
  android:src="@drawable/ferrari" />
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.*



Paso 2: *Separe las imágenes que se visualizan en el HorizontalScrollView.*

```
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal" >

  <ImageView
  android:id="@+id/imageView1"
  android:layout_width="150dp"
  android:layout_height="150dp"
  android:layout_alignParentTop="true"
  android:layout_centerHorizontal="true"
  android:paddingLeft="20dp"
  android:src="@drawable/ferrari" />

  <ImageView
  android:id="@+id/imageView2"
  android:layout_width="150dp"
  android:layout_height="150dp"
  android:layout_alignParentTop="true"
  android:layout_centerHorizontal="true"
  android:paddingLeft="20dp"
  android:src="@drawable/ferrari" />
```

Paso 3: *Modifique el archivo 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" >

    <ImageView
        android:id="@+id/imageView0"
        android:layout_width="250dp"
        android:layout_height="250dp"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:src="@drawable/ferrari" />

    <HorizontalScrollView
        android:id="@+id/horizontalScrollView1"
        android:layout_width="250dp"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/imageView0"
        android:scrollbars="horizontal|none" >

        <LinearLayout
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal" >

            <ImageView
            android:id="@+id/imageView1"
            android:layout_width="150dp"
            android:layout_height="150dp"
            android:layout_alignParentTop="true"
            android:layout_centerHorizontal="true"
            android:paddingLeft="20dp"
            android:src="@drawable/ferrari" />

            <ImageView
            android:id="@+id/imageView2"
            android:layout_width="150dp"
            android:layout_height="150dp"
            android:layout_alignParentTop="true"
            android:layout_centerHorizontal="true"
            android:paddingLeft="20dp"
            android:src="@drawable/audi" />

            <ImageView
            android:id="@+id/imageView3"
            android:layout_width="150dp"
            android:layout_height="150dp"
            android:layout_alignParentTop="true"
            android:layout_centerHorizontal="true"
            android:paddingLeft="20dp"
            android:src="@drawable/chevrolet" />

            <ImageView
            android:id="@+id/imageView4"
            android:layout_width="150dp"
```

```
        android:layout_height="150dp"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:paddingLeft="20dp"
        android:src="@drawable/dodge" />

        <ImageView
        android:id="@+id/imageView5"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:paddingLeft="20dp"
        android:src="@drawable/kia" />

        <ImageView
        android:id="@+id/imageView6"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:paddingLeft="20dp"
        android:src="@drawable/toyota" />
        </LinearLayout>
    </HorizontalScrollView>
</RelativeLayout>
```

6.11.3. Añadir funcionalidad

Paso 1: Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.



Paso 2: Cada vez que se muestran las imágenes de las diferentes marcas de autos al darle clic, la imagen principal cambie por la elegida. Para ello, modifique la actividad MainActivity.java.

```
package com.nolasco.wallpaper;

import com.nolasco.wallpaper.R.id;

import android.os.Bundle;
import android.app.Activity;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.ImageView;

public class MainActivity extends Activity implements OnClickListener{

    ImageView mostrar,ferrari,audi,chevrolet,dodge,kia,toyota;
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        mostrar = (ImageView) findViewById(id.imageView0);
        ferrari = (ImageView) findViewById(id.imageView1);
        audi = (ImageView) findViewById(id.imageView2);
        chevrolet = (ImageView) findViewById(id.imageView3);
        dodge = (ImageView) findViewById(id.imageView4);
        kia = (ImageView) findViewById(id.imageView5);
        toyota = (ImageView) findViewById(id.imageView6);

        ferrari.setOnClickListener(this);
        audi.setOnClickListener(this);
        chevrolet.setOnClickListener(this);
        dodge.setOnClickListener(this);
        kia.setOnClickListener(this);
        toyota.setOnClickListener(this);

    }

    public void onClick(View arg0) {
            // TODO Auto-generated method stub
            switch (arg0.getId())
            {
            //ferrari
            case R.id.imageView1:
        mostrar.setImageResource(R.drawable.ferrari);
        break;
            //audi
            case R.id.imageView2:
            mostrar.setImageResource(R.drawable.audi);
        break;
            //chevrolet
            case R.id.imageView3:
            mostrar.setImageResource(R.drawable.chevrolet);
        break;
            //dodge
            case R.id.imageView4:
            mostrar.setImageResource(R.drawable.dodge);
```

```
            break;
                //kia
                case R.id.imageView5:
                mostrar.setImageResource(R.drawable.kia);
            break;
                //toyota
                case R.id.imageView6:
                mostrar.setImageResource(R.drawable.toyota);
            break;
                }
        }
    }
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.*



Paso 2: *Para colocar la imagen seleccionada como Wallpaper, añada un botón de comando y modifique el archivo activity_main.xml.*

```
    <Button
        android:id="@+id/btnColocar"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="37dp"
        android:text="Colocar como Wallpaper" />

</RelativeLayout>
```

```
package com.nolasco.wallpaper;
import java.io.IOException;
import com.nolasco.wallpaper.R.id;
import android.os.Bundle;
import android.app.Activity;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.ImageView;

public class MainActivity extends Activity implements OnClickListener{

    ImageView mostrar,ferrari,audi,chevrolet,dodge,kia,toyota;
    Button colocar;
    int celular;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        mostrar = (ImageView) findViewById(id.imageView0);
        ferrari = (ImageView) findViewById(id.imageView1);
        audi = (ImageView) findViewById(id.imageView2);
        chevrolet = (ImageView) findViewById(id.imageView3);
        dodge = (ImageView) findViewById(id.imageView4);
        kia = (ImageView) findViewById(id.imageView5);
        toyota = (ImageView) findViewById(id.imageView6);
        colocar = (Button) findViewById(id.btnColocar);

        ferrari.setOnClickListener(this);
        audi.setOnClickListener(this);
        chevrolet.setOnClickListener(this);
        dodge.setOnClickListener(this);
        kia.setOnClickListener(this);
        toyota.setOnClickListener(this);
        colocar.setOnClickListener(this);

    }

    public void onClick(View arg0) {
            // TODO Auto-generated method stub
            switch (arg0.getId())
            {
            //ferrari
            case R.id.imageView1:
        mostrar.setImageResource(R.drawable.ferrari);
        celular=R.drawable.ferrari;
        break;
            //audi
            case R.id.imageView2:
            mostrar.setImageResource(R.drawable.audi);
        celular=R.drawable.audi;
        break;
            //chevrolet
            case R.id.imageView3:
            mostrar.setImageResource(R.drawable.chevrolet);
        celular=R.drawable.chevrolet;
```

```
            break;
                //dodge
                case R.id.imageView4:
                mostrar.setImageResource(R.drawable.dodge);
            celular=R.drawable.dodge;
            break;
                //kia
                case R.id.imageView5:
                mostrar.setImageResource(R.drawable.kia);
            celular=R.drawable.kia;
            break;
                //toyota
                case R.id.imageView6:
                mostrar.setImageResource(R.drawable.toyota);
            celular=R.drawable.toyota;
            break;
                //boton colocar
                case R.id.btnColocar:
                Bitmap    imagen    =    BitmapFactory.decodeStream(getResources().
    openRawResource(celular));
                            try {
                                    getApplicationContext().setWallpaper(imagen);
                            } catch (IOException e) {
                                    e.printStackTrace();
                            }
            break;

                }

        }

    }
```

6.11.3. Modificar el manifiesto

*Wallpaper Manifest

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.nolasco.wallpaper"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="15" />
    <uses-permission android:name="android.permission.SET_WALLPAPER"/>

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="@string/title_activity_main" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

EJECUTANDO LA APLICACIÓN

Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.



6.12. WEBVIEW



Permite embeber un navegador en una actividad. El navegador incluye opciones para navegar hacia adelante y hacia atrás, zoom.
Seguidamente, se diseñará una aplicación que permita mostrar el navegador.

Paso 1: Una vez creada la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto* con el nombre de proyecto: EjemploNavegar (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 2: Edite el archivo main.xml y añada el botón.

Paso 3: Añada el WebView:

main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.and
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
    <WebView
        android:id="@+id/webView"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</LinearLayout>
```

Paso 4: Edite el archivo de manifiesto AndroidManifest.xml y añada lo siguiente:

- Defina permisos y pantalla completa

EjemploNavegar Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.an
    package="com.nolascov.EjemploNavegar"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk android:minSdkVersion="14" />
    <uses-permission android:name="android.permission.INTERNET" />
    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name" >
        <activity
            android:label="@string/app_name"
            android:name=".EjemploNavegarActivity"
            android:theme="@android:style/Theme.NoTitleBar"
            >
            <intent-filter >
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Permiso para acceder a Internet

Pantalla sin titulo

- Añada la vista a la actividad

EjemploNavegarActivity.java

```
package com.nolascov.EjemploNavegar;

import android.app.Activity;

public class EjemploNavegarActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        Uri uri = Uri.parse("http://www.google.es");
        Intent intent = new Intent(Intent.ACTION_VIEW, uri) ;
        startActivity(intent);
        WebView webview = new WebView(this);
        setContentView(R.layout.main);
    }
}
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.*



Paso 2: *Si el emulador se bloquea, es necesario deslizar el botón de desbloqueo para desbloquearlo.*



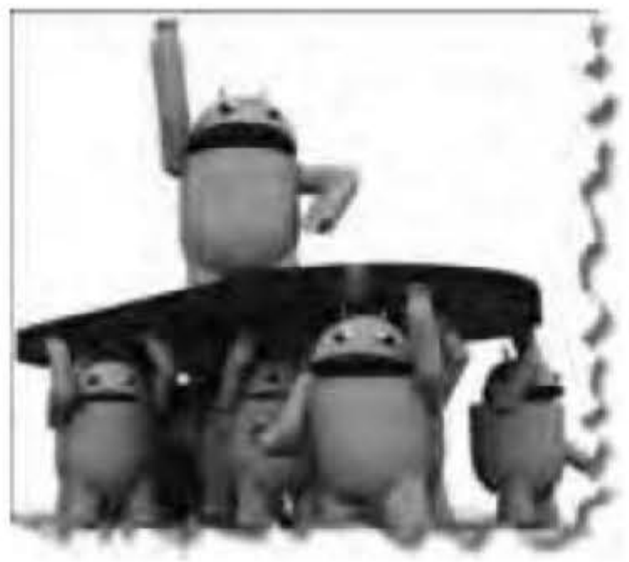
La figura muestra la aplicación que se ejecuta en el emulador de Android.





Uso de Intents and Services

7.1. INTENTS



El intent es un mecanismo de paso de mensajes entre una actividad a otra que puede trabajar entre las propias actividades o entre diferentes aplicaciones. Los intents pueden ser utilizados como:

- Declarar la intención de una actividad o servicio que va ser iniciada al realizar una acción.
- Difundir un evento ocurrido.
- Iniciar un servicio en particular o una actividad.

Uno de los usos más frecuentes de los intents es iniciar nuevas actividades. Existen dos formas de intents que se utilizan.

- ***Explícita:*** Cuando se especifica la clase a cargar)
- ***Implícitas:*** Por una solicitud sobre una acción que va ser realizada sobre unos datos específicos.

Los intents también son utilizados como emisores o comunicadores (*broadcast)* de eventos a través del sistema. Cualquier aplicación puede registrar este tipo de emisiones o Eventos y reaccionar por medio de los *intents*.

Así por ejemplo, los *intents* pueden anunciar eventos del sistema, como el cambio de estado en la conexión de Internet o el nivel de batería. Aplicaciones nativas de Android como el marcador del teléfono (*Phone Dialer*) o el SMS Messenger registran componentes para escuchar eventos específicos como una llamada entrante o un mensaje SMS recibido.

7.2. ACCIÓN

Una cadena con el nombre de la acción a realizar o en el caso de los intentos de difusión, la acción que llevó a cabo y se está informando. La clase de Intención define una serie de constantes acciones, incluyendo los siguientes:

CONSTANTE	COMPONENTE	ACCION
ACTION_CALL	activity	Iniciar una llamada telefonica
ACTION_EDIT	activity	Visualizar datos para editarlos
ACTION_MAIN	activity	Arranque de una Activity para entrada de datos
ACTION_SYNC	activity	Sincroniza Datos de unServidor con un Dispositivo Movil
ACTION_BATTERY_LOW	broadcast receiver	Mensaje de Bateria Baja
ACTION_HEADSET_PLUG	broadcast receiver	Conexión o desconexión de un Auricular
ACTION_SCREEN_ON	broadcast receiver	Pantalla Encendida
ACTION_TIMEZONE_CHANGED	broadcast receiver	Cambio de la Configuracion de Zona Horaria

7.3. APLICACIÓN DEL USO DE INTENTS

Cree una aplicación que permita demostrar el uso de intents invocando a:

- URL,
- YouTube
- calculadora
- cámara.

Después de haber creado la primera aplicación para Android, elabore un nuevo proyecto seleccionando ARCHIVO > NUEVO > ANDROID PROYECTO, el nombre de proyecto: Ejemplo_Intents (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 1: Diseñe la vista de los elementos de la lista en res/layout/main.xml.

Paso 2: Edite el archivo main.xml y añada lo siguiente:

```
<?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:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="@string/hello" />

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/boton1" />

        <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/boton2" />

        <Button
        android:id="@+id/button3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/boton3" />

        <Button
        android:id="@+id/button4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/boton4" />

</LinearLayout>
```

Paso 3: Modifique Ejemplo_piscoActivity.java.

```
package com.nolascov.Intens1;
import java.io.File;
import android.app.Activity;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.os.Environment;
import android.provider.MediaStore;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;

public class IntensActivity extends Activity implements OnClickListener {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        Button b =  (Button) findViewById(R.id.button1);
        b.setOnClickListener(this);

        Button c =  (Button) findViewById(R.id.button2);
        c.setOnClickListener(this);


        Button d =  (Button) findViewById(R.id.button3);
        d.setOnClickListener(this);

        Button e =  (Button) findViewById(R.id.button4);
        e.setOnClickListener(this);
    }

    public void onClick(final View clicked) {
            Intent i;

              switch(clicked.getId()) {
          case R.id.button1:
                    i = new Intent(Intent.ACTION_VIEW,Uri.parse("http://www.nolascov.
com/moodle/"));
                    startActivity(i);
          break;
          case R.id.button2:
                    i = new Intent(Intent.ACTION_VIEW,Uri.parse("http://www.youtube.
com/watch?v=pZYTb1aVb6c"));
                    startActivity(i);
          break;

          case R.id.button3:

              i = new Intent();
                   i.setClassName("com.android.calculator2","com.android.calculator2.
Calculator");
              startActivity(i);
              break;


          case R.id.button4:
              String _path;
                 _path = Environment.getExternalStorageDirectory() + "/images/make_
machine_example.jpg";
```

```
                File file = new File( _path );
            Uri outputFileUri = Uri.fromFile( file );

            Intent  intent  =  new  Intent(android.provider.MediaStore.ACTION_IMAGE_
CAPTURE );
            intent.putExtra( MediaStore.EXTRA_OUTPUT, outputFileUri );

            startActivityForResult( intent, 0 );

             break;

                    }
        }

    }
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.*



Paso 2: *Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.



Paso 3: *Presione el botón Llamar a URL.*



Paso 4: *Presione el botón Llamar YOUTUBE.*



Paso 5: *Presione el botón Calculadora.*



Paso 6: *Presione el botón Cámara.*





Estilos y temas

8.1. APLICAR ESTILOS Y TEMAS EN ANDROID

Un estilo es un conjunto de propiedades que especifican el aspecto y el formato de un elemento de la interfaz. Un estilo puede especificar propiedades tales como altura, color de relleno de la fuente, tamaño de fuente, color de fondo, etc.

Un estilo se define dentro de un recurso XML. Desempeña parecidas funciones de los CSS en el diseño web. A continuación, se explicarán algunos estilos por medio de un ejemplo.

8.1.1. EjemploEstilo

Para crear una aplicación denominada EjemploEstilo, realice los siguientes pasos:

Paso 1: Una vez creada la primera aplicación para Android, cree un nuevo proyecto seleccionando Archivo > Nuevo > Android proyecto con el nombre de proyecto: EjemploEstilo (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 2: En la carpeta res/ values, cree un archivo XML.



Paso 3: Escriba el nombre del archivo: estilo y luego presione Finish.



Paso 4: Edite el estilo.xml para definir dos estilos:

estilo.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <style name="Texto">
        <item name="android:textSize">20sp</item>
        <item name="android:textColor">#AA44FF</item>
    </style>
    <style name= "Etiqueta">
        <item name="android:layout_width">fill_parent</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:textColor">#00FF10</item>
    </style>
</resources>
```

Paso 5: Modifique el main.xml.

main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
    <TextView
        style="@style/Etiqueta"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Etiqueta" />

    <EditText
        style="@style/Texto"
        android:id="@+id/editText1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        />
</LinearLayout>
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.*



Paso 2: *Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.





SQLite

9.1. ASPECTOS GENERALES



SQLite

SQLite es un sistema de gestión de bases de datos relacional compatible con ACID, contenida en una relativamente pequeña (~275 kiB) biblioteca escrita en C. SQLite es un proyecto de dominio público1 creado por D. Richard Hipp.

A diferencia de los sistema de gestión de bases de datos cliente-servidor, el motor de SQLite no es un proceso independiente con el que el programa principal se comunica. En lugar de eso, la biblioteca SQLite se enlaza con el programa pasando a ser parte integral del mismo. El programa utiliza la funcionalidad de SQLite a través de llamadas simples a subrutinas y funciones. Esto reduce la latencia en el acceso a la base de datos, debido a que las llamadas a funciones son más eficientes que la comunicación entre procesos. El conjunto de la base de datos (definiciones, tablas, índices, y los propios datos), son guardados como un sólo fichero estándar en la máquina host. Este diseño simple se logra bloqueando todo el fichero de base de datos al principio de cada transacción.

En su versión 3, SQLite permite bases de datos de hasta 2 Terabytes de tamaño, y también permite la inclusión de campos tipo BLOB.

El autor de SQLite ofrece formación, contratos de soporte técnico y características adicionales como compresión y cifrado.

CARACTERÍSTICAS

La biblioteca implementa la mayor parte del estándar SQL-92, incluyendo transacciones de base de datos atómicas, consistencia de base de datos, aislamiento, y durabilidad (ACID), triggers y la mayor parte de las consultas complejas.

SQLite usa un sistema de tipos inusual. En lugar de asignar un tipo a una columna como en la mayor parte de los sistemas de bases de datos SQL, los tipos se asignan a los valores individuales. Por ejemplo, se puede insertar un string en una columna de tipo entero (a pesar de que SQLite tratará en primera instancia de convertir la cadena en un entero). Algunos usuarios consideran esto como una innovación que hace que la base de datos sea mucho más útil, sobre todo al ser utilizada desde un lenguaje de scripting de tipos dinámicos. Otros usuarios lo ven como un gran inconveniente, ya que la técnica no es portable a otras bases de datos SQL. SQLite no trataba de transformar los datos al tipo de la columna hasta la versión 3.

Varios procesos o hilos pueden acceder a la misma base de datos sin problemas. Varios accesos de lectura pueden ser servidos en paralelo. Un acceso de escritura sólo puede ser servido si no se está sirviendo ningún otro acceso concurrentemente. En caso contrario, el acceso de escritura falla devolviendo un código de error (o puede automáticamente reintentarse hasta que expira un tiempo de expiración configurable). Esta situación de acceso concurrente podría cambiar cuando se está trabajando con tablas temporales. Sin embargo, podría producirse un interbloqueo debido al multihilo.[1] Este punto fue tratado en la versión 3.3.4, desarrollada el 11 de febrero de 2006.

Fuente: es.wikipedia.org

9.1.1. Versión actual

La versión 3.7.9 de SQLite se recomienda para todos los nuevos desarrollos. Ha habido una actualización desde la versión (3.7.6.3), (3.7.7), (3.7.7.1) y (3.7.8) es opcional.

9.2. Instalación de SQLite

En esta parte, se creará una base de datos SQLite que servirá en aplicaciones Android. Existen muchas maneras de manejarse con bases de datos SQLite; en este caso, se utilizará mediante FireFox (add-on de Firefox: SQLite Manager), compatible en principio con sus versiones en Mac OS, Windows y Linux.

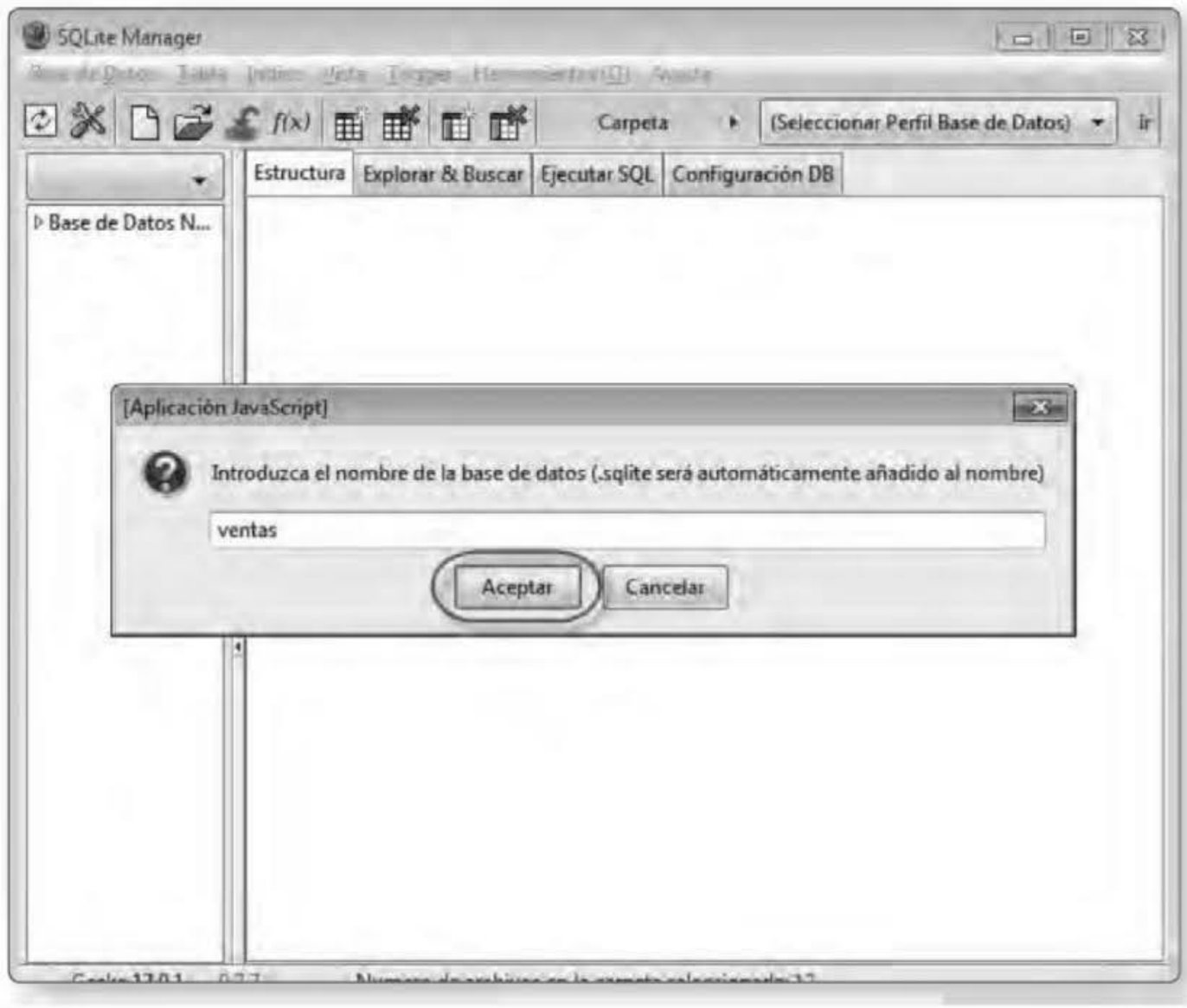
Paso 1: Para instalarlo, abra Firefox y diríjase a la siguiente dirección:



Paso 2: Diríjase a Herramientas > SQLite Manager.



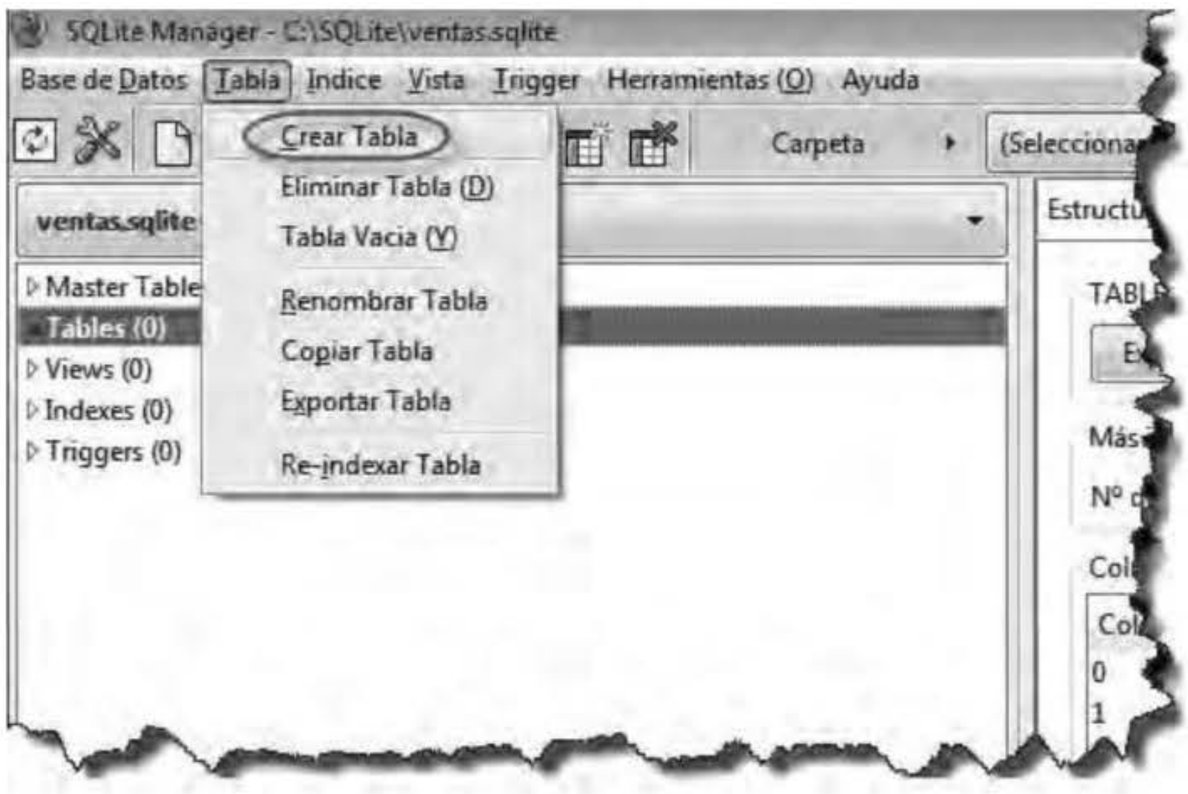
Paso 3: Cree una nueva base de datos y coloque un nombre: Ventas, por ejemplo.



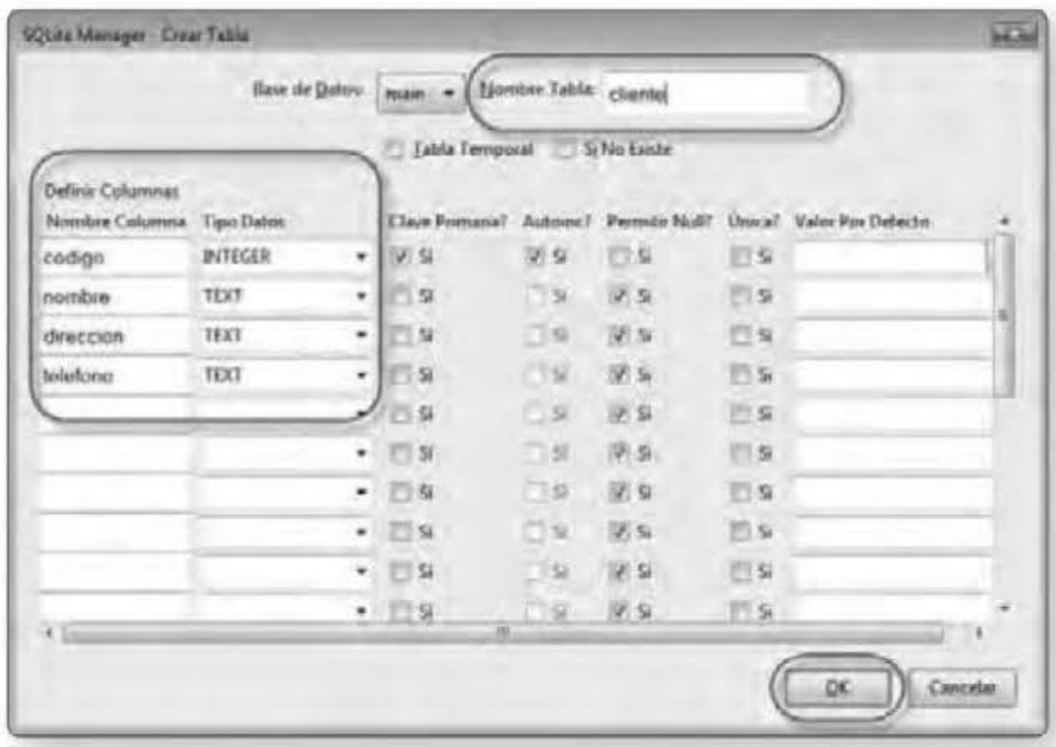
Paso 4: Cree la primera tabla la cual tendrá cuatro campos:

- código (integer–primary– autoincrement)
- nombre (TEXT)
- dirección (TEXT)
- teléfono (TEXT)

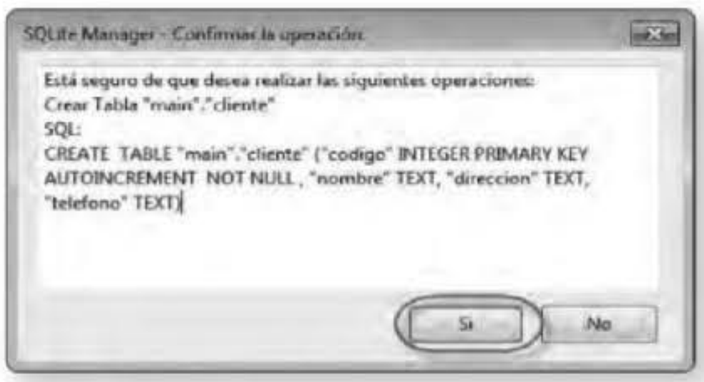
Paso 5: Para crear la tabla, seleccione a la izquierda Tables y con el botón derecho elija Crear tabla. Observe que la traducción al castellano está a medias; pero la funcionalidad está pulida.



Paso 6: Escriba los nombres de los campos.



Paso 7: Confirme la operación.



Existe un programa independiente de nombre «sqlite» que puede ser utilizado para consultar y gestionar los ficheros de base de datos SQLite. Asimismo, sirve como ejemplo para la escritura de aplicaciones utilizando la biblioteca SQLite.

9.3. EJEMPLO ALMACENAMIENTO DE DATOS

Ahora diseñe una aplicación que permita mostrar el uso de almacenamiento de datos.

Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando Archivo > Nuevo > Android proyecto.

Paso 2: Ingrese el nombre del proyecto: EjemploSQLite (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 3: Cree la siguiente interfaz:



Paso 4: En la carpeta res/values, edite el archivo strings.xml y defina las cadenas constantes en su aplicación.

strings.xml

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

    <string name="hello">Hello World, Form!</string>
    <string name="app_name">Ejemplo SQLite</string>
    <string name="mensaje">Nuevo Cliente</string>
    <string name="nom">Nombre</string>
    <string name="direccion">Direccion</string>
    <string name="tel">Telefono</string>
    <string name="cancel">Cancelar</string>
    <string name="add">Agregar</string>

</resources>
```

Nota:

La interfaz inicial creada, en este ejemplo, estará basada en una organización con tablas. El diseño se organizará en filas (elemento TableRow) y dentro de cada fila, se pondrá los elementos.



Paso 5: Edite el archivo main.xml y añada lo siguiente:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
<TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:focusable="false"
        android:gravity="center"
        android:text="@string/mensaje"
        android:textAllCaps="true"
        android:textSize="25px"
        android:textStyle="bold" />
<TableLayout
        android:id="@+id/tableLayout1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="0.27" >
<TableRow
            android:id="@+id/tableRow1" >
<TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:bufferType="editable"
```

```
                android:padding="10sp"
                android:text="@string/nom" />
<EditText
                android:id="@+id/nombre"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_weight="1" android:completionThreshold="3"/>
<requestFocus />
</TableRow>
<TableRow
            android:id="@+id/tableRow2" >
<TextView
                android:id="@+id/textView1"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:bufferType="editable"
                android:padding="6sp"
                android:text="@string/direccion" />
<EditText
                android:id="@+id/direccion"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_weight="1"/>
</TableRow>
<TableRow
            android:id="@+id/tableRow3" >
<TextView
                android:id="@+id/textView1"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:bufferType="editable"
                android:padding="9sp"
                android:text="@string/tel" />
<EditText
                android:id="@+id/telefono"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_weight="1" />
</TableRow>
<TableRow
            android:id="@+id/tableRow4" >
<Button
                android:id="@+id/cancel"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:onClick="onCancelarClick"
                android:text="@string/cancel" />

<Button
                android:id="@+id/add"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:onClick="onAgregarClick"
                android:text="@string/add" />
</TableRow>
</TableLayout>
</LinearLayout>
```

Paso 6: Indique qué sucederá al presionar el botón Cancel o Add:

```
<TableRow
    android:id="@+id/tableRow4" >

    <Button
        android:id="@+id/cancel"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="onCancelarClick"
        android:text="@string/cancel" />

    <Button
        android:id="@+id/add"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="onAgregarClick"
        android:text="@string/add" />
</TableRow>
</TableLayout>

</LinearLayout>
```

Al presionar el boton cancel se ejecuta el metodo onCancelarClick

Al presionar el boton add se ejecuta el metodo onAgregarClick

Paso 7: Agregue un par de cadenas constantes y edite el archivo strings.xml:

strings.xml

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

    <string name="hello">Hello World, Form!</string>
    <string name="app_name">Ejemplo SQLite</string>
    <string name="mensaje">Nuevo Cliente</string>
    <string name="nom">Nombre</string>
    <string name="direccion">Direccion</string>
    <string name="tel">Telefono</string>
    <string name="cancel">Cancelar</string>
    <string name="add">Agregar</string>
    <string name="titulo">Resultado</string>
    <string name="mensaje2">Exito</string>

</resources>
```

Paso 8: Edite la actividad inicial (Form) y muestre su codificación, que en adelante se explicará.

Form.java

```
package com.nolascov.Tutorial4;
import android.app.Activity;

public class Form extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }

    public void onCancelarClick(View botton)    {
        Intent intent = new Intent();
    intent.setComponent(new ComponentName(this,Form2.class));
    intent.putExtra("Cancel","Cancel");
    startActivity(intent);
    }

    public void onAgregarClick(View botton)    {
        EditText xnombre =  (EditText)   findViewById(R.id.nombre);
        EditText xdireccion =  (EditText)  findViewById(R.id.direccion);
        EditText xtelefono =  (EditText)   findViewById(R.id.telefono);

        Intent intent = new Intent();
        intent.setClass(this,Form2.class);

        intent.putExtra("nombre", xnombre.getText().toString());
        intent.putExtra("direccion",xdireccion.getText().toString());
        intent.putExtra("telefono",xtelefono.getText().toString());
        startActivity(intent);
    }
}
```

- Al iniciar la actividad Form

```
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
// lee el xml main para mostrar una pantalla.
setContentView(R.layout.main);
    }
```

- Al presionar el botón Cancelar

```
public void onCancelarClick(View botton)   {
// Un intent es una clase que permite especificar una activity a ejecutar.
Intent intent = new Intent();
//indicamos a que actividad queremos ir: Form2
intent.setComponent(new ComponentName(this,Form2.class));
//guarda la cadena Cancel en la etiqueta Cancel para ser utilizada en la actividad:
Form2
intent("Cancel","Cancel");
//comensamos la actividad Form2
startActivity(intent);
}
```

- Al presionar el botón Agregar

```
public void onAgregarClick(View botton)   {
// Accedemos a los valores nombre, dirección y teléfono y lo guardamos en 3 variables
locales
// xnombre , xdireccion y xtelefono
    EditText xnombre =  (EditText)   findViewById(R.id.nombre);
    EditText xdireccion =  (EditText)  findViewById(R.id.direccion);
    EditText xtelefono =  (EditText)   findViewById(R.id.telefono);
// Un intent es una clase que permite especificar una activity a ejecutar.
//indicamos a que actividad queremos ir: Form2
    Intent intent = new Intent();
    intent.setClass(this,Form2.class);
//guardamos las variables en 3 etiquetas para utilizarlos en la actividad Form2
    intent.putExtra("nombre", xnombre.getText().toString());
    intent.putExtra("direccion",xdireccion.getText().toString());
    intent.putExtra("telefono",xtelefono.getText().toString());
//comensamos la actividad Form2
    startActivity(intent);
    }
```

Paso 9: Cree un nuevo XML cuyo resultado se mostrará en un mensaje de éxitos más los datos del cliente o un mensaje Cancelar.

resultado.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <TextView android:layout_width="fill_parent"
        android:layout_height="wrap_content" android:textStyle="bold"
        android:focusableInTouchMode="false" android:textSize="25sp"
        android:padding="10sp" android:gravity="center" android:text="@string/titulo" />
    <TextView android:layout_width="fill_parent"
        android:layout_height="fill_parent" android:textStyle="bold"
        android:focusableInTouchMode="false" android:textSize="20sp"
        android:padding="10sp" android:gravity="center" android:text=""
        android:id="@+id/resultText" android:inputType="textMultiLine"/>
</LinearLayout>
```

Paso 10: Cree la actividad Form2 y muestre su codificación, que se explicará a continuación:

Form2.java

```
package com.nolascov.Tutorial4;
import com.nolascov.Tutorial4.R;

public class Form2 extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.resultado);

        TextView resultText = (TextView) findViewById(R.id.resultText);
        Bundle bundle = getIntent().getExtras();

        if (bundle.getString("Cancel") != null) {
            resultText.setText(getString(R.string.cancel));
        } else {
            String xnombre = bundle.getString("nombre");
            String xdireccion = bundle.getString("direccion");
            String xtelefono = bundle.getString("telefono");
            insertBook(xnombre, xdireccion ,xtelefono);
            resultText.setText(getString(R.string.mensaje2) +
                    "\n" + xnombre
                    + "\n" + xdireccion + "\n" + xtelefono);
        }
    }

}
```

- Al iniciar la actividad Form2

```
public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
// Lee el xml main para mostrar una pantalla.
            setContentView(R.layout.resultado);
// Capturamos el TextView resultText
            TextView resultText = (TextView) findViewById(R.id.resultText);
// Obtenemos las cadenas desde Form
            Bundle bundle = getIntent().getExtras();
// si existe la cadena Cancel significa que presionamos el botón Cancelar
            if (bundle.getString("Cancel") != null) {
                    resultText.setText(getString(R.string.cancel));
            }
//de lo contrario presionamos el botón agregar
else {
//obtenemos los valores de nombre, dirección y teléfono y los almacenamos en las
variables
                    String xnombre = bundle.getString("nombre");
                    String xdireccion = bundle.getString("direccion");
                    String xtelefono = bundle.getString("telefono");
// Mostramos en resultText los datos insertados
                    resultText.setText(getString(R.string.mensaje2) + "\n" +
                    "Nombre : " + xnombre + "\n" +
                    "Direccion :" + xdireccion + "\n" +
                    "Telefono : " + xtelefono);

            }
    }
```

Antes de ejecutar la aplicación en Android, existe un archivo XML llamado «AndroidManifest»; aunque no forme parte del código principal de la aplicación, es necesario para su correcto funcionamiento. Este archivo es el fichero de control que le dice al sistema qué tiene que hacer con todos los componentes; para ello, agregue la nueva actividad al archivo AndroidManifest:

EjemploSQLite Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.nolascov.Tutorial4"
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk android:minSdkVersion="15" />
    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name" >
        <activity
            android:label="@string/app_name"
            android:name=".Form" >
            <intent-filter >
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:label="@string/titulo"
            android:name=".Form2" >
        </activity>
    </application>
</manifest>
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:*



Paso 2: *Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.*



La figura muestra la aplicación que se ejecuta en el emulador de Android.



Paso 3: *Si presiona el botón Agregar.*



Paso 4: *Si presiona el botón Cancelar.*



Paso 5: *Escriba el código necesario para guardar la información; para ello, cree una clase para definir la estructura de la base de datos ventas la cual será explicada a continuación:*

ManejoDatabase.java

```
import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class ManejoDatabase extends SQLiteOpenHelper {
    private static final String BASEDATOS = "ventas.db";
    public static final String NOMBRE = "nombre";
    public static final String DIRECCION = "direccion";
    public static final String TELEFONO = "telefono";

    public ManejoDatabase(Context context) {
        super(context, BASEDATOS, null, 1);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL( "CREATE TABLE cliente (codigo INTEGER PRIMARY KEY  AUTOINCREMENT  NOT NULL , nombre TEXT, direccion TEXT, telefono TEXT);");
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        android.util.Log.w("cliente","Upgrading database, which will destroy all old data");
        db.execSQL("DROP TABLE IF EXISTS cliente");
        onCreate(db);
    }
}
```

- *Explicacion de la clase ManejoDatabase*

```
// Crear la clase ManejoDatabase  que SQLiteOpenHelper

public class ManejoDatabase extends SQLiteOpenHelper {
// Nombre de la base de datos que se creara si no existe
    private static final String BASEDATOS = “ventas.db”;
// Creamos algunas constantes con los Nombre de los campos de la tabla cliente
    public static final String NOMBRE = “nombre”;
    public static final String DIRECCION = “direccion”;
    public static final String TELEFONO = “telefono”;
// Constructor de la clase
    public ManejoDatabase(Context context) {
// Manejo de Numero de versión de la BD
            super(context, BASEDATOS, null, 1);
    }
// Cuando se crea por primera vez la base de datos se llamara al método onCreate
//donde se creara la Base de Datos Ventas y la Tabla cliente
    @Override
    public void onCreate(SQLiteDatabase db) {
            db.execSQL(     “CREATE  TABLE  cliente  (codigo  INTEGER  PRIMARY  KEY
AUTOINCREMENT  NOT NULL , nombre TEXT, direccion TEXT, telefono TEXT);”);
    }
// Método onUpgrade() se ejecuta cuando se actualizan la base de datos a una nueva
versión
// (elimina la existente y crea una nueva)
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
            android.util.Log.w(“cliente”,”Upgrading database, which will destroy all
old data”);
            db.execSQL(“DROP TABLE IF EXISTS cliente”);
            onCreate(db);
    }
}
```

- *Utilizacion de la clase ManejoDatabase: Para ello, se modificará la actividad Form2.*

Form2.java

```
public class Form2 extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.resultado);

        TextView resultText = (TextView) findViewById(R.id.resultText);
        Bundle bundle = getIntent().getExtras();

        if (bundle.getString("Cancel") != null) {
            resultText.setText(getString(R.string.cancel));
        } else {
            String xnombre = bundle.getString("nombre");
            String xdireccion = bundle.getString("direccion");
            String xtelefono = bundle.getString("telefono");
            insertarCliente(xnombre, xdireccion ,xtelefono);

            resultText.setText(getString(R.string.mensaje2) + "\n" +
                    "Nombre : " + xnombre + "\n" +
                    "Direccion :" + xdireccion + "\n" +
                    "Telefono : " + xtelefono);
        }
    }
    private void insertarCliente(String xnombre, String xdireccion, String xtelefono) {
        ManejoDatabase manejoDatabase = new ManejoDatabase(this);
        SQLiteDatabase db = manejoDatabase.getWritableDatabase();
        ContentValues cv = new ContentValues();
        cv.put(manejoDatabase.NOMBRE, xnombre);
        cv.put(manejoDatabase.DIRECCION, xdireccion);
        cv.put(manejoDatabase.TELEFONO, xtelefono);
        db.insert("cliente", manejoDatabase.NOMBRE, cv);
        db.close();
    }
}
```

- *Explicación de la llamada al método insertarCliente*

```
TextView resultText = (TextView) findViewById(R.id.resultText);
Bundle bundle = getIntent().getExtras();

if (bundle.getString("Can
    resultText.setText(ge
} else {
    String xnombre = bundle.
    String xdireccion = bundle.getString("direccion");
    String xtelefono = bundle.getString("telefono");
    insertarCliente(xnombre, xdireccion ,xtelefono);

    resultText.setText(getString(R.string.mensaje2) + "\n" +
            "Nombre : " + xnombre + "\n" +
            "Direccion :" + xdireccion + "\n" +
            "Telefono : " + xtelefono);
}
```

LLamamos al metodo insertarCliente pasamos como paremetros xnombre,xdireccion y xtelefono

- *Explicacion del método InsertarCliente*

```
private void insertarCliente(String xnombre, String xdireccion, String xtelefono) {
// Creamos una instancia de la clase ManejoDatabase
            ManejoDatabase manejoDatabase = new ManejoDatabase(this);
// Abrimos la Base de Datos para escritura
            SQLiteDatabase db = manejoDatabase.getWritableDatabase();
// Creamos un ContentValues para insertar en cada campo de la tabla sus valores
            ContentValues cv = new ContentValues();
            cv.put(manejoDatabase.NOMBRE, xnombre);
            cv.put(manejoDatabase.DIRECCION, xdireccion);
            cv.put(manejoDatabase.TELEFONO, xtelefono);
//Ahora el objeto cv está preparado para insertarse en la BD -  Tabla cliente
            db.insert(“cliente”, manejoDatabase.NOMBRE, cv);
// Cerramos la Base de Datos
            db.close();
    }
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.*



Paso 2: Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.



La figura muestra la aplicación que se ejecuta en el emulador de Android



Paso 3: *Si presiona el botón Agregar.*



Paso 4: *Si presiona el botón Cancelar.*





MySQL

10.1. Aspectos generales

MySQL

MySQL es un sistema de gestión de bases de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. MySQL AB —desde enero de 2008 una subsidiaria de Sun Microsystems y ésta a su vez de Oracle Corporation desde abril de 2009— desarrolla MySQL como software libre en un esquema de licenciamiento dual.

Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en ANSI C.

Al contrario de proyectos como Apache, donde el software es desarrollado por una comunidad pública y los derechos de autor del código están en poder del autor individual, MySQL es patrocinado por una empresa privada, que posee el copyright de la mayor parte del código.

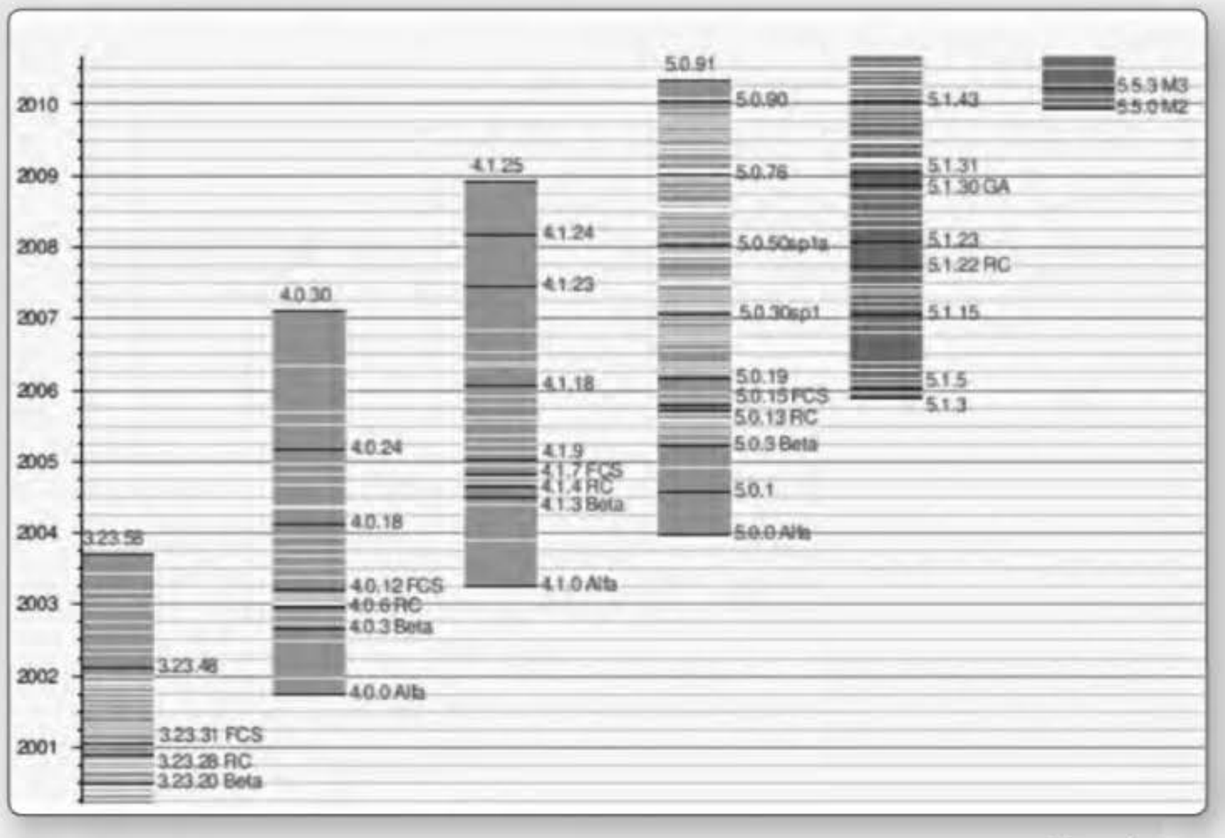
Esto es lo que posibilita el esquema de licenciamiento anteriormente mencionado. Además de la venta de licencias privativas, la compañía ofrece soporte y servicios. Para sus operaciones contratan trabajadores alrededor del mundo que colaboran vía Internet. MySQL AB fue fundado por David Axmark, Allan Larsson y Michael Widenius.

APLICACIONES

MySQL es muy utilizado en aplicaciones web, como Drupal o phpBB, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL.

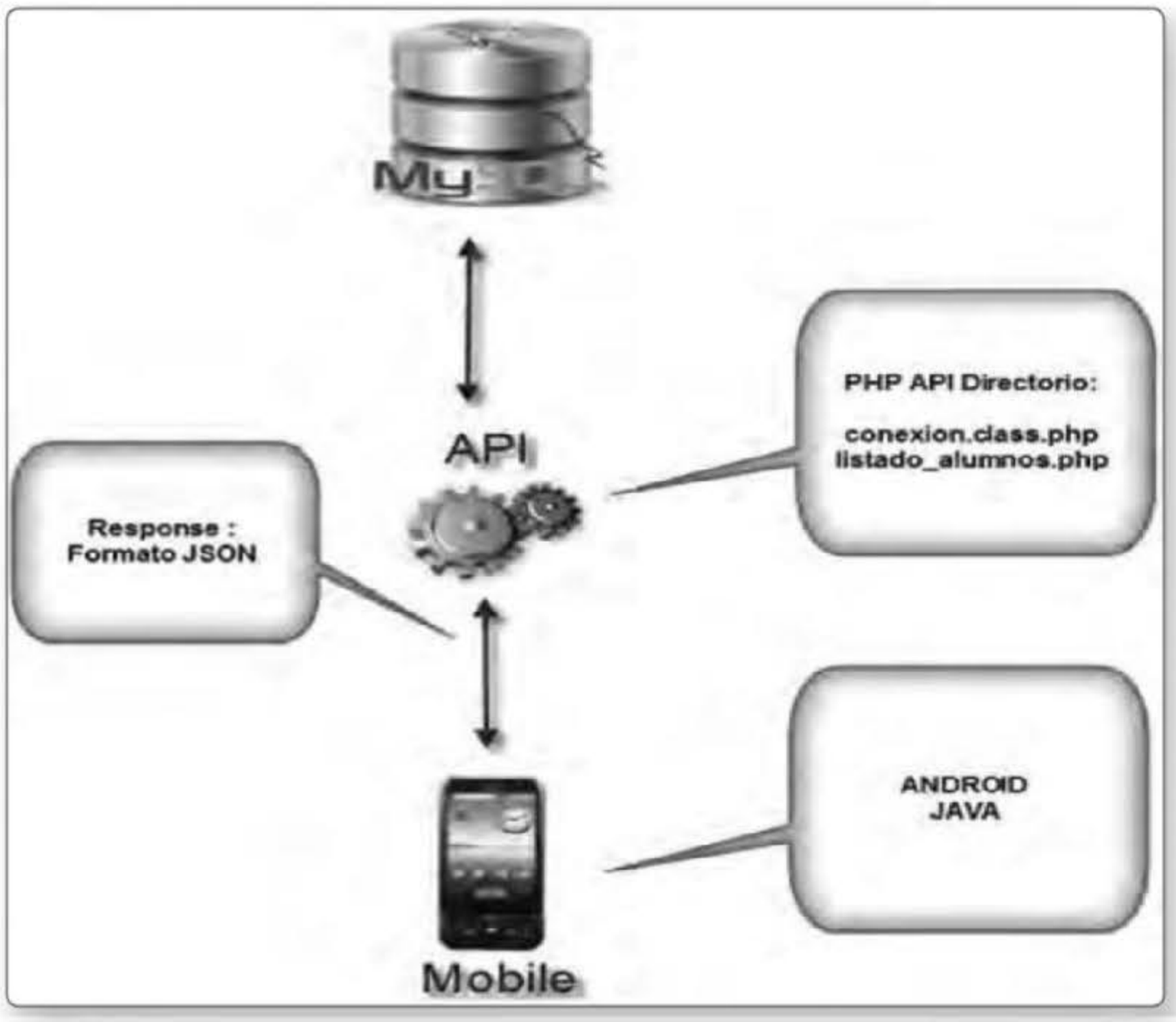
MySQL es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones. Sea cual sea el entorno en el que va a utilizar MySQL, es importante monitorizar de antemano el rendimiento para detectar y corregir errores tanto de SQL como de programación.

Versiones de MySQL



Fuente: es.wikipedia.org

Seguidamente, se presentará la manera cómo trabaja MySQL mediante el siguiente gráfico:



Para diseñar una aplicación que permita mostrar el uso del almacenamiento de datos con MySQL, primero debe crear su propio proyecto y tener en cuenta algunos aspectos (ver siguientes apartados).

10.1.1. Requisitos de software adicional necesario

- Servidor Web (Apache): El servidor web es el que se encarga de recibir peticiones de páginas y responder en páginas de tipo HTML (donde se instalará su módulo PHP).
- Modulo PHP (PHP): Permite leer el código en PHP y mediante el servidor web, mostrarlo en HTML (versión 5).
- Base de datos (MySQL): No va a servir para almacenar cierta informacion de una base de datos (versión 5).
- Editor de texto (Bloc de notas, Dreamweaver, TextMate): Se necesita un editor de texto para escribir el código PHP.

10.1.2. Configurar un servidor web localmente

Paso 1: Descargue desde cualquiera de las siguientes fuentes para instalar:

Servidor Web Apache + Modulo PHP + Base de Datos MySQL + phpMyAdmin

Paso 2: Descargue el AppServ desde:

http://www.appservnetwork.com/

AppServ 2.5.10 and 2.6.0 with PHP6 New Released !!!
Posted by apples on Saturday, May 10 @ 00:59:41 ICT (520726 reads)

AppServ 2.5.10

- Apache 2.2.8
- PHP 5.2.6
- MySQL 5.0.51b
- phpMyAdmin-2.10.3

Download
Sourceforge.net : http://prdownloads.sourceforge.net/appserv/appserv-win32-2.5.10.exe?download

MD5SUM : 279c0c39866fbecb8a3904969fd5d0f4

Paso 3: Una vez descargado, ejecute el instalador del AppServ:

- *Ventana de bienvenida de la instalación:* Haga clic en Next.



- *Carpeta de destino:* Se observa la carpeta o unidad destino donde se instalará el AppServ. Luego presione clic en Next.



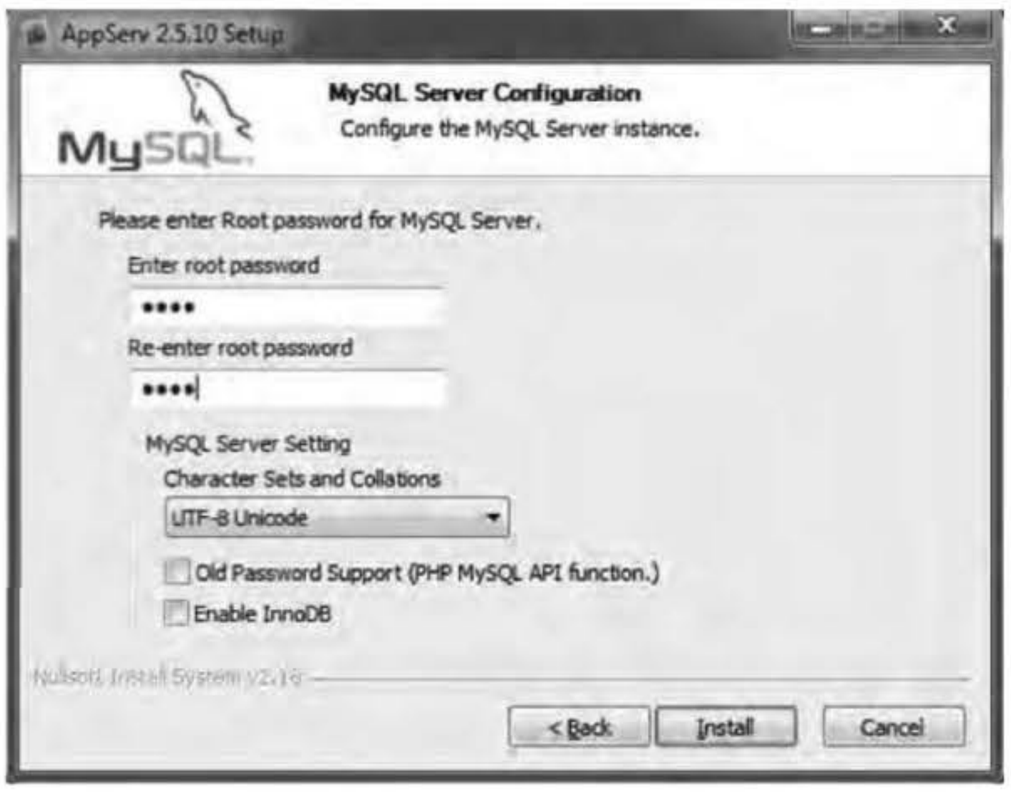
- *Componentes del paquete de AppServ:* Se presentan los cuatro componentes necesarios para realizar su trabajo. Haga clic en Next.



- *Sobre el puerto:* Respecto al puerto es usual utilizar el puerto por defecto 80. Luego haga clic en Next.



- *Sobre la configuración:* El UTF-8 U es el juego de caracteres que sirve para evitar conflictos con los acentos o caracteres especiales. Para ello, seleccione UTF-8 Unicode (modelo estándar más difundido). Luego haga clic en Install.



- Haga clic en Finish y automáticamente se iniciará Apache y mySQL.



- Para probar la instalación, digite en el explorador (Internet Explorer) lo siguiente:



MySQL

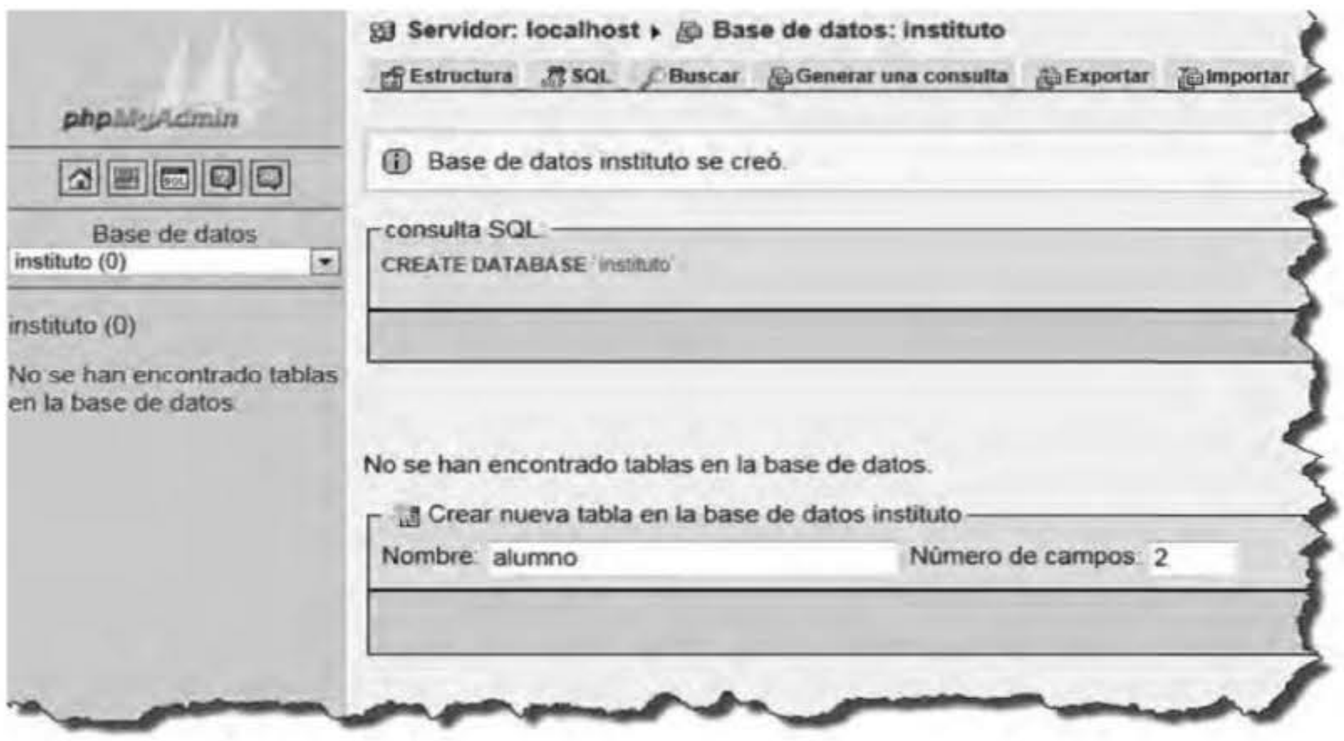
Paso 4: Cree la base de datos a utilizar.



Paso 5: Cree la base de datos instituto.



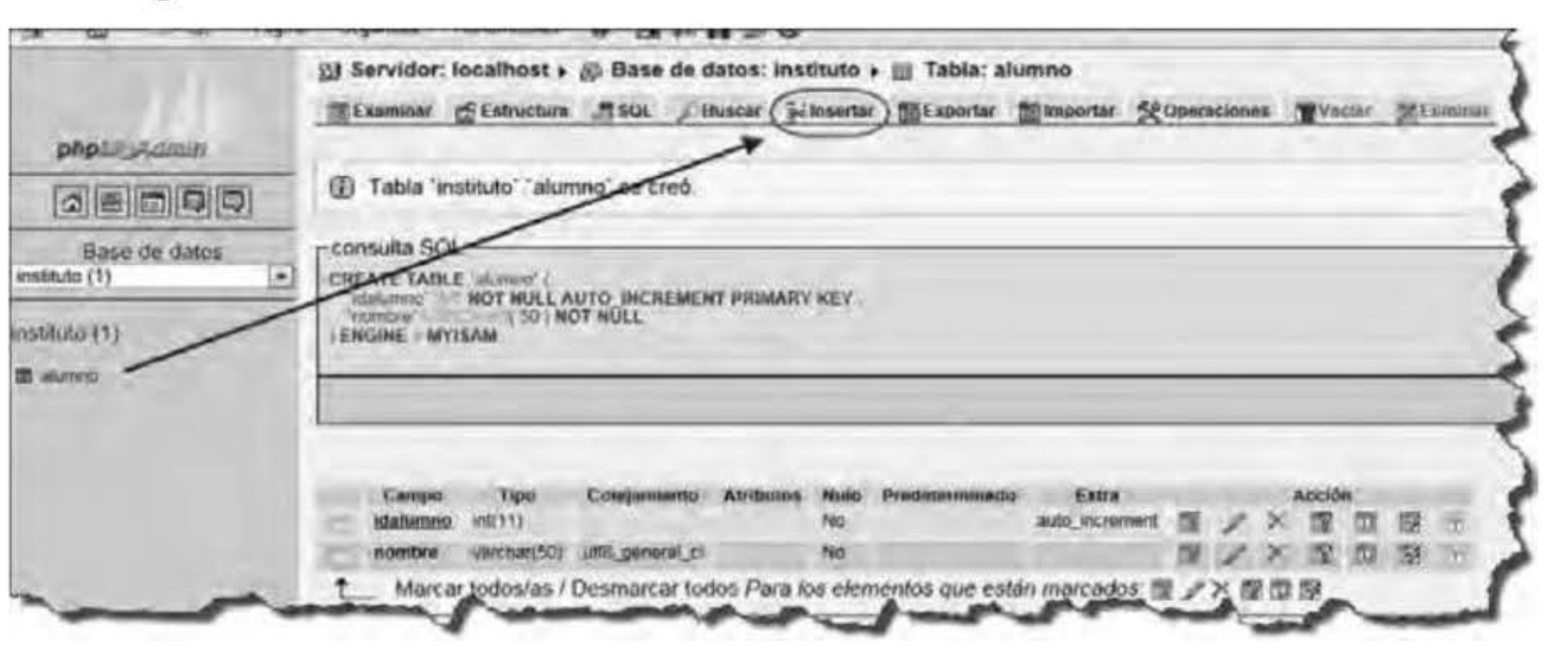
Paso 6: Cree la tabla "Alumno", insertando algunos registros:



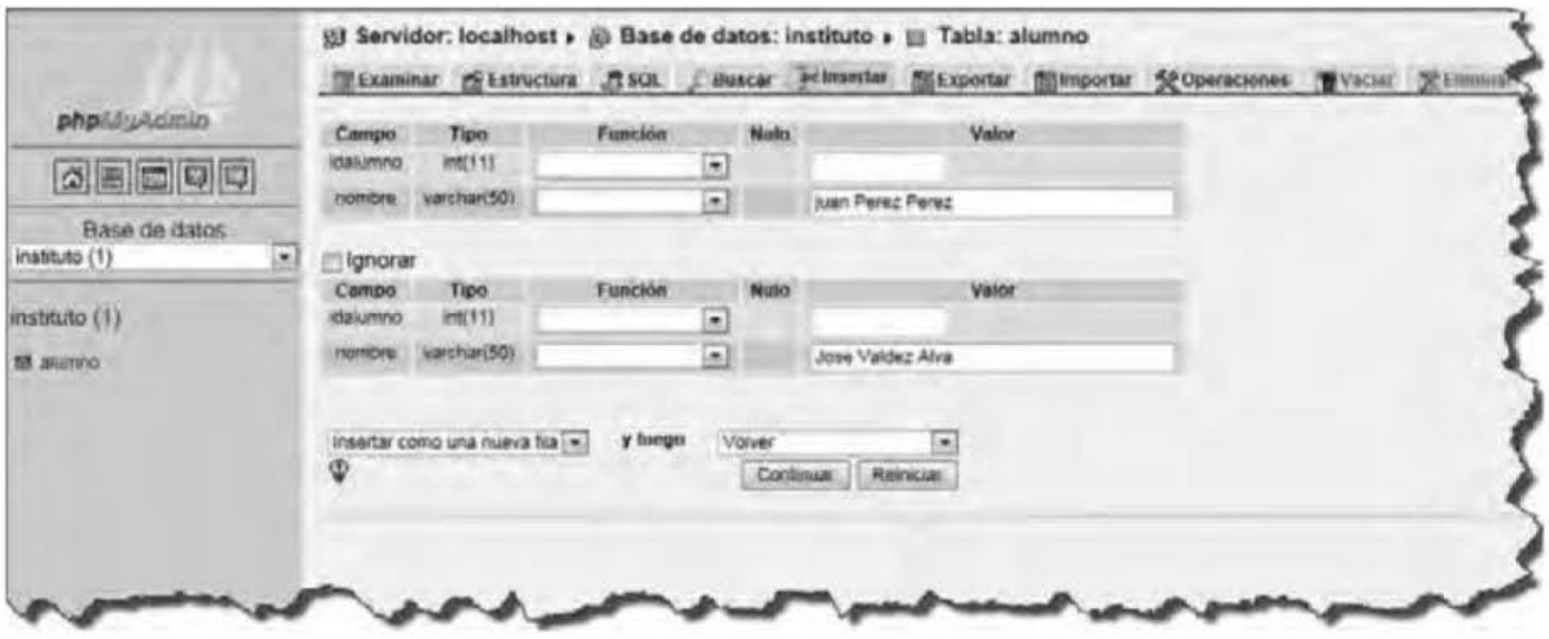
Paso 7: Cree la estructura de la tabla "Alumno", con los siguientes campos:



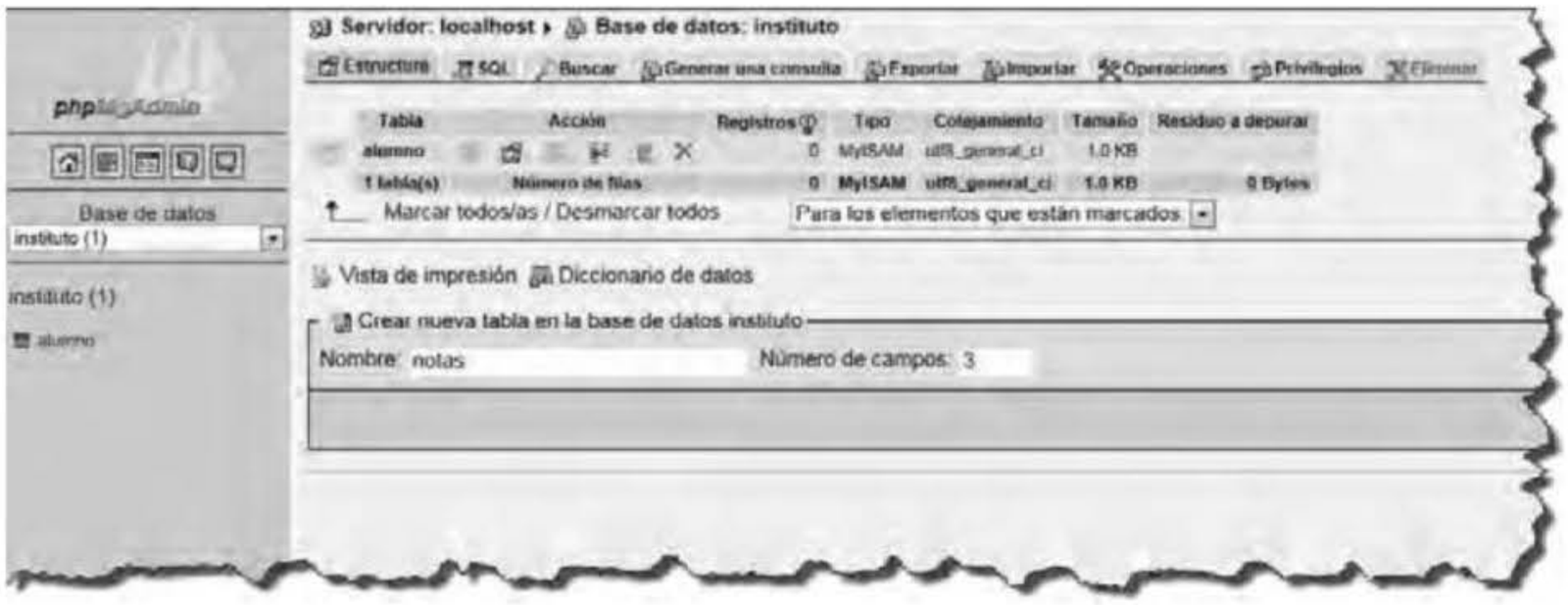
Paso 8: Inserte registros en la tabla "Alumnos".



Paso 9: Aproximadamente dos registros.

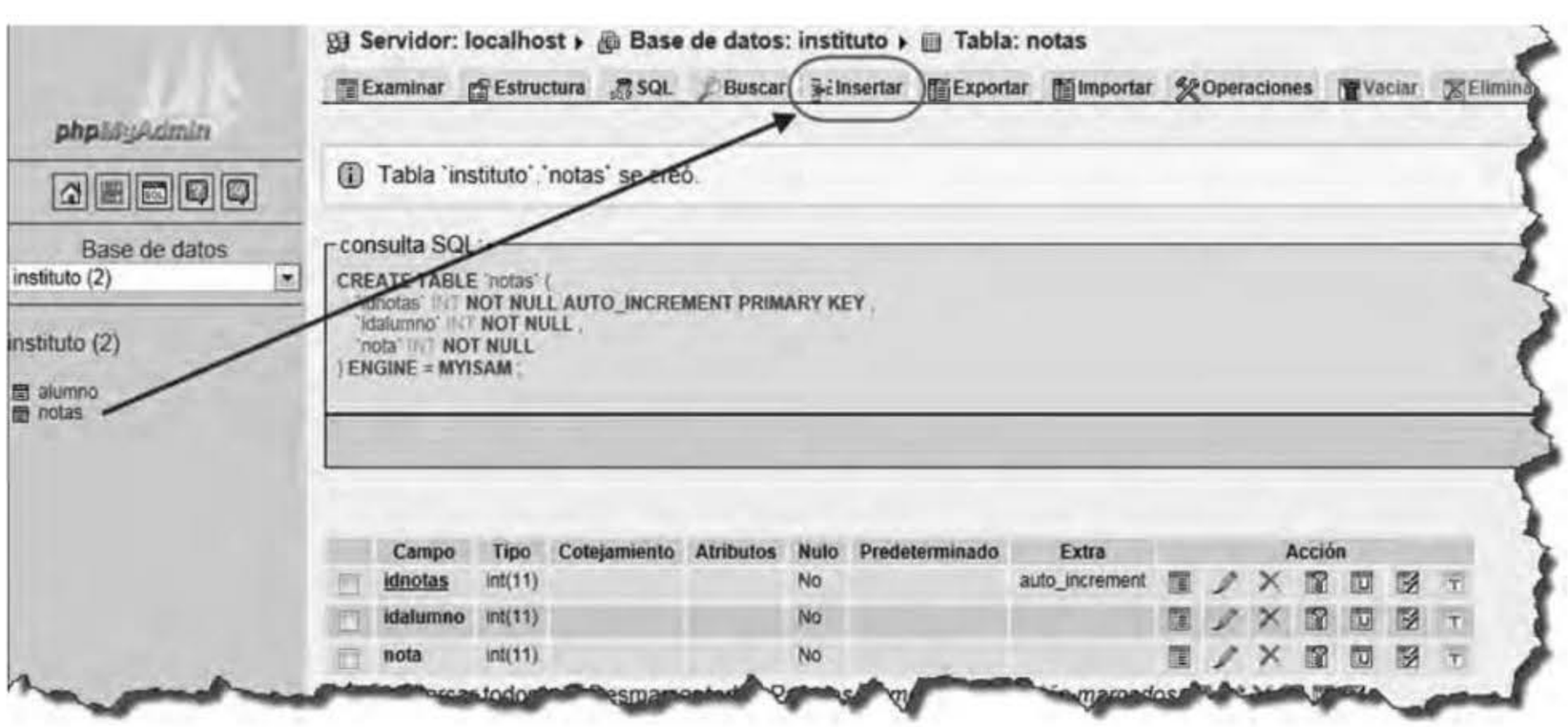


Paso 10: Cree la tabla "Notas".

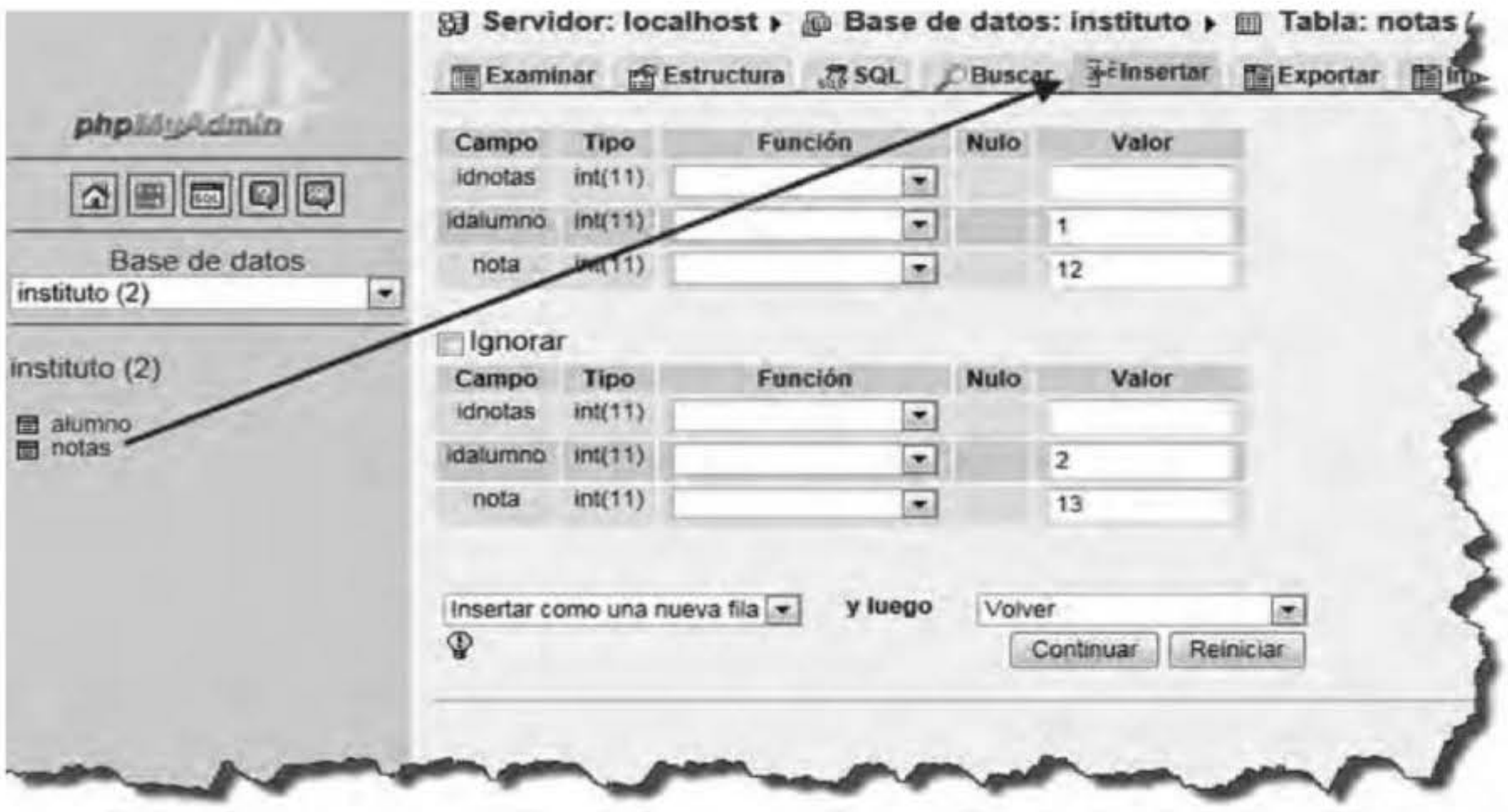


Paso 11: Cree la estructura de la tabla "Calificación" con los siguientes campos.

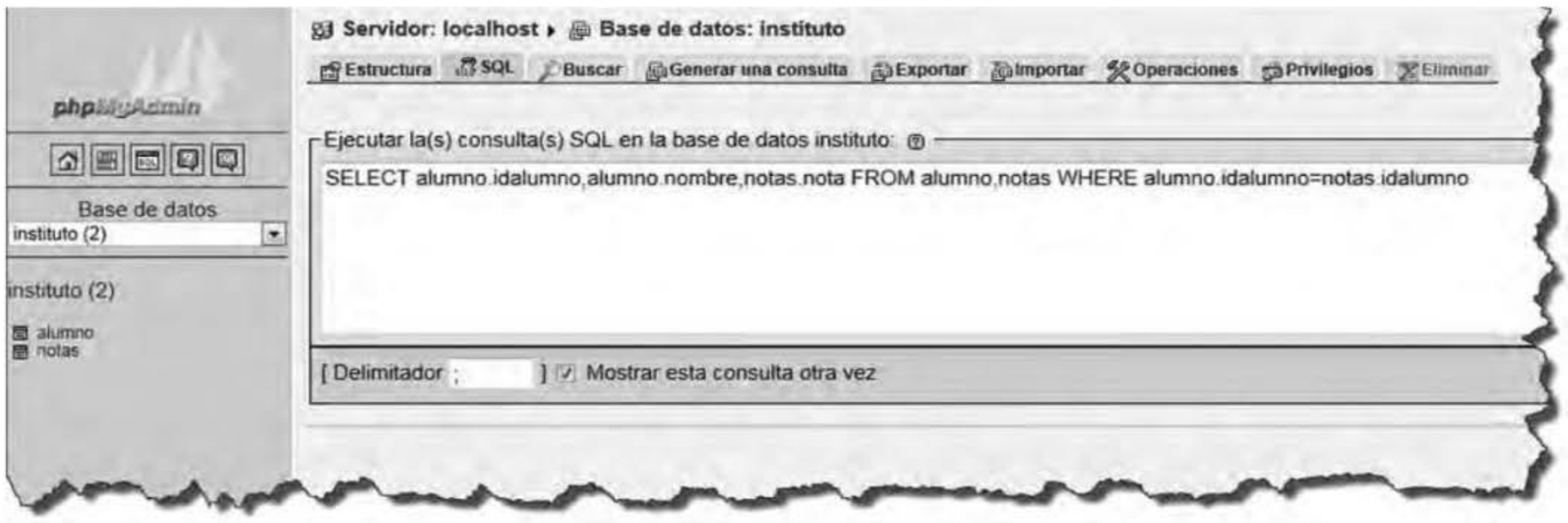




Paso 12: Inserte algunos registros, aproximadamente seis registros.



Paso 13: Indique la sentencia SQL Ansi para seleccionar los alumnos con sus repectivas calificaciones:





10.2. PHP

Cree una aplicación en PHP que permita realizar todas las tracciones necesarias.

10.2.1. Codificación de la clase

Primero se codifica los atributos necesarios para manejar la conexión con la base de datos y los métodos necesarios para conectarse con el servidor de la base de datos (Localhost o la IP 127.0.0.1). Segundo, seleccione la base de datos y, al finalizar, opte por el método para cerrar la conexión con la base de datos.

conexion.class.php

```
<?php
// creando la clase ManejoBD para manejar la base de datos
class ManejoBD{
     var $conect;  #atributo que contendra la conexion con al base de datos
     var $BaseDatos; #atributo para el nombre de la base de datos
     var $Servidor;  #atributo para el nombre del servidor de  base de datos
     var $Usuario;  #atributo para el nombre del usuario
     var $Clave;  #atributo para la clave del usuario
     //creando el metodo constructor de la clase para asignar valores a las propiedades
     function ManejoBD(){
             $this->BaseDatos = "instituto"; # aignar valor a la propiedad $BaseDatos
             $this->Servidor = "localhost"; # aignar valor a la propiedad $Servidor
             $this->Usuario = "root";  # aignar valor a la propiedad $Usuario
             $this->Clave = "1234"; # aignar valor a la propiedad $Clave
     }
     /* creando el metodo conectar para conectarme con el Servidor de Base de Datos
        y seleccionar la base de datos */
      function conectar()
      {
             //conectarme con el servidor de base de datos
             $con=mysql_connect($this->Servidor,$this->Usuario,$this->Clave);
             // si hubo problemas al conectarme con el servidor de base de datos
             if (!$con)
             {
                     die("Error al conectarme con el Servidor de Base de Datos");
             }
             //Seleccionar la base de datos
             $bd=mysql_select_db($this->BaseDatos,$con);
             //error al seleccionar la base de datos
             if (!$bd)
             {
                     die("Error al seleccionar la Base de Datos");
             }
             //almaceno la conexion en la propiedad conect
             $this->conect=$con;
             //devuelvo la conexion desde donde fue invocada
             return true;
      }

       function cerrar() {
          // cerrando la conexion
          mysql_close();
      }
}

?>
```

listado_alumnos.php

```
<?php
/*
 * Autor : jorge nolasco valenzuela
 * Programa : listado_alumno.php
 * Funcionalidad : lista los alumnos y sus respectivas calificaciones
 */

// array para JSON
$response = array();


// incluir la clase conexion.class.php
require_once("conexion.class.php");

// conectandose con el servidor de BD
$db = new ManejoBD();
// Seleccionar la BD instituto
$db->conectar();

// recuperar los alumnos y sus calificaciones mediante sentencia ANSI
$result = mysql_query("SELECT alumno.idalumno,alumno.nombre,notas.nota FROM alumno,notas
WHERE alumno.idalumno=notas.idalumno") or die(mysql_error());


// existe por lo menos 1 registro
if (mysql_num_rows($result) > 0) {

    $response["alumnos"] = array();

    while ($row = mysql_fetch_array($result)) {
        // llenar el arreglo con los registros

        $alumnos = array();
        $alumnos["idalumno"] = $row["idalumno"];
        $alumnos["nombre"] = $row["nombre"];
        $alumnos["nota"] = $row["nota"];

            /*
            echo $alumnos["idalumno"];
            echo $alumnos["nombre"];
            echo $alumnos["nota"];
         */

        array_push($response["alumnos"], $alumnos);
    }
    // exito
        $response["success"] = 1;
        // echo exito JSON
        echo json_encode($response);
    } else {
        // fracaso
        $response["success"] = 0;
        $response["message"] = "Fracaso";

        // echo no usuarios JSON
        echo json_encode($response);
    }
    ?>
```

Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando Archivo > Nuevo > Android proyecto.

Paso 2: Ingrese el nombre del proyecto: Ejemplo_MySql (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 3: Cree la siguiente interfaz:

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" >

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="28dp"
        android:text="Instituto Superior Tecnologico ABC"
        android:textSize="19dp" />

    <Button
        android:id="@+id/btnListar"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/textView1"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="22dp"
        android:text="Listado de Calificaciones" />

</RelativeLayout>
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:*



Paso 2: *Cree una interfaz donde se mostrará las calificaciones de los alumnos.*

listado_alumnos.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <ListView
        android:id="@+id/lista"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
    </ListView>

</LinearLayout>
```

Codificando MainScreenActivity.java

```
package com.nolascov.ejemplomysql;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
public class MainScreenActivity extends Activity{

    Button btnVerNotas;
    Button btnAddCalificacion;

    @Override
    public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.main_screen);

            // boton btnVerNotas ( ver notas )
            btnVerNotas= (Button) findViewById(R.id.btnVerNotas);
            // boton btnAddCalificacion ( ver calificacion )
            btnAddCalificacion = (Button) findViewById(R.id.btnAddCalificacion);

            // evento click de btnVerNotas ( ver notas )
            btnVerNotas.setOnClickListener(new View.OnClickListener() {

                    public void onClick(View view) {
                            // cargando Listado_calificacion
                            Intent i = new Intent(getApplicationContext(), Listado_
calificacion.class);
                            startActivity(i);

                    }
            });

            // evento click de btnAddCalificacion ( Agregar Calificaciones )
            btnAddCalificacion.setOnClickListener(new View.OnClickListener() {
```

```
                    public void onClick(View view) {
                            // cargando Nueva_calificacion
                            Intent  i  =  new  Intent(getApplicationContext(),Nueva_
calificacion.class);
                            startActivity(i);

                    }
            });
    }
}
```

10.3. JavaScript Object Notation (JSON)



La eficiencia en el uso de los recursos al momento de desarrollar aplicaciones móviles es muy importante; sobre todo en el caso del consumo de datos de Internet que se debe optimizar al máximo posible para ahorrar batería y aprovechar las tarifa de datos existentes actuales. En este sentido JSON, gracias a su estructura, ahorra considerablemente respecto a XML.

JSONParser.java

```
package com.example.ejemplomysql;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.util.List;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.client.utils.URLEncodedUtils;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONException;
import org.json.JSONObject;

import android.util.Log;

public class JSONParser {

    static InputStream is = null;
    static JSONObject jObj = null;
    static String json = "";

    // constructor
    public JSONParser() {

    }
    //function get json de url, con metodos HTTP POST o GET
    public JSONObject makeHttpRequest(String url, String method,
```

```
                List<NameValuePair> params) {

        // Haciendo peticiÃ³n HTTP
        try {

                // Comprobando el Metodo de peticion que se utilizara
                if(method == “POST”){
                        //Metodo POST , defaultHttpClient
                        DefaultHttpClient httpClient = new DefaultHttpClient();
                        HttpPost httpPost = new HttpPost(url);
                        httpPost.setEntity(new UrlEncodedFormEntity(params));

                        HttpResponse httpResponse = httpClient.execute(httpPost);
                        HttpEntity httpEntity = httpResponse.getEntity();
                        is = httpEntity.getContent();

                }else if(method == “GET”){
                        // Metodo  GET
                        DefaultHttpClient httpClient = new DefaultHttpClient();
                        String paramString = URLEncodedUtils.format(params, “utf-
8”);
                        url += “?” + paramString;
                        HttpGet httpGet = new HttpGet(url);

                        HttpResponse httpResponse = httpClient.execute(httpGet);
                        HttpEntity httpEntity = httpResponse.getEntity();
                        is = httpEntity.getContent();
                }


        } catch (UnsupportedEncodingException e) {
                e.printStackTrace();
        } catch (ClientProtocolException e) {
                e.printStackTrace();
        } catch (IOException e) {
                e.printStackTrace();
        }

        try {
                BufferedReader reader = new BufferedReader(new InputStreamReader(
                                is, “iso-8859-1”), 8);
                StringBuilder sb = new StringBuilder();
                String line = null;
                while ((line = reader.readLine()) != null) {
                        sb.append(line + “\n”);
                }
                is.close();
                json = sb.toString();
        } catch (Exception e) {
                Log.e(“Buffer Error”, “Error converting result “ + e.toString());
                }

                // intenta analizar la cadena en un objeto JSON
                try {
                        jObj = new JSONObject(json);
                } catch (JSONException e) {
                        Log.e(“JSON Parser”, “Error parsing data “ + e.toString());
                }

                // Retornando JSON String
                return jObj;

        }
}
```

Listado_calificacion.java

```
package com.example.ejemplomysql;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;

import org.apache.http.NameValuePair;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import android.app.ListActivity;
import android.app.ProgressDialog;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ListAdapter;
import android.widget.ListView;
import android.widget.SimpleAdapter;
import android.widget.TextView;

public class Listado_calificacion extends ListActivity {

    // barra de progreso
    private ProgressDialog pDialog;

    // creacion del objeto   JSON
    JSONParser jParser = new JSONParser();


    ArrayList<HashMap<String, String>> listado;
    //listado de calificaciones

    // url para obtener el listado de calificaciones
    private static String url_listado = "http://10.0.2.2/instituto/listado_alumnos.
php";

    //url_all_products

    // Nombre de Nodos JSON
    private static final String TAG_SUCCESS = "success";
    private static final String TAG_ALUMNO = "alumnos";
    private static final String TAG_NOMBRE = "nombre";
    private static final String TAG_NOTA = "nota";


    // calificacion JSONArray
    JSONArray calificacion = null;


    public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.all_products);
        //setContentView(R.layout.list_item);

        // Hashmap para el ListView
        listado = new ArrayList<HashMap<String, String>>();

        // cargando calificaciones en Background Thread
        new CargarCalificacion().execute();

        // obtener listview
        ListView lv = getListView();


        // Al seleccionar el alumno , se procedera la edicion
        lv.setOnItemClickListener(new OnItemClickListener() {
                public void onItemClick(AdapterView<?> parent, View view,
                                int position, long id) {
                        // obtener valores de la lista
                        String  pid  =  ((TextView)  view.findViewById(R.id.pid)).
getText()
                                        .toString();

                        // iniciar un nuevo  intent
                        Intent in = new Intent(getApplicationContext(),
                                        Editar_Calificacion.class);
                        // enviar el TAG_NOMBRE a la siguiente actividad
                        in.putExtra(TAG_NOMBRE, pid);

                        // empenzado la nueva actividad y espera alguna respuesta
                                startActivityForResult(in, 100);
                }
        });

}

// Respuesta desde la eidcion de la ctividad Product
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        // si el resultado del codigo es 100
        if (resultCode == 100) {
                //si el cÃ³digo de resultado 100 se recibiÃ³
                //: el usuario editar / eliminar producto
                //vuelve a cargar la pantalla de nuevo
                Intent intent = getIntent();
                finish();
                startActivity(intent);
        }

}


//Async cargar todo las calificaciones al hacer una solicitud HTTP
class CargarCalificacion extends AsyncTask<String, String, String> {

        //Antes de comenzar thread Mostrar cuadro de diÃ¡logo Progreso
        @Override
        protected void onPreExecute() {
                super.onPreExecute();
                ///CAMBIOS EN LA SIGUIENTE LINEA
                pDialog = new ProgressDialog(Listado_calificacion.this);
```

```
                    pDialog.setMessage("Cargando      Calificaciones...Espero      un
Momento...");
                    pDialog.setIndeterminate(false);
                    pDialog.setCancelable(false);
                    pDialog.show();
                }


                //obtener calificaciones desde la url
                protected String doInBackground(String... args) {
                        // Construyendo Parametros
                        List<NameValuePair> params = new ArrayList<NameValuePair>();
                        // consiguiendo la cadena  JSON desde URL
                        JSONObject  json  =  jParser.makeHttpRequest(url_listado,  "GET",
params);

                        // Check your log cat for JSON reponse
                        Log.d("All Products: ", json.toString());

                        try {
                                // comprobacion de for SUCCESS TAG
                                int success = json.getInt(TAG_SUCCESS);

                                if (success == 1) {
                                        // calificacion encontrada
                                        // obtener  Array de calificacion
                                        calificacion = json.getJSONArray(TAG_ALUMNO);

                                        // recorriendo todas las  calificacion
                                        for (int i = 0; i < calificacion.length(); i++) {
                                                JSONObject     c     =     calificacion.
getJSONObject(i);

                                                // almacenan el nombre y la nota en la
variable json
                                                String id = c.getString(TAG_NOMBRE);
                                                String name = c.getString(TAG_NOTA);

                                                System.out.println(id);
                                                System.out.println(name);

                                                // creando un nuevo HashMap
                                                HashMap<String,   String>   map   =   new
HashMap<String, String>();

                                                // aÃ±adiendo nodos clave y valor
                                                map.put(TAG_NOMBRE, id);
                                                map.put(TAG_NOTA, name);

                                                // aÃ±adiendo HashList al ArrayList
                                                listado.add(map);
                                        }
                                } else {
                                        // no calificaciones encontradas
                                        // inicia el Intent Nueva_calificacion
                                        Intent i = new Intent(getApplicationContext(),
                                                        Nueva_calificacion.class);
                                        // Cierra la actividad anterior
                                        i.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                                        startActivity(i);
```

```
                        }
                } catch (JSONException e) {
                        e.printStackTrace();
                }

                return null;
        }


        //DespuÃ©s de completar la tarea  cerra el diÃ¡logo de progreso
        protected void onPostExecute(String file_url) {
                // cerra el dialogo despues de recibir las calificaciones
                pDialog.dismiss();
                // actualzas las UI en segundo plano
                runOnUiThread(new Runnable() {
                        public void run() {
                                //ActualizaciÃ³n  datos JSON en el ListView
                                ///CAMBIOS EN LA SIGUIENTE LINEA
                                ListAdapter adapter = new SimpleAdapter(
                                                Listado_calificacion.this, listado,
                                                R.layout.list_item,  new  String[]  {
TAG_NOMBRE,
                                                                TAG_NOTA},
                                                new int[] { R.id.pid, R.id.name });
                                // actualizar el  listview
                                setListAdapter(adapter);
                        }
                });

                }
        }
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:*



10.3.1. Conceptos básicos relacionados

Con la creciente popularidad de los servicios web, XML se ha convertido prácticamente de facto en el estándar para la transmisión de datos; pero para realizar una tarea en Internet se necesita más bytes de información que bien se podría llevar a cabo con un flujo de información menor.

Al respecto, se han desarrollado nuevas formas de compresión XML así como nuevos formatos XML completos, tales como Binary XML (XML binario). Todas estas soluciones funcionan ampliando o añadiéndose a XML; conviniendo como tema relevante los aspectos de compatibilidad descendente. El Ing. Douglas Crockford propuso un nuevo formato de datos construido sobre la base de JavaScript llamado JSON (JavaScript Object Notation, 'notación de objetos JavaScript').

Por tanto, JSON es un formato ligero de datos basado en un subconjunto de la sintaxis de JavaScript (literales de matrices y objetos). Al usar la sintaxis JavaScript, las definiciones JSON pueden incluirse dentro de archivos JavaScript y acceder a ellas sin ningún análisis adicional como los necesarios con lenguajes basados en XML.

10.3.2. Sintaxis de JSON

La sintaxis de JSON es la mezcla de literales de objeto y matrices para almacenar datos. JSON representa solamente datos; por lo que no incluye el concepto de variables, asignaciones o igualdades.

Este código:

```
var oPersona3 =
[{"nombre":"Robert", "edad":30, "hijos":["Jaime","Pepe","Alfonso"] },
{"nombre":"Maria", "edad":36, "hijos":["Hijo Maria","Hijo2 Maria"] }];
```

Quedaría

```
[{"nombre":"Robert", "edad":30, "hijos":["Jaime","Pepe","Alfonso"] },
{"nombre":"Maria", "edad":36, "hijos":["Hijo Maria","Hijo2 Maria"] }]
```

En este caso, se ha eliminado la variable oPersona3, asf como el punto y coma del final. Si se transmite esta información a través de HTTP a un navegador, será bastante rápido, debido al reducido número de caracteres. Suponga que recupera esta informacion y la almacena en una variable llamada «sJSON». Ahora, dispondrá de una cadena de información, no de un objeto. Para transformarla en un objeto, simplemente utilice la función eval ():

```
var sJSON =
'[{"nombre":"Robert", "edad":30, "hijos":["Jaime","Pepe","Alfonso"] },
{"nombre":"Maria", "edad":36, "hijos":["Hijo Maria","Hijo2 Maria"] }]';

var oPersona4    = eval("(" + sJSON + ")");

alert(oPersona4[1].edad);
```

Es importante incluir los paréntesis adicionales alrededor de cualquier cadena JSON antes de pasarla a eval (). Recuerde que las llaves representan también sentencias en JavaScript (como en la sentencia if). La única forma que tiene el intérprete de saber que las llaves representan un objeto y no una sentencia es buscar un signo igual al principio o buscar paréntesis rodeando el texto lo cual indica que el código es una expresión que se tiene que evaluar en lugar de una sentencia para ejecutar.

10.3.1. Conceptos básicos relacionados

Con la creciente popularidad de los servicios web, XML se ha convertido prácticamente de facto en el estándar para la transmisión de datos; pero para realizar una tarea en Internet se necesita más bytes de información que bien se podría llevar a cabo con un flujo de información menor.

Al respecto, se han desarrollado nuevas formas de compresión XML así como nuevos formatos XML completos, tales como Binary XML (XML binario). Todas estas soluciones funcionan ampliando o añadiéndose a XML; conviniendo como tema relevante los aspectos de compatibilidad descendente. El Ing. Douglas Crockford propuso un nuevo formato de datos construido sobre la base de JavaScript llamado JSON (JavaScript Object Notation, 'notación de objetos JavaScript').

Por tanto, JSON es un formato ligero de datos basado en un subconjunto de la sintaxis de JavaScript (literales de matrices y objetos). Al usar la sintaxis JavaScript, las definiciones JSON pueden incluirse dentro de archivos JavaScript y acceder a ellas sin ningún análisis adicional como los necesarios con lenguajes basados en XML.

10.3.2. Sintaxis de JSON

La sintaxis de JSON es la mezcla de literales de objeto y matrices para almacenar datos. JSON representa solamente datos; por lo que no incluye el concepto de variables, asignaciones o igualdades.

Este código:

```
var oPersona3 =
[{"nombre":"Robert", "edad":30, "hijos":["Jaime","Pepe","Alfonso"] },
{"nombre":"Maria", "edad":36, "hijos":["Hijo Maria","Hijo2 Maria"] }];
```

Quedaría

```
[{"nombre":"Robert", "edad":30, "hijos":["Jaime","Pepe","Alfonso"] },
{"nombre":"Maria", "edad":36, "hijos":["Hijo Maria","Hijo2 Maria"] }]
```

En este caso, se ha eliminado la variable oPersona3, asf como el punto y coma del final. Si se transmite esta información a través de HTTP a un navegador, será bastante rápido, debido al reducido número de caracteres. Suponga que recupera esta informacion y la almacena en una variable llamada «sJSON». Ahora, dispondrá de una cadena de información, no de un objeto. Para transformarla en un objeto, simplemente utilice la función eval ():

```
var sJSON =
'[{"nombre":"Robert", "edad":30, "hijos":["Jaime","Pepe","Alfonso"] },
{"nombre":"Maria", "edad":36, "hijos":["Hijo Maria","Hijo2 Maria"] }]';

var oPersona4    = eval("(" + sJSON + ")");

alert(oPersona4[1].edad);
```

Es importante incluir los paréntesis adicionales alrededor de cualquier cadena JSON antes de pasarla a eval (). Recuerde que las llaves representan también sentencias en JavaScript (como en la sentencia if). La única forma que tiene el intérprete de saber que las llaves representan un objeto y no una sentencia es buscar un signo igual al principio o buscar paréntesis rodeando el texto lo cual indica que el código es una expresión que se tiene que evaluar en lugar de una sentencia para ejecutar.

10.3.3. Codificar y decodificar JSON

Como parte de los recursos de JSON, Crockford creó una utilidad que se puede emplear para (des) codificar objetos JSON y JavaScript. El código fuente de esta herramienta se puede descargar en la siguiente dirección:

http://www.json.org/js.html

En este apartado, en lugar de este código, se empleará el codificador/decodificador de MOOTOOLS. Cabe advertir que no se puede decodificar JSON, utilizando la función eval () porque hay un problema inherente en el uso de eval(). Este evalúa cualquier código JavaScript que se pasa a la función lo cual constituye un alto riesgo de seguridad.

A. MOOTOOLS JSON

Json facilita la tarea de pasar de un string Json a un objeto Javascript y viceversa.

> ***Nota:***
>
> *Todo lo referente a JSON respecto a la versión anterior (1.11) ha cambiado bastante, haciéndose completamente incompatibles. Por ejemplo, antes se escribía Json y ahora es JSON.*

B. JSON.ENCODE(OBJETO)

Permite pasar un objeto Javascript a un simple string.

```
var objeto = {
nombre: 'Mootools',
descripcion: 'Framework'
}

var str = JSON.encode(objeto);
//{"nombre":"Mootools","descripcion":"Framework"}
```

C. JSON.DECODE(STRING, SECURE)

Permite pasar de un string Json a un objeto Javascript.

```
var str = '{"nombre":"Mootools","descripcion":"Framework"}';
var objeto = JSON.decode(str);
alert(objeto.nombre);              //Mootools
alert(objeto.descripcion);         //Framework
```

Secure es un parámetro opcional que por defecto es false (falso), pero si se cambiase a true (verdadero), se chequeará la sintaxis del string Json. En caso de que el string sea incorrecto, Json. decode devolverá null (nulo).

```
var str = '{"nombre":"Mootools","descripcion":"Framework}';
//Faltan " para Framework
var objeto = JSON.decode(str, true);
if(objeto != null)
alert(objeto.nombre);
else
alert('El string JSON es incorrecto.');
//El string JSON es incorrecto.
```

CAP.

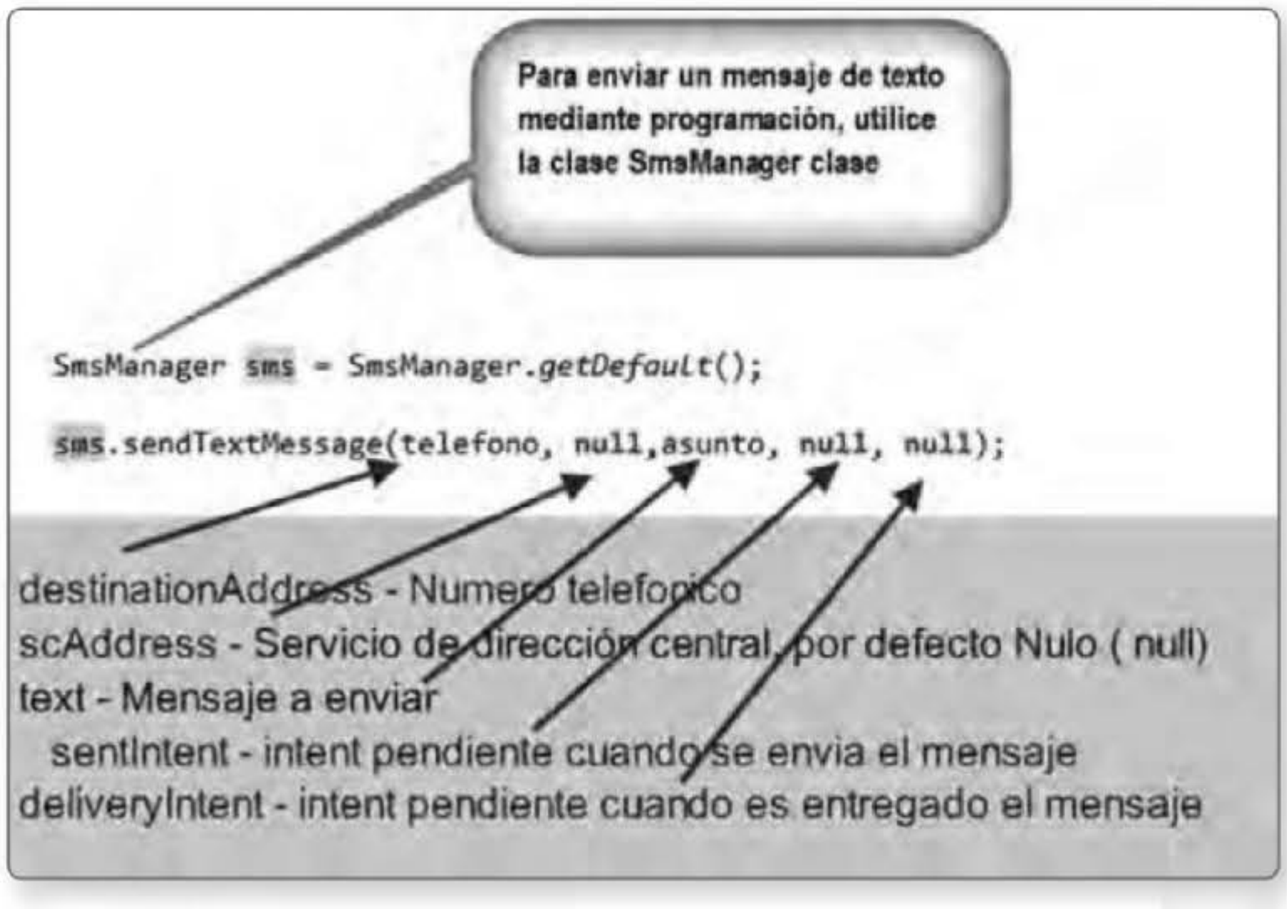
11

SMS

11.1. INTRODUCCIÓN



Puede desarrollar aplicaciones con la capacidad de comunicarse con el mundo exterior. Si desea, su aplicación podría enviar un mensaje de SMS a otro teléfono cuando ocurriese un evento (por ejemplo, cuando una ubicación geográfica determinada se alcanza) o puede que desee acceder a un servicio web que proporcione ciertos servicios (por ejemplo, cambio de moneda, clima, etc.).



11.2. EJEMPLOSMS

Diseñe una aplicación que permita el envío de un mensaje a un número telefónico específico.

Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto*.

Paso 2: Ingrese el nombre del proyecto: EjemploSMS (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).



Paso 3: Conceda permiso de SMS necesario.

EjemploSMS Manifest

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

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="17" />
    <uses-permission android:name="android.permission.SEND_SMS"/>

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.inkadroid.ejemplosms.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

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

    <Button
        android:id="@+id/btn_mensaje"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginLeft="15dp"
        android:layout_marginTop="14dp"
        android:onClick="Mensaje"
        android:text="Envio de Mensaje" />
</RelativeLayout>
```

MainActivity

```
package com.inkadroid.ejemplosms;

import android.os.Bundle;
import android.app.Activity;
import android.telephony.SmsManager;
import android.view.Menu;
import android.view.View;
import android.widget.Toast;
public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {

            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);
    }
     public void Mensaje(View v) {
            SmsManager sms = SmsManager.getDefault();
            sms.sendTextMessage("51999999999", null,"Hola", null, null);


        }

}
```

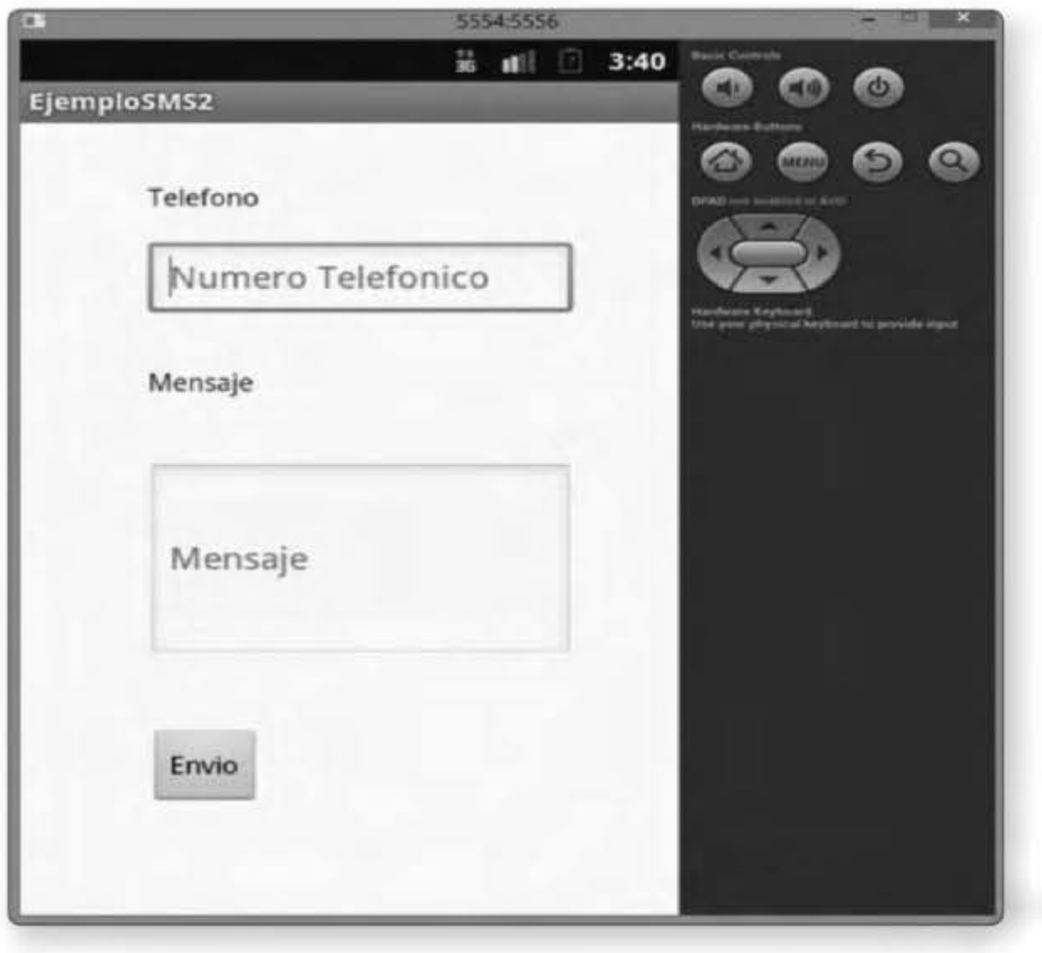
EJECUTANDO LA APLICACIÓN

Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:



11.3. EJEMPLOSMS2

Diseñe una aplicación que permita el envío de mensaje a un número telefónico el cual puede especificarse.



Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto.*

Paso 2: Ingrese el nombre del proyecto: EjemploSMS (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 3: Conceda permiso de SMS necesario.

EjemploSMS2 Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.inkadroid.ejemplosms2"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="17" />
    <uses-permission android:name="android.permission.SEND_SMS"/>

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.inkadroid.ejemplosms2.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Manifest | Application | Permissions | Instrumentation | AndroidManifest.xml

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

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginLeft="46dp"
        android:layout_marginTop="19dp"
        android:text="Telefono" />

    <EditText
        android:id="@+id/txtNumero"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textView1"
        android:layout_below="@+id/textView1"
        android:layout_marginTop="17dp"
        android:ems="10"
        android:hint="Numero Telefonico" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/txtNumero"
        android:layout_below="@+id/txtNumero"
        android:layout_marginTop="28dp"
        android:text="Mensaje" />

        <requestFocus />

    <EditText
        android:id="@+id/txtMensaje"
        android:layout_width="wrap_content"
        android:layout_height="120dp"
        android:layout_alignLeft="@+id/textView2"
        android:layout_below="@+id/textView2"
        android:layout_marginTop="40dp"
        android:ems="10"
        android:hint="Mensaje"
        android:inputType="textMultiLine" />

    <Button
        android:id="@+id/cmdEnvio"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/txtMensaje"
        android:layout_below="@+id/txtMensaje"
        android:layout_marginTop="42dp"
        android:onClick="Mensaje"
        android:text="Envio" />

</RelativeLayout>
```

MainActivity.java

```
package com.inkadroid.ejemplosms2;

import android.os.Bundle;
import android.app.Activity;
import android.telephony.SmsManager;
import android.view.Menu;
import android.view.View;
import android.widget.Toast;
import android.widget.EditText;


public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);
    }


     public void Mensaje(View v) {

                String telefono = ((EditText) findViewById(R.id.txtNumero)).getText().
toString();
            String  asunto    =  ((EditText)  findViewById(R.id.txtMensaje)).getText().
toString();

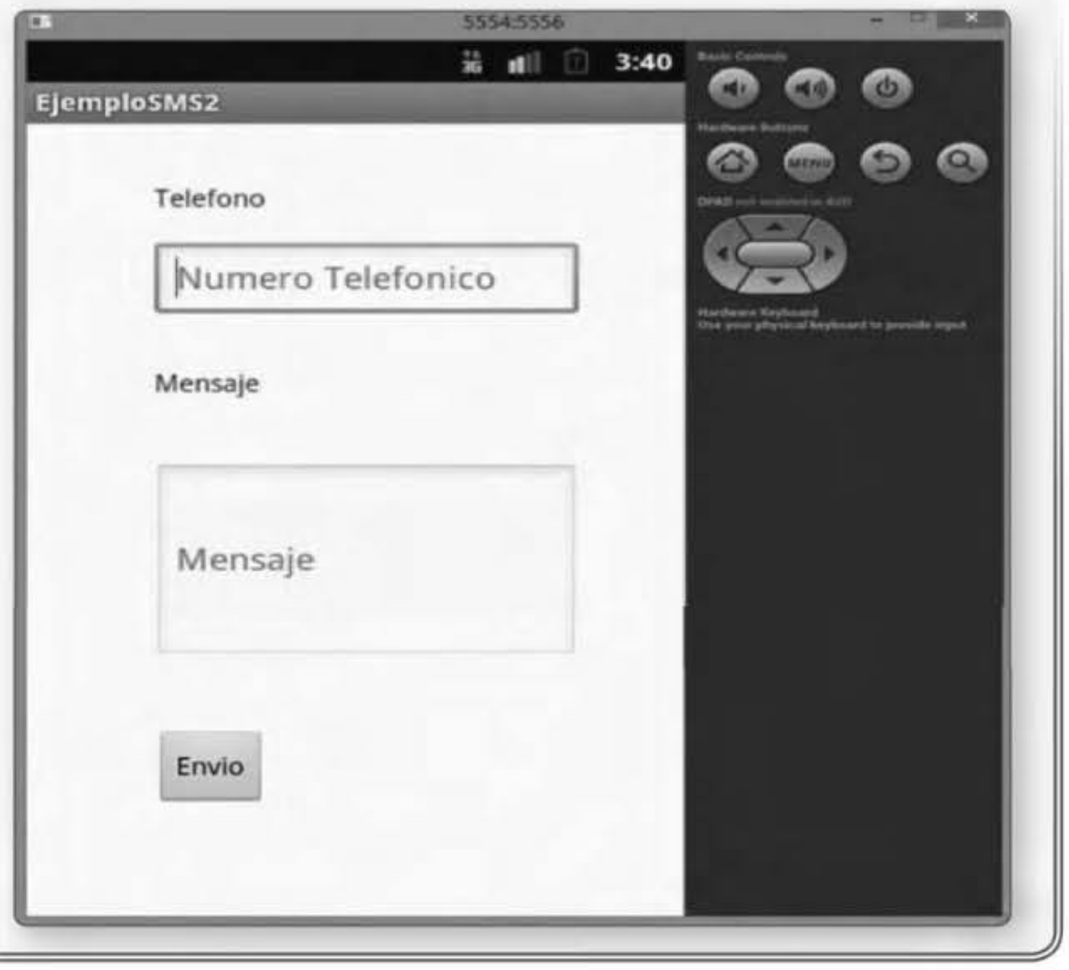
                        SmsManager sms = SmsManager.getDefault();
                sms.sendTextMessage(telefono, null,asunto, null, null);

            }

}
```

EJECUTANDO LA APLICACIÓN

Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:



11.4. EJEMPLO SMS CON MENSAJE DE NOTIFICACIÓN

Diseñe una aplicación que permita el envío de mensaje a un número telefónico el cual puede especificarse y además saber si el mensaje fue enviado correctamente.

Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto.*

Paso 2: Ingrese el nombre del proyecto: EjemploSMS (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 3: Conceda permiso de SMS necesario.

EjemploSMS3 Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.inkadroid.ejemplosms3"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="17" />
        <uses-permission android:name="android.permission.SEND_SMS"/>

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.inkadroid.ejemplosms3.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Manifest | Application | Permissions | Instrumentation | AndroidManifest.xml

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

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginLeft="46dp"
        android:layout_marginTop="19dp"
        android:text="Telefono" />
```

```
    <EditText
        android:id="@+id/txtNumero"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textView1"
        android:layout_below="@+id/textView1"
        android:layout_marginTop="17dp"
        android:ems="10"
        android:hint="Numero Telefonico" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/txtNumero"
        android:layout_below="@+id/txtNumero"
        android:layout_marginTop="28dp"
        android:text="Mensaje" />


        <requestFocus />

    <EditText
        android:id="@+id/txtMensaje"
        android:layout_width="wrap_content"
        android:layout_height="120dp"
        android:layout_alignLeft="@+id/textView2"
        android:layout_below="@+id/textView2"
        android:layout_marginTop="40dp"
        android:ems="10"
        android:hint="Mensaje"
        android:inputType="textMultiLine" />

    <Button
        android:id="@+id/cmdEnvio"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/txtMensaje"
        android:layout_below="@+id/txtMensaje"
        android:layout_marginTop="42dp"
        android:onClick="Mensaje"
        android:text="Envio" />

</RelativeLayout>
```

MainActivity.java

```
package com.inkadroid.ejemplosms3;

import android.os.Bundle;
import android.app.Activity;
import android.app.PendingIntent;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.telephony.SmsManager;
import android.view.Menu;
import android.view.View;
import android.widget.Toast;
import android.widget.EditText;



public class MainActivity extends Activity {

  String ENVIADO = “mensaje enviado”;
  String ENTREGADO = “mensaje entregado”;
  PendingIntent enviadoPI, entregadoPI;
  BroadcastReceiver smsSentReceiver, smsDeliveredReceiver;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);

            //Estos dos objetos PendingIntent se utiliza para enviar transmisiones
            //cuando un mensaje SMS es enviado o entregado
     enviadoPI = PendingIntent.getBroadcast(this, 0,new Intent(ENVIADO), 0);
     entregadoPI = PendingIntent.getBroadcast(this, 0,new Intent(ENTREGADO), 0);
}


 @Override
  public void onResume() {
    super.onResume();
    //Creamos un Broadcast Receiver es una especie de receptor de los eventos
    //que produce el sistema operativo Android como SMS ENVIADO
    smsSentReceiver = new BroadcastReceiver(){
                          @Override
                          public void onReceive(Context arg0, Intent arg1) {

                                  switch (getResultCode())
                                  {
        case Activity.RESULT_OK:
          Toast.makeText(getBaseContext(), “SMS ENVIADO”,
              Toast.LENGTH_SHORT).show();
          break;
        case SmsManager.RESULT_ERROR_GENERIC_FAILURE:
          Toast.makeText(getBaseContext(), “FALLA GENERICA”,
              Toast.LENGTH_SHORT).show();
          break;
        case SmsManager.RESULT_ERROR_NO_SERVICE:
          Toast.makeText(getBaseContext(), “NO HAY SERVICIO”,
              Toast.LENGTH_SHORT).show();
          break;
```

```
        case SmsManager.RESULT_ERROR_NULL_PDU:
          Toast.makeText(getBaseContext(), "PAQUETE VACIO",
              Toast.LENGTH_SHORT).show();
          break;
        case SmsManager.RESULT_ERROR_RADIO_OFF:
          Toast.makeText(getBaseContext(), "RADIO APAGADO",
              Toast.LENGTH_SHORT).show();
          break;
        }
                    }
           };

          //reescribimos el método onReceive que se activará cuando se produzca
          //algun evento relacionado al envio de mensaje
        smsDeliveredReceiver = new BroadcastReceiver(){
          public void onReceive(Context arg0, Intent arg1) {
            switch (getResultCode())
            {
            case Activity.RESULT_OK:
              Toast.makeText(getBaseContext(), "SMS ENTREGADO",
                  Toast.LENGTH_SHORT).show();
              break;
            case Activity.RESULT_CANCELED:
              Toast.makeText(getBaseContext(), "SMS NO ENTREGADO",
                  Toast.LENGTH_SHORT).show();
              break;
            }
          }
          };
          //Registar los dos BroadcastReceivers necesarios para notifacacion
          registerReceiver(smsDeliveredReceiver, new IntentFilter(ENTREGADO));
          registerReceiver(smsSentReceiver, new IntentFilter(ENVIADO));
        }


            @Override
            public void onPause() {
              super.onPause();
              //---unregister the two BroadcastReceivers---
              unregisterReceiver(smsSentReceiver);
              unregisterReceiver(smsDeliveredReceiver);
            }


           //metodo de envio de mensaje SMS
            public void Mensaje(View v) {

                    String telefono = ((EditText) findViewById(R.id.txtNumero)).getText().
toString();
                    String asunto = ((EditText) findViewById(R.id.txtMensaje)).getText().
toString();
                          SmsManager sms = SmsManager.getDefault();

                          sms.sendTextMessage(telefono,       null,asunto,       enviadoPI,
entregadoPI);
              }
}
```

EJECUTANDO LA APLICACIÓN

Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:



CAP.

12

Marketing

12.1. ASPECTOS GENERALES



Android Market

Android Market es una tienda de software en línea desarrollada por Google para los dispositivos Android. Es una aplicación ("app") llamada "Mercado" que está preinstalada en la mayoría de los dispositivos Android y que permite a los usuarios buscar y descargar aplicaciones publicadas por desarrolladores terceros, alojada en Android Market. Los usuarios también pueden buscar y obtener información sobre aplicaciones en esta página web. En enero de 2012, Android Market dispone de más de 500.000 aplicaciones.

Fuente: sistemaoperativoandroid.wordpress.com/

El Android Market permite acceder a mercados que antes no eran posible con los métodos tradicionales; actualmente el Market de Android supera cualquier pronóstico de negociación y se ha convertido en una fuente de oportunidad de negocios para empresas de diferentes giros.

El Android Market se ha disparado desde la introducción de menos de 500 aplicaciones para más de 70 000 aplicaciones. Al igual que los proveedores de los supermercados, todos los desarrolladores de aplicaciones está compitiendo por espacio virtual, con la expectativa de atraer la atención de los compradores.

Aproximadamente, se publica diariamente entre 100 y 300 aplicaciones en el Android Market relacionados a temas de:

Juegos	35%
Libros	15%
Entretenimiento	19%
Viajes	13%
Educación	8%
Otros	10%

Respecto a las ventas que se realizan en el Android Market, Google no publica sus ventas, pero lo utiliza como instrumento para evaluar la distribución de su producto Android en los dispositivos móviles.

Los juegos para móviles desbancan el mercado de las videoconsolas portátiles

Los juegos para teléfonos móviles representan el 58% del mercado de los dispositivos portátiles en Estados Unidos, según un estudio realizado por Flurry y NPD. En el 2009, las ventas de juegos para móviles suponía el 19% del mercado en comparación con la cuota del 70% de Nintendo DS y del 11% de PSP. Dos años después, se han alterado estos porcentajes de forma vertiginosa dejando un 35% a la portátil de Nintendo y en un 6% a la de Sony.
La falsificación o apropiación de derechos también ha influido en la disminución de las ventas; ya que se estima que solo el 34% de los poseedores de una consola portátil compran sus juegos legalmente. El nuevo Nintendo 3DS ha tenido que bajar drásticamente su precio para poder funcionar en el mercado japonés.

12.2. Orientar la aplicación para un cliente

Elementos claves que permiten orientar mi aplicación al cliente:

- Su aplicación debe poseer alguna funcionalidad que sirva a los clientes. Dicha funcionalidad deberá estar definida por el cliente y no por el fabricante; para ello, tenga presente las tendencias del mercado.
- Conocer al detalle la funcionalidad de su aplicación. Es importante definir el perfil funcional de su aplicación; por ejemplo, debe saber si su aplicación está orientada al ocio, deporte, comida, etc.
- Identificar a sus competidores; ya que en un mercado globalizado el cliente tiene la opción de elegir entre una variedad de aplicaciones; para ello, realice un estudio para conocer a sus competidores. Existen tres tipos de competidores:
  - Comparten clientes.
  - Constituyen una referencia, pero difícilmente llegarán a ser competidores.
  - Pueden ser una amenaza.
- Posicionamiento y constante actualización de su aplicación; ya que compite con otras en un mercado abierto, este mercado valora una serie de funcionalidades críticas así como los atributos que cumplan estas funcionalidades y otros que no; para ello, es necesario que se realice actualizaciones no solo para desechar las funcionalidades que no agregan valor sino para ofrecer nuevas funcionalidades que enriquezcan las mismas.

12.3. Clasificación de las aplicaciones móviles referente a su funcionalidad

Comunicaciones	Clientes de redes sociales (Facebook, Twitter)
	Mensajería instantánea (What´s App)
	Clientes de e-mail
	Navegadores web
	Servicios Noticias
	Voz IP

Juegos	Cartas o de casino
	Puzzle o estrategia
	Acción o aventura
	Deportes
	Deportes de ocio
Multimedia	Visores de gráfico o imágenes
	Visores de presentaciones
	Reproductores de video (Youtube)
	Reproductores de audio
	Reproductores de streaming (audio/video)
Productividad	Calendarios
	Calculadoras
	Diarios
	Notas, recordatorios o procesadores de texto
	Hojas de cálculo
	Servicios de directoio
	Bancos o finanzas
Viajes	Guías de ciudades
	Convertidores de moneda
	Traductores
	Mapas GPS
	Itinerarios programados
	Previsión meteorológico
Compras	Lectores de código de barras y bases de datos de productos
	Clientes de tiendas web
	Subastas
	Cupones de descuento
	Lista de compra
Utilidades	Gestores de perfiles de usuario
	Salvapantallas
	Libretas de direcciones
	Gestor de procesos
	Gestor de llamadas
	Gestor de ficheros

Entretenimiento	Lectores de libros
	Horóscopos
	Guías de programación de televisión, radio, etc.
	Recetas
	Cómics
Bienestar	Seguimiento de dietas
	Primeros auxilios
	Consejos al embarazo
	Entrenamiento personal
	Guías de salud

Como se puede notar en la tabla anterior, existen aplicaciones para casi todas las necesidades.

12.4. TECNOLOGÍAS MÓVILES EN LA CADENA DE VALOR

Infraestructura de la Empresa
Administración general, planeación, asuntos legales, finanzas contabilidad, calidad (Aplicaciones móviles que me permitan visualizar los estado financiero, el balance, cuentas por pagar y cobrar, etc).

Dirección de Recursos Humanos
Selección, contratación, evaluación, capacitación, compensación (Aplicaciones móviles que me permitan visualizar sobre historial de planillas, capacitaciones, concursos, etc).

Desarrollo Tecnológico
Incluye la tecnología que soporta a todas las actividades de la cadena de valor.

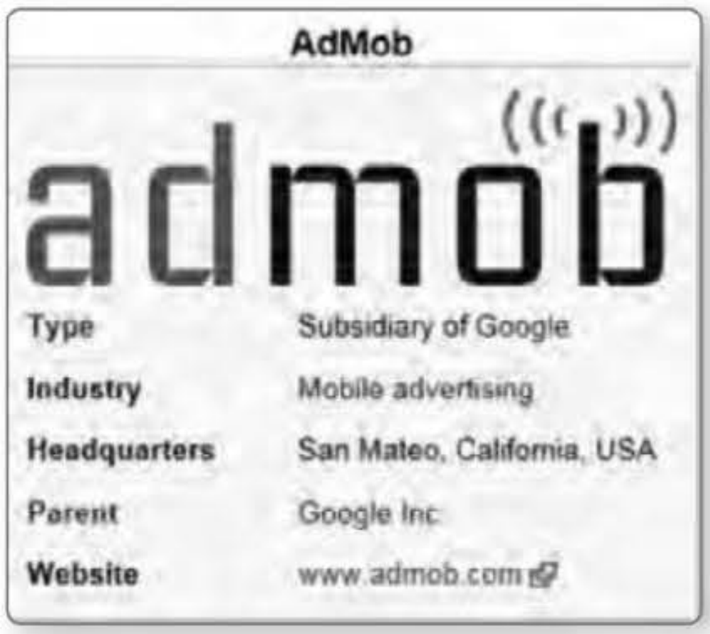
Adquisiciones
Cotización, evaluación de proveedor, envío de orden o compromiso (Aplicaciones móviles que permitan visualizar cotizaciones, establecer cuadros comparativos)

Margen

Logística Interna	Operaciones	Logística Externa	Marketing y Ventas	Servicios
Recepción **Almacenaje** **Manipulación de materiales** **Inventarios** **Devoluciones** (Aplicaciones móviles para la consulta de las actividades descritas)	(Aplicaciones móviles que me permiten visualizar: estados de equipos, instrumentos, indicadores, consultas de calendarios)	**Almacenaje** **Procesamiento pedidos** **Distribución** **Devolución** **Entrega al cliente final** (Aplicaciones móviles para la consulta de las actividades descritas)	**Publicidad** **Promoción** **Fuerza de ventas** **Cotizaciones** **Selección de canales** (Aplicaciones móviles para la consulta de actividades descritas)	**Instalación** **Reparación** **Entrenamiento** **Suministro de repuestos** (Aplicaciones móviles: Monitoreo de equipos y productos, Recepción de retroalimentación, fallas, sugerencias, quejas)

Margen

12.5. RED PUBLICITARIA MÓVIL ADMOB



AdMob es la plataforma más grande de la publicidad móvil, que trabaja con el sistema operativo Android. La empresa fue fundada en el año 2006 por Omar Hamoui y cuatro años después fue comprada por Google a unos 750 millones de dólares.

Los anuncios AdMob pueden ser comparados con Google AdSense, excepto que está diseñado para móviles. AdMob está dirigido principalmente a los desarrolladores de aplicaciones; aunque Google está introduciendo la idea de combinar Adsense con AdMob; es decir, está intentando integrar AdMob en la plataforma Double Click.

12.5.1. Mediación de redes publicitarias de AdMob

Este producto llamado "mediación de AdMob" se coordina con las redes publicitarias para ayudar a maximizar el porcentaje de anuncios publicados e incrementar los ingresos.

La mediación de AdMob envía solicitudes de anuncios a varias redes publicitarias, incluida AdMob, para asegurarse que encuentre la mejor red disponible para publicar su anuncio. Si su primera elección no satisface la solicitud de anuncio, el producto estudia una lista de alternativas y se comunica con ellas en el orden especificado. De manera que continúa buscando hasta que encuentre la red que satisfaga la solicitud o hasta que haya probado con todas las redes elegidas.

La configuración le indica a la mediación de AdMob cómo debe elegir entre las redes. Puede asignar tráfico en función del porcentaje, la ubicación y el eCPM (coste efectivo por mil impresiones, una estimación de los ingresos publicitarios).

Por ejemplo, si desea usar la mediación de AdMob para enviar la solicitud primero a una red determinada cuando proceda de un país específico o cuando desee dividir el tráfico de forma equitativa entre distintas redes o asignar tráfico a la red con el eCPM más elevado. Para cada solicitud de anuncio, la mediación de AdMob adapta la búsqueda de red a sus prioridades.

Google superará este año a Facebook en ingresos por publicidad online

Google apunta a ser la mayor fuente de ingresos en los Estados Unidos por publicidad este año, tomando como referencia a la (sic) actual número uno que es Facebook. Además, se espera que consolide su presencia en la publicidad online.

La creciente influencia de Google en los anuncios incrustados -las cajas con imágenes y video encontrados en sitios web- ha llegado al tiempo que crece la publicidad de vídeo en su filial Youtube (sic). Asimismo, aumenta su presencia a través de la publicidad móvil AdMob, una empresa adquirida por Google en 2009.

Este año, Google podrá reclamar un 15,4 por ciento de los beneficios de visualización de su servicio de anuncios, o lo que es lo mismo, 2.310 millones de dólares (1.770 millones de euros). Por su parte, Facebook reclamará un 14,4 por ciento de los beneficios de su servicio similar, que equivale

a 2.160 millones de dólares (1.655 millones de euros), según proyecciones de eMarketer, una firma de investigación de publicidad digital.

Además, la firma agregó que los editores en general han sido heridos por una demanda más débil de la esperada este año, ya que las grandes marcas han evitado el derroche en costosas campañas digitales.

Las ganancias en publicidad de Google y de Facebook fueron seguidas por Yahoo!, Microsoft y AOL, que tienen redes de publicidad a través de sus propiedades de contenido digital.

Facebook afirmó esta semana que comenzaría a desarrollar una red de publicidad móvil. Esta permitirá a los anunciantes orientar los anuncios a partir de datos de Facebook a través de toda la Red y no sólo (sic) de sitios web pertenecientes a Facebook.

SAN FRANCISCO, 20 Sep. (Reuters/EP)

Fuente: www.europapress.es

12.5.2. AdMob para desarrolladores

Para maximizar la rentabilidad de las aplicaciones móviles, debe elegir si poner precio, coloque publicidad (aunque muchos optan por arrancar ROMS con el fichero host modificado para que esta no aparezca. Esta alternativa dificulta el trabajo de los desarrolladores) o por donación. Se recomienda utilizar las dos primeras opciones, teniendo dos versiones de la aplicación. Para esto, deberá seguir los siguientes pasos:

Paso 1: Registre en AdMob y rellene los datos bancarios o añada su cuenta PayPal.







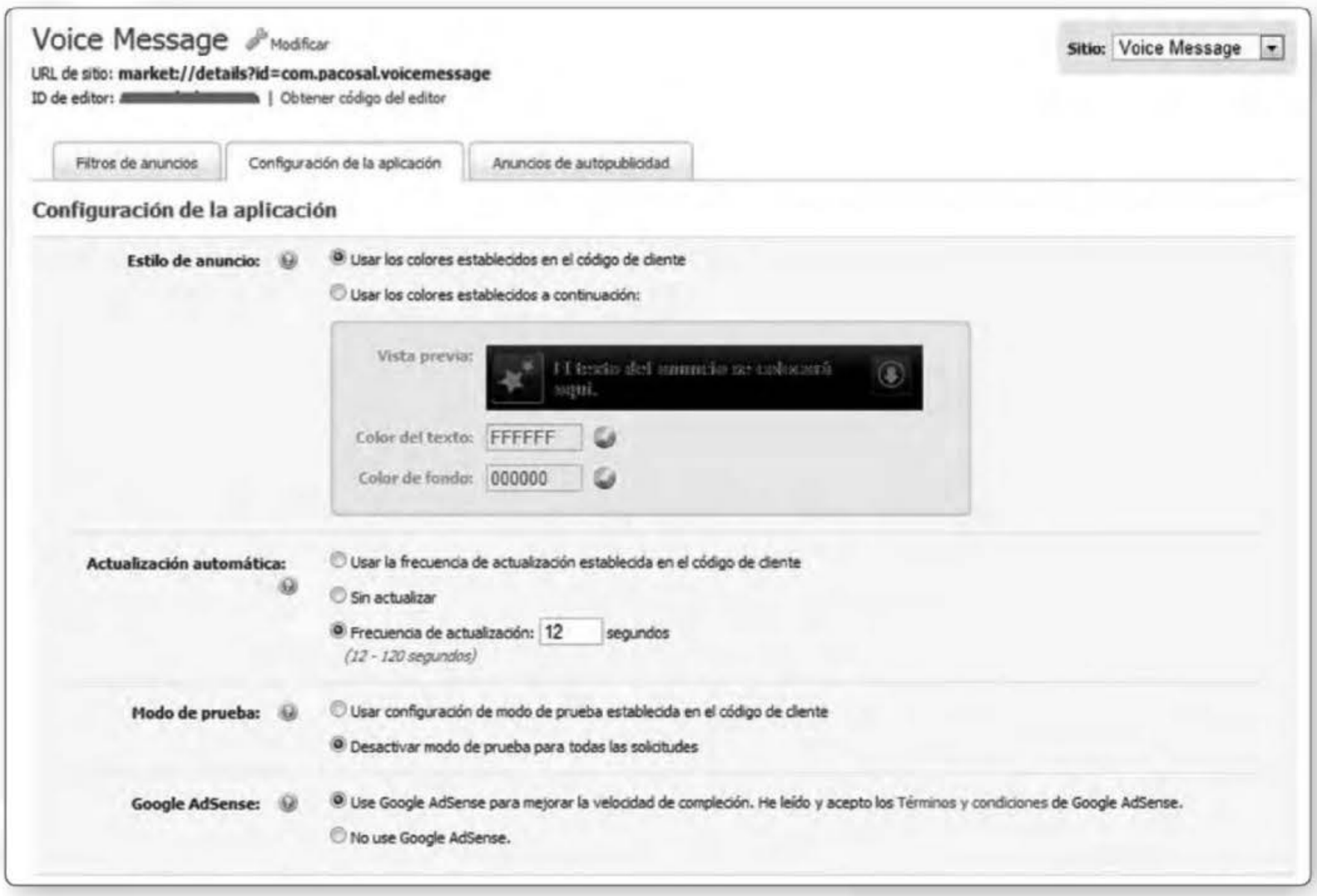
Paso 2: Para recibir la publicidad de Admob, debe dar de alta a la aplicación, con la opción correspondiente (ver fig.).



Paso 3: Pulsando el botón Agregar aplicación, una selección del tipo de dispositivo aparecerá; ya que esta red da soporte a todo tipo de sistemas.



Paso 4: Al seleccionar Android, incluya los datos básicos de su app (ver fig.).



Para el desarrollador, es importante utilizar el código (uno diferente por cada aplicación) que está resaltado en rojo (ver fig. ant.) porque sirve para incluirlo en el desarrollo de su aplicación para que AdMob lo identifique y pueda así registrar anuncios.

12.5.3. Desarrollar para AdMob

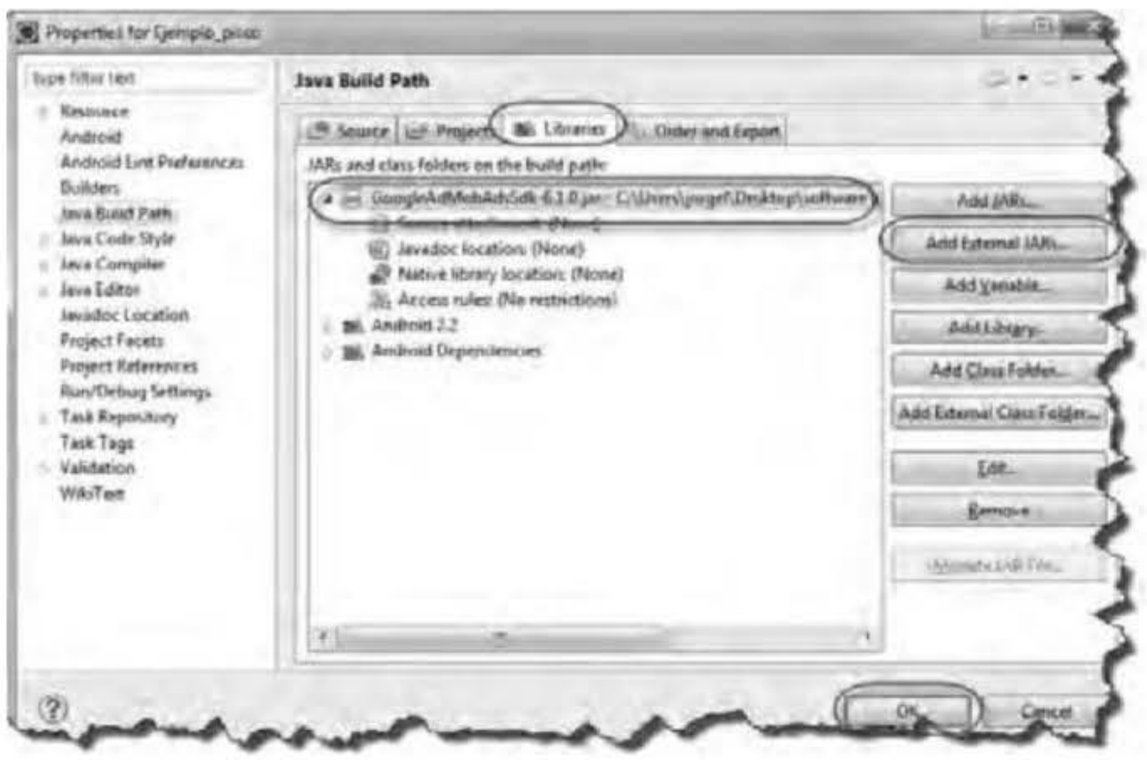
Una vez obtenido el código de la aplicación, debe descargar la librería de AdMob, que viene comprimida en formato .zip. El link de descarga de la librería AdMob es el siguiente:

https://developers.google.com/mobile-ads-sdk/download?hl=es

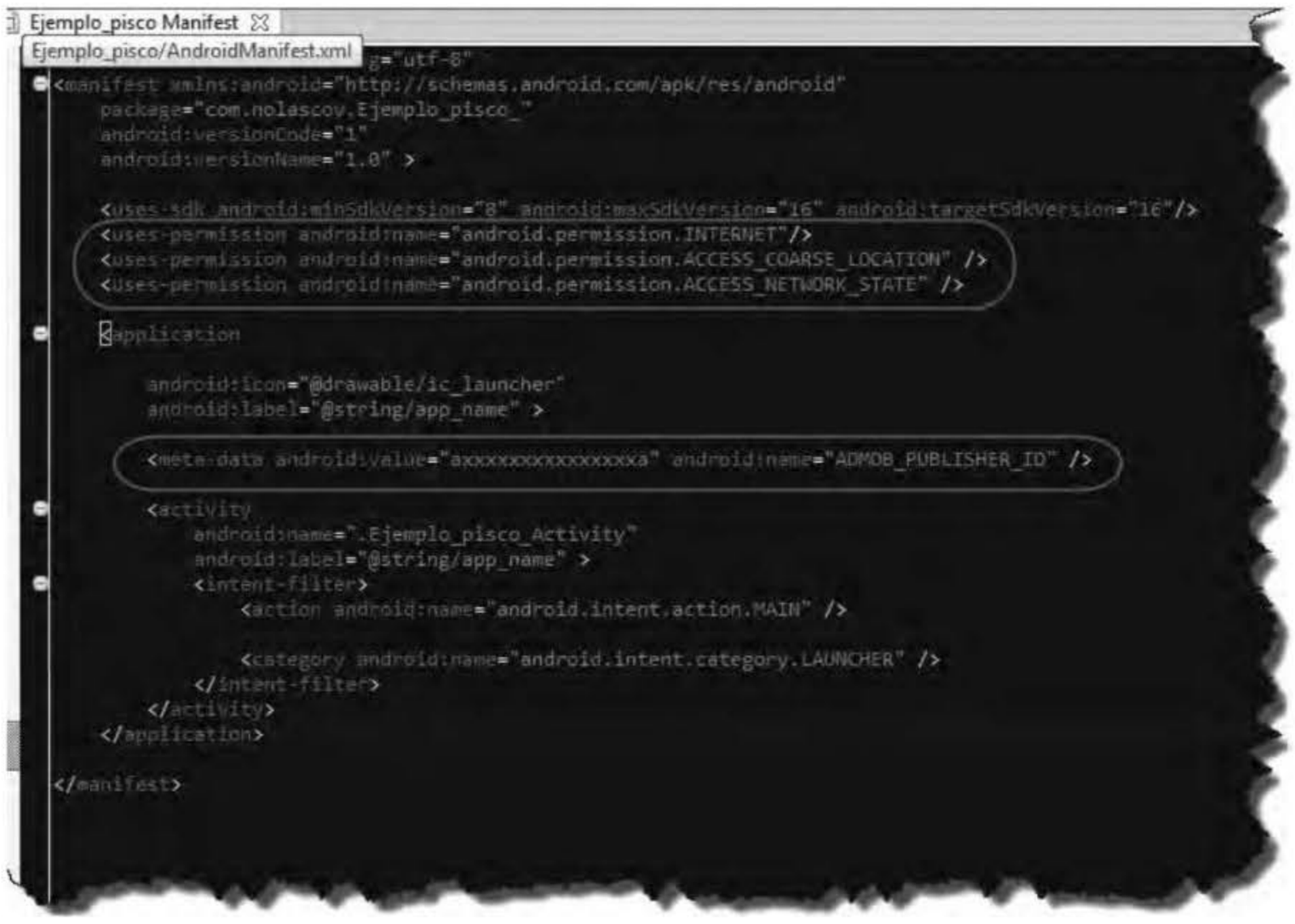
Paso 1: Pulse el botón (ver fig.) que aparecerá luego de seleccionar la opción Obtener código del editor de la pantalla anterior.



Paso 2: Almacene el fichero .jar en una carpeta y añádala a su proyecto como una librería externa.



Paso 3: En el fichero AndroidManifest.xml de la app, registre el código de la aplicación añadiendo una línea en el apartado <Application> y dé los permisos necesarios para que pueda conectarse a Internet y si requiere los permisos de localización para que los anuncios tengan mayor sentido y los usuarios los pulsen.



Paso 4: Los anuncios se muestran en las actividades y suelen colocarse arriba o abajo. Para ello, incluya el código correspondiente en el Layout de la Activity.



Paso 5: Quite o agregue los anuncios.

```
Si queremos poner o quitar los anuncios de forma dinámica, solo tenemos que referenciar
el componente y cambiar sus propiedades:
// quit ads
  com.admob.android.ads.AdView  ad  =  (com.admob.android.ads.AdView  )findViewById(R.
id.ad);
 ad.setVisibility(com.admob.android.ads.AdView.GONE);
// set ads
  com.admob.android.ads.AdView  ad  =  (com.admob.android.ads.AdView  )findViewById(R.
id.ad);
 ad.setVisibility(com.admob.android.ads.AdView.VISIBLE);
```

12.5.4. Estrategias para la comercialización de las aplicaciones móviles



La tecnología de dispositivos móviles ha revolucionado la forma en que las personas interactúan con las empresas y marcas. Es por esta razón que el desarrollo de aplicaciones móviles se ha convertido en una necesidad que todas las empresas deben implementar si quieren mantener el "ritmo" que los clientes requieren para pensar y tomar decisiones.

El desarrollo de aplicaciones móviles es uno de los negocios más activos del momento lo que constituye una gran oportunidad de crecimiento tanto para desarrolladores como para empresas. Esto viene sustentado por tres elementos importantes:

- La redefinición del ecosistema móvil.
- La segmentación de desarrolladores, los benéficos y el coste del desarrollo de aplicaciones.
- El marketing y la distribución de aplicaciones.
- La penetración de las aplicaciones móviles por países.

Se estima que el precio medio de una aplicación móvil distribuida a través del Android Market es de cuatro unidades monetarias; debido al alto número de aplicaciones móviles actuales, esto podría ser desalentador. En Post PC, la estrategia a seguir es bien analizada si se desea obtener utilidades a través de la venta de la aplicación móvil o por medio de la publicidad.

La segmentación ha sido relevante en la reconfiguración de Android Market; por ejemplo, de To Ventas, Destacados, Categorías.



Respecto al resultado de las aplicaciones, solo se puede esperar que las aplicaciones:

- Produzcan altos ingresos solo al principio.
- Sean sostenibles durante el tiempo y ganancias sostenibles.
- No obtengan ingresos.

12.6. EJEMPLO_MARKETING



Cree una aplicación lector para el feed del diario *El Comercio* el cual poseerá una publicidad.

Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando Archivo > Nuevo > Android proyecto.

Paso 2: Ingrese el nombre del proyecto: Ejemplo_Marketing (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 3: Para recibir la publicidad de AdMob, dé de alta a la aplicación con la opción correspondiente (ver fig.).



Paso 4: Seleccione Aplicación Android.



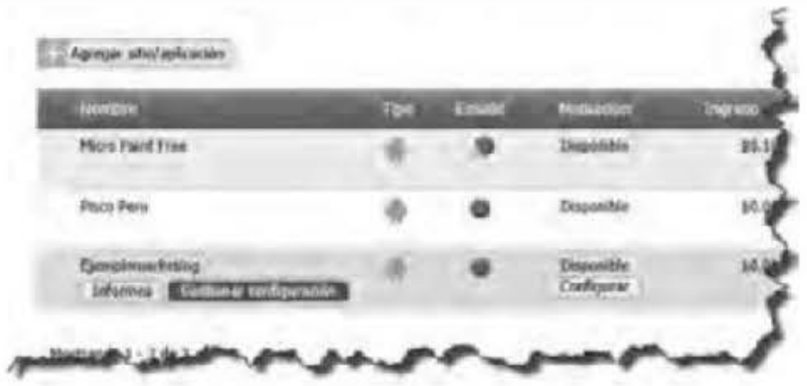
Paso 5: Al seleccionar Android, incluya los datos básicos de su app y luego presione el botón Continuar.



Paso 6: Presione el botón Ir a sitios/aplicaciones.



Paso 7: Presione el botón Gestionar configuración.





Paso 8: Obtendrá el ID.

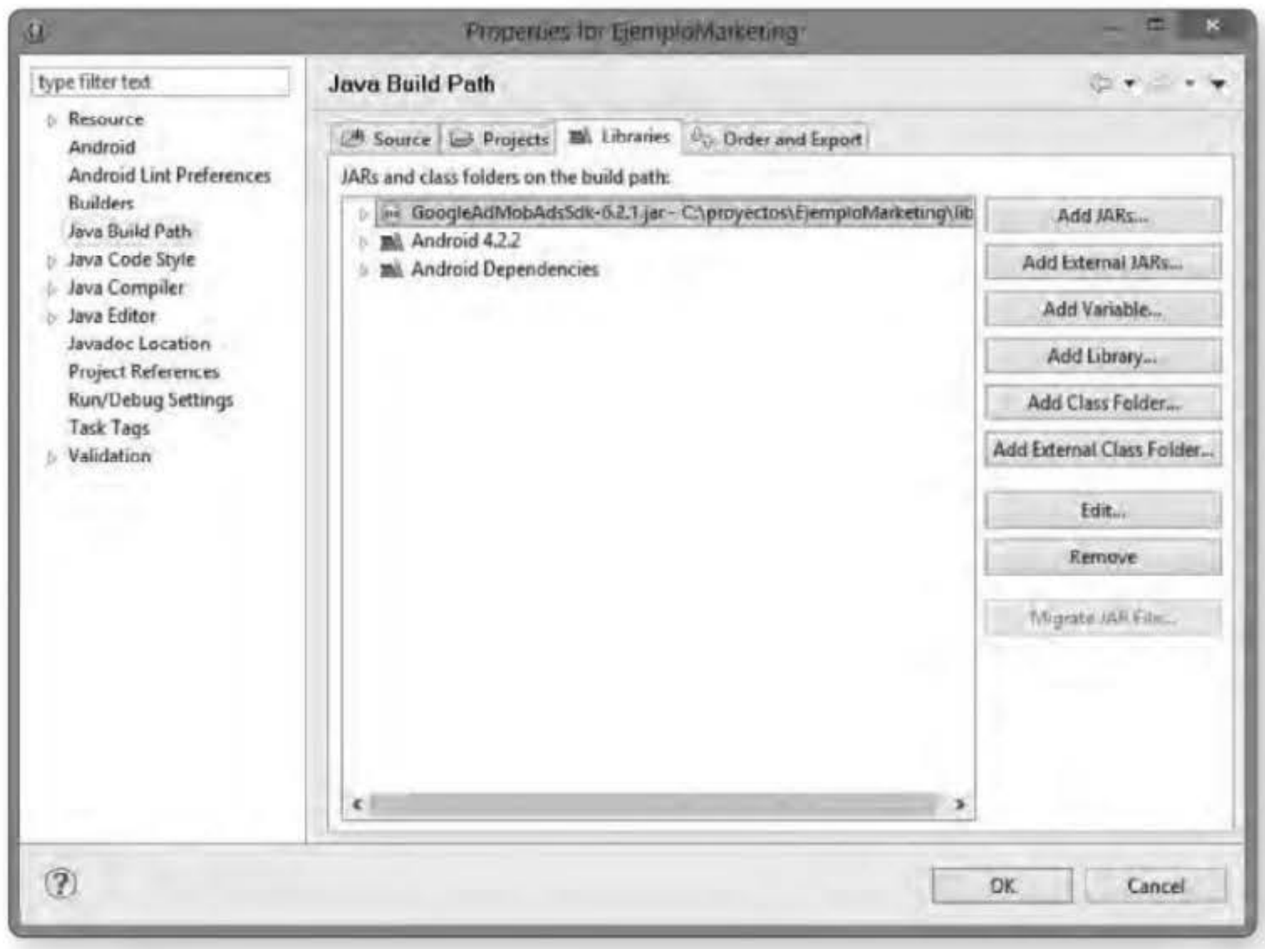
Nota:

Es importante para el desarrollador que utilice el código (uno diferente por aplicación) que está en rojo (ver fig. ant.); ya que es el que se debe incluir en su desarrollo para que AdMob identifique su aplicación y pueda registrar anuncios.

Paso 9: Dé el permiso a la aplicación para que pueda accesar a Internet mediante el archivo de manifiesto:



Paso 10: Almacene el fichero .jar en una carpeta y añádala a su proyecto como una librería externa.



Archivo de manifiesto



MainActivity.java

```
package com.inkadroid.ejemplomarketing;
import java.util.HashMap;
import java.util.LinkedList;
import com.google.ads.AdSize;
import com.google.ads.AdView;
import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.view.View;
```

```
import android.view.Window;
import android.view.WindowManager;
import android.widget.AdapterView;
import android.widget.ListView;
import android.widget.SimpleAdapter;
import android.widget.AdapterView.OnItemClickListener;
import com.google.ads.*;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.LinearLayout;
import android.widget.ListView;
import android.widget.Toast;
import android.widget.AdapterView.OnItemClickListener;

public class MainActivity extends Activity {

    //codigo ID para registrar anuncios
    private static final String MY_AD_UNIT_ID =”a15154cc443b790”;

    //control donde se mostrara el banner
    private AdView adView;

    //utilizaremos 2 constantes para guardar la información en un HashMap

    static final String TITULO = “T”;
    static final String LINK = “L”;

     //utilizamos una estructura de datos estática para guardar la información del feed
    static LinkedList<HashMap<String, String>> datos;

    //direccion del feed
    static String URL = “http://elcomercio.feedsportal.com/rss/portada.xml”;

    //es necesario un dialogo mientras se carga los datos
    private ProgressDialog barra;

     //un manejador(Handler) para enviar un mensaje de un hilo a otro cuando la carga
de datos
     //haya terminado.

    private final Handler progressHandler = new Handler() {
            @SuppressWarnings(“unchecked”)
            public void handleMessage(Message msg) {
                    if (msg.obj != null) {
                            datos = (LinkedList<HashMap<String, String>>)msg.obj;
                            //llamar a la funcion que rellena el listview
                            setData(datos);
                    }
                    barra.dismiss();
      }
    };
  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    //desactivar titulo
    requestWindowFeature(Window.FEATURE_NO_TITLE);
```

```
    //mostrar en toda la pantalla
      getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,  WindowManager.
LayoutParams.FLAG_FULLSCREEN);

    setContentView(R.layout.activity_main)

    //Crear la adView
    adView = new AdView(this, AdSize.BANNER, MY_AD_UNIT_ID);
    final LinearLayout layout = (LinearLayout) findViewById(R.id.main);


    // Añadirle la adView
    layout.addView(adView);

                    ListView lista = (ListView) findViewById(R.id.lst);

                    barra = ProgressDialog.show(MainActivity.this,"","Cargandose los
datos...", true);

            new Thread(new Runnable(){
                    @Override
                    public void run() {
                            Parser parser = new Parser(URL);
            Message msg = progressHandler.obtainMessage();
            msg.obj = parser.parse();
                            progressHandler.sendMessage(msg);
                    }}).start();


      //al presionar un click en algun elemento de la lista
    lista.setOnItemClickListener(new OnItemClickListener() {

            @Override
            public void onItemClick(AdapterView<?> av, View v, int position,
                            long id) {
                    //obtenemos el elemento que se presiono click
                    HashMap<String, String> entry = datos.get(position);

                    //abrimos una pagina a travez de un inten
                    Intent   browserAction   =   new   Intent(Intent.ACTION_VIEW,Uri.
parse(entry.get(LINK)));
                    startActivity(browserAction);
            }
    });
  }

    //rellenar el listview
    private void setData(LinkedList<HashMap<String, String>> datos){
      SimpleAdapter sAdapter = new SimpleAdapter(getApplicationContext(), datos,
                      android.R.layout.two_line_list_item,
                      new String[] { TITULO, LINK },
                      new int[] { android.R.id.text1, android.R.id.text2 });
      ListView lista = (ListView) findViewById(R.id.lst);
      lista.setAdapter(sAdapter);
    }

}
```

Parser.java

```
package com.inkadroid.ejemplomarketing;

import java.net.MalformedURLException;
import java.net.URL;
import java.util.HashMap;
import java.util.LinkedList;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

public class Parser {
    private URL url;

    public Parser(String url) {
            try {
                    this.url = new URL(url);
            } catch (MalformedURLException e) {
                    e.printStackTrace();
            }
    }

    public LinkedList<HashMap<String, String>> parse() {
            DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
            LinkedList<HashMap<String, String>> entries = new LinkedList<HashMap<String,
String>>();
            HashMap<String, String> entry;
            try {
                    DocumentBuilder builder = factory.newDocumentBuilder();
                    Document    dom    =    builder.parse(this.url.openConnection().
getInputStream());
                    Element root = dom.getDocumentElement();
                    NodeList items = root.getElementsByTagName("item");
                    for (int i=0;i<items.getLength();i++){
                            entry = new HashMap<String, String>();

                            Node item = items.item(i);
                            NodeList properties = item.getChildNodes();
                            for (int j=0;j<properties.getLength();j++){
                                    Node property = properties.item(j);
                                    String name = property.getNodeName();
                                    if (name.equalsIgnoreCase("title")){
                                            entry.put(MainActivity.TITULO, property.
getFirstChild().getNodeValue());
                                    } else if (name.equalsIgnoreCase("link")){
                                            entry.put(MainActivity.LINK,   property.
getFirstChild().getNodeValue());
                                    }
                            }
                            entries.add(entry);
                    }
            } catch (Exception e) {
                    throw new RuntimeException(e);
            }

            return entries;
    }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
 android:id="@+id/main"
 android:layout_width="fill_parent"
 android:layout_height="fill_parent"
 android:orientation="vertical"
 android:background="#000000"
 xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:ads="http://schemas.android.com/apk/lib/com.google.ads">

    <com.google.ads.AdView android:id="@+id/ad"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            ads:adSize="BANNER"
            ads:adUnitId="a15111d27f1f49e"
            ads:testDevices="TEST_EMULATOR,TEST_DEVICE_ID_GOES_HERE"
            ads:loadAdOnCreate="true"/>

 <TextView
   android:id="@+id/textView1"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:layout_gravity="center_horizontal"
   android:text="El Diario el Comercio"
   android:textSize="30px"/>

    <ListView
      android:id="@+id/lst"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:padding="5dip" >

    </ListView>

</LinearLayout>
```

EJECUTANDO LA APLICACIÓN

Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:



12.7. FACEBOOK PARA ANDROID



El SDK de Facebook para Android es la forma más fácil de integrar su aplicación con la plataforma Android de Facebook. El SDK proporciona soporte para autenticación de sesión con Facebook, leer y escribir a las API de Facebook, así como dar soporte a los elementos de interfaz de usuario (como los recolectores y diálogos).

12.7.1. Requisitos

Eclipse es la opción más usual para el desarrollo de aplicaciones de Android. Específicamente, Eclipse Classic, que se encuentra en la página de descargas, es suficiente para el desarrollo de Android. También necesita Java v1.6 o posterior.

Una vez que tenga Eclipse, descargue el SDK de Android, disponible en el sitio de Android Developers, e instálelo tal como se describe. Además, asegúrese de seguir las instrucciones para instalar el Android Developer Tools (ADT) Plugin. En ocasiones, se solicita reiniciar Eclipse para completar la instalación. Para ello, tendrá que tener el Android 2.2 (API 8) y los componentes instalados con el fin de utilizar el SDK de Facebook.

Note que los dos nuevos íconos en la barra de herramientas de Eclipse y las entradas en la ventana del menú para lanzar el SDK de Android y AVD Manager Gerente (para los emuladores de dispositivos), respectivamente.

12.7.2. Instalación del SDK de Facebook para Android y el APK Facebook

Para empezar a desarrollar con el SDK de Facebook para Android, realice los siguientes pasos:

Paso 1: Descargue y extraiga el archivo ZIP SDK.

Paso 2: La carpeta resultante, facebook-android-sdk-3.0.2.b, contiene el propio SDK. Anote la ubicación de esta carpeta.

El SDK utiliza Facebook aplicación nativa de Facebook para proporcionar soporte para la autenticación cuando está presente. En un dispositivo real, solo tiene que instalar esta aplicación de forma gratuita desde Google Play para comprobarlo. Sin embargo, no es posible acceder a Google Play en un emulador, así que si quiere probar el flujo de ahí, utilice una copia incluida del SDK que se pueden instalar.

Paso 3: La APK instalable de la aplicación de Facebook está en la carpeta bin del SDK y nombre FBAndroid-1.9.10.apk o similar.

Paso 4: Para instalarlo en un emulador de Android, primero tendrá que iniciar el emulador y en Windows, la instalación sería de la siguiente manera:

```
adb install %HOMEPATH%\facebook-android-sdk-3.0.2.b\bin\FBAndroid-1.9.10.apk
```

12.7.3. Importar el SDK dentro de Eclipse

La carpeta de instalación contiene un directorio llamado facebook-android-sdk-3.0.2.b, que es el proyecto en sí SDK. Asimismo, hay una selección de proyectos de ejemplo en la carpeta Ejemplos. Puede importar todos estos proyectos en un área de trabajo limpio de una sola vez. En particular, asegúrese de que no exista un proyecto Facebook en su espacio de trabajo. Si existe, se hace referencia a un posible mayor Facebook SDK; por ello, usted debe borrar o cambiar su espacio de trabajo.

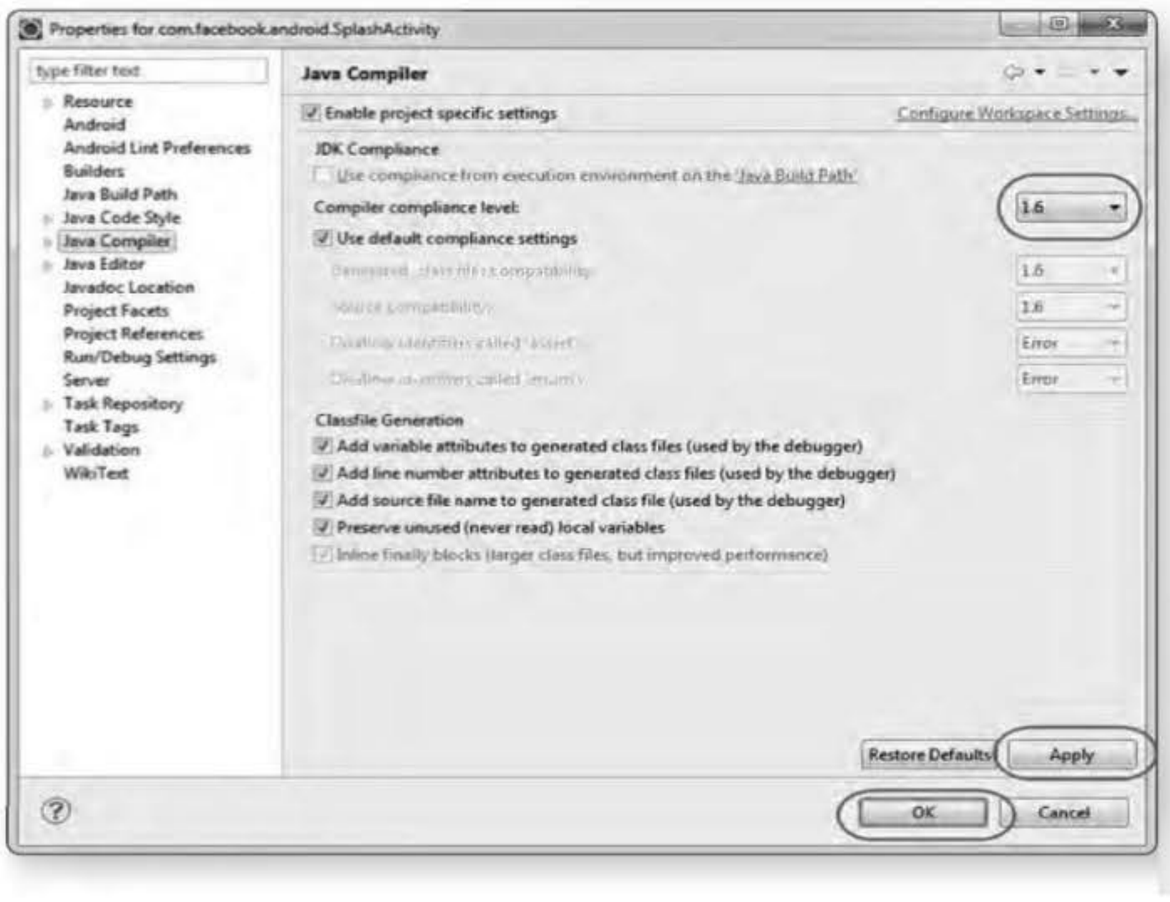


Paso 1: Especifique la ruta de la carpeta facebook-android-sdk-3.0.2.b y luego presione Finalizar.



Paso 2: Si los proyectos de ejemplo están mostrando errores, basta con utilizar la opción Proyecto > Limpiar en Eclipse para actualizar todos los estados del proyecto.

Adicionalmente revise el nivel de compilador Eclipse. Si observa errores como "Android compilador requiere el nivel de cumplimiento 5.0 o 6.0.", solo tiene que asegurarse de que el proyecto (o Eclipse) se encuentre en la versión 1.6.



12.7.4. Ejecutando los ejemplos



A. ProfilePictureSample

Demuestra que no ha iniciado la sesión en el uso del control de imagen de perfil.



B. Hackbook

Aplicación de ejemplo que muestra una selección Gráfica de llamada de Api.

Para ello, seleccione APP Request.





AdBlock: La aplicación que amenaza el negocio móvil de Facebook y Google



Parece que Google, Facebook y otras compañías cuyos ingresos publicitarios dependen de la publicidad online han puesto el grito en el cielo gracias a una sencilla aplicación llamada AdBlock. Hasta la fecha, más de 100.000 usuarios descargan diariamente este software que permite bloquear cookies, pop-ups y demás anuncios que estorban la navegación a través de la red.

Un hecho que hasta la fecha no parecía amenazar a gigantes como Google o Facebook. Sin embargo, Eyeo, la compañía detrás de la aplicación, lanzará mañana una aplicación para móviles Android que promete a sus clientes aniquilar cualquier anuncios (sic) tanto en el navegador como en otras aplicaciones.

Según adelanta el Financial Times, la versión móvil AdBlock Plus bloqueará anuncios tanto en el navegador del móvil como en otras aplicaciones como Facebook o la conocida Angry Birds.

Es decir, redes sociales como Facebook o Twitter podrían ver como sus esfuerzos por incrementar sus ingresos a través de la publicidad procedente de los smartphones podría irse al traste si nadie frena el avance de esta aplicación.

El co-fundador de Eyeo, Till Faida, explicó al Financial Times que, AdBlock Plus "será la primera aplicación para eliminar todos los anuncios en los teléfonos móviles, donde existe una gran necesidad de que esto ocurra".

Según el periódico británico, la versión de PC de AdBlock Plus, que ha sido descargada ya por más de 50 millones de personas, bloquea las ventanas emergentes y anuncios de sitios web, así como los anuncios que preceden a los vídeos que nutren sitios web como YouTube.

Hasta la fecha, AdBlock Plus es descargado cerca de 100.000 veces al día a nivel mundial y cuenta con una "lista blanca" de sitios web con anuncios no intrusivos, como Amazon, cuyos anuncios no están bloqueados. Según Faida, no es posible "comprar" el acceso esta lista.

"El objetivo principal es asegurarse de que los pequeños editores y autores de blogs no se vean perjudicados por el bloqueo de anuncios, para que puedan seguir prestando sus contenidos de forma gratuita", explicó al FT.

Fuente: www.eleconomista.es

CAP.

13

Seguridad de las aplicaciones móviles

13.1. INTRODUCCIÓN

Se ha iniciado la era postPC donde los dispositivos móviles constituyen la nueva web, el futuro. La importancia de las aplicaciones móviles radica en lo siguiente:

- Se pronostica que la venta de los smartphones en el 2015 alcanzará los 982 millones de dispositivos.
- La entidad financiera estadounidense Morgan Stanley estima que la venta de smartphones superará a los PC para el 2012.
- La media de aplicaciones instaladas es de 65, pero el consumidor medio solo utiliza unas 15 Apps habitualmente por semana.
- Existen aplicaciones acorde a las necesidades exigidas por los usuarios.

13.2. SEGURIDAD



Android ha tenido un impacto significativo en el mercado de teléfonos inteligentes y, por ende, en el área de seguridad. Actualmente, es una de las plataformas más atacadas.

Para el 2011, se calculó la presencia de 40 variantes de malware, donde el 70% de ellas aparecieron a finales de ese año. El malware ataca especialmente esta plataforma; por lo cual se está desarrollando una serie de nuevas características de seguridad.

13.3. CARACTERÍSTICAS DEL DISPOSITIVO MÓVIL

Android fue diseñado para ser compatible con una amplia gama de hardware, en gran parte al núcleo de Linux, que con los años ha evolucionado hasta ser compatible con una gran variedad de hardware. Esta es una característica importante de la plataforma, ya que permite libertad a los fabricantes a diseñar, adquirir o de otro modo integrar los componentes ideales con el sistema operativo Android. Esta estrategia ha llevado al desarrollo de potentes dispositivos Android Quad-Core (cuatro núcleos) capaz de procesar significativamente así como el nivel de entrada de los dispositivos dirigidos a planes de servicio inalámbrico de nivel de entrada. Aunque la compatibilidad de hardware es muy bueno para los fabricantes, los proveedores de servicios inalámbricos, los consumidores en última instancia y la diversidad plantea desafíos para los analistas forenses y de seguridad.



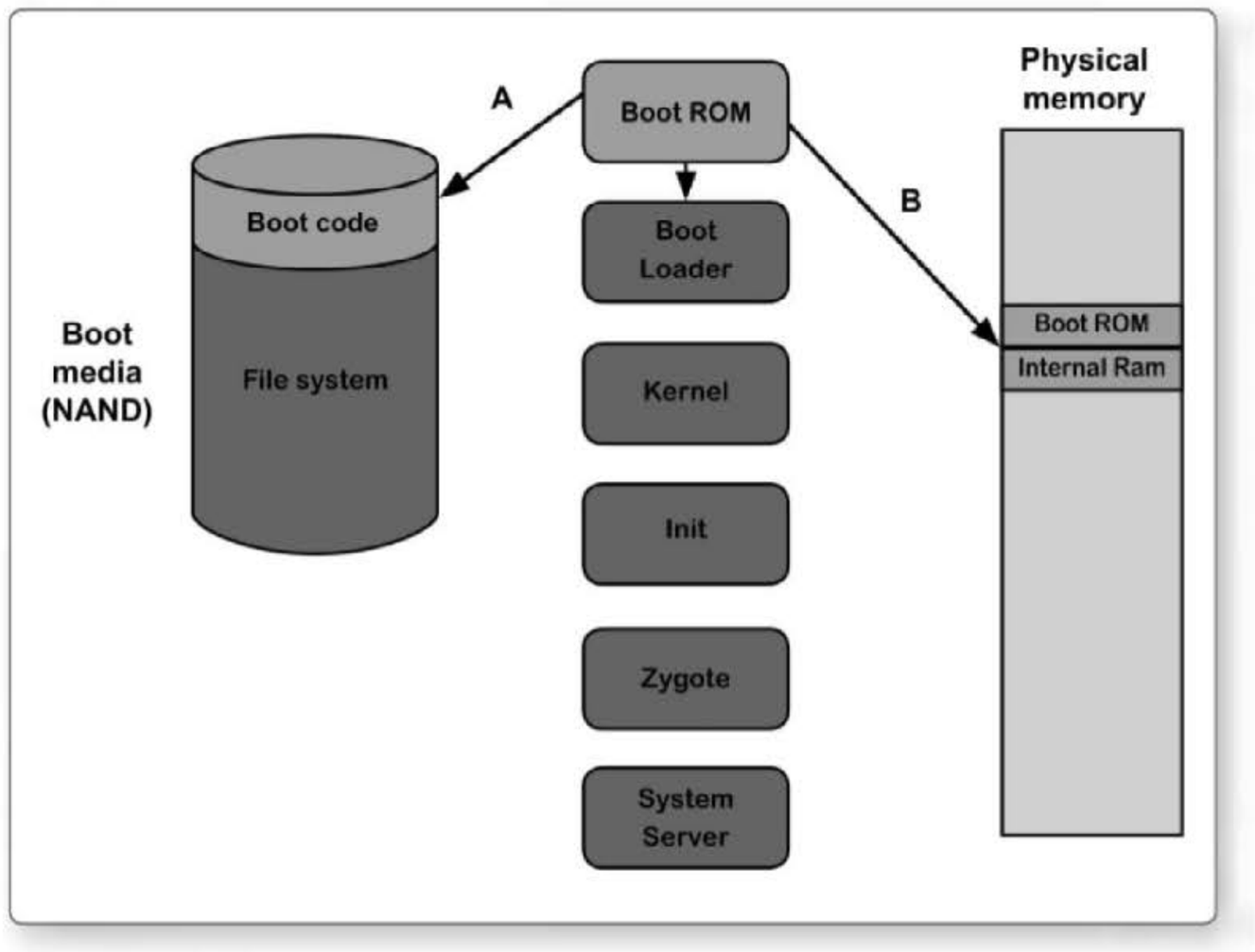
La comprensión de los componentes de hardware, tipos de dispositivo así como el proceso de arranque para Android ayudará a la comprensión global de Android.

13.4. ROM

Los dispositivos Android, como cualquier otro equipo, tienen un proceso de arranque bastante estándar que permite que el dispositivo cargue el firmware necesario, OS y los datos de usuario en la memoria a apoyo pleno funcionamiento. Aunque el proceso de arranque en sí está bien definido, el firmware ROM varía según el fabricante y el tipo de dispositivo. El objetivo de esta sección es proporcionar una visión general de alto nivel del proceso de arranque de Android. Esto sobre la base de la propuesta dada por Matías y otros en el Xdin Android Hub.

13.4.1. Proceso de inicio en Android

A. El encendido y la ejecución en el chip ROM de arranque código

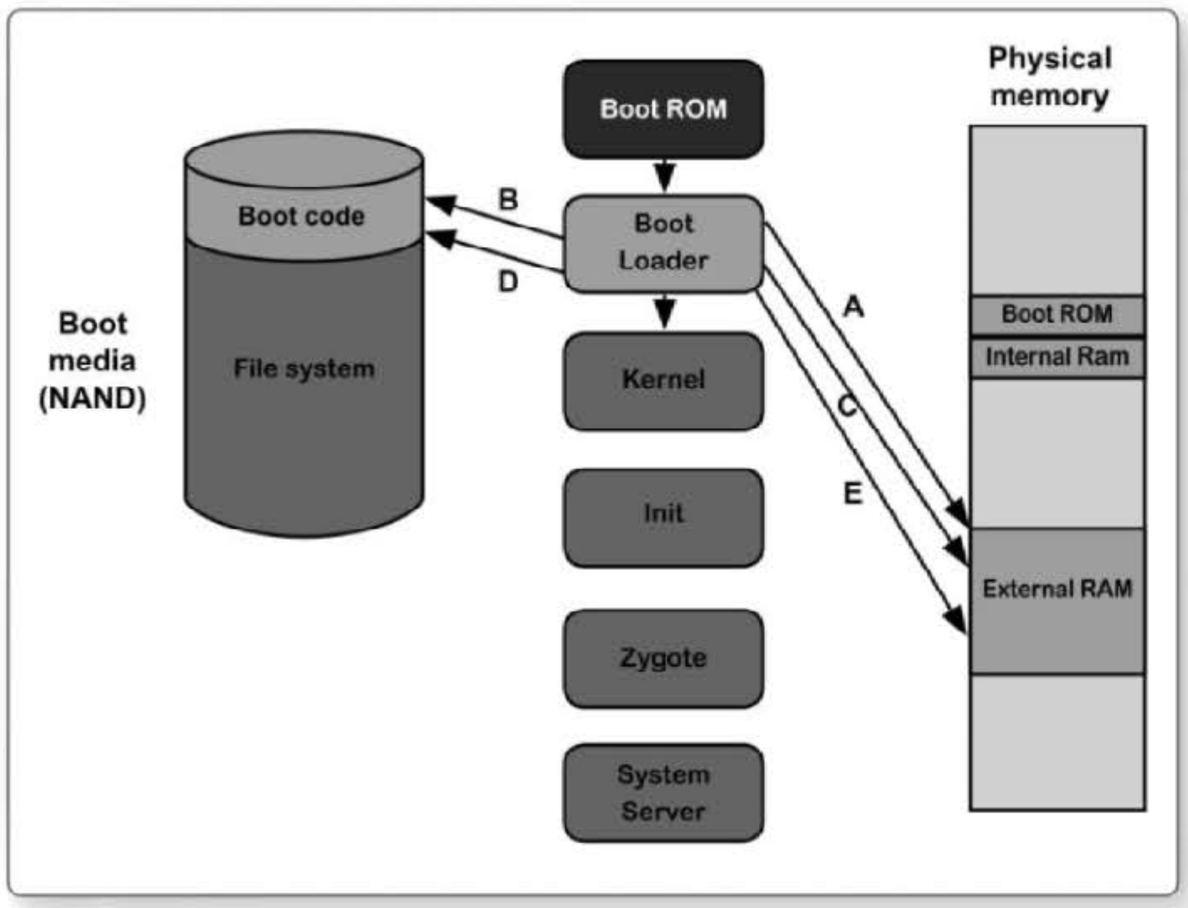


Al encender la CPU, estará en un estado donde no se han realizado las inicializaciones. Los relojes internos no están establecidos y la memoria disponible es la memoria RAM interna.
Cuando las fuentes de alimentación son estables a la ejecución comenzará con el código ROM de arranque. Esta es una pequeña pieza de código que está cableado en el ASIC CPU.

(A) El código ROM de arranque detectará el medio de arranque mediante un sistema de registro que se asigna a algunas bolas físicas en el ASIC. Esto es para determinar dónde se encuentra la primera etapa del cargador de arranque.

(B) Una vez que la secuencia de arranque de los medios de comunicación se ha establecido, la ROM de arranque intentará cargar el gestor de arranque primero en la RAM interna. Una vez que el gestor de arranque está en su lugar el código ROM de arranque llevará a cabo un salto y la ejecución continúa en el gestor de arranque.

13.4.2. Gestor de arranque



El **gestor de arranque** es un programa especial separado del núcleo Linux que se utiliza para crear recuerdos iniciales y cargar el kernel en la memoria RAM. En los sistemas de escritorio de los gestores de arranque son programas como GRUB y Linux embebido uBoot suele ser el gestor de arranque de la opción. Los fabricantes de dispositivos utilizan a menudo sus propios cargadores de arranque de propiedad.

Los requisitos de un gestor de arranque de Linux que se ejecuta en un sistema de brazo se puede encontrar en el documento de arranque en / Documentation / brazo en el árbol de fuentes del núcleo.

13.4.3. El Linux kernel

El núcleo de Linux se inicia de una manera similar en Android como en otros sistemas. Se creará todo lo que se necesita para que el sistema funcione. Inicializa controladores de interrupción, establecer protecciones de memoria, cachés y programación.

(A) Una vez que las unidades de gestión de memoria y las memorias caché se han inicializado el sistema será capaz de utilizar la memoria virtual y procesos de lanzamiento de espacio de usuario.
(B) El núcleo se verá en el sistema de archivos raíz para el proceso init (que se encuentra bajo el sistema / core / init en el árbol de código fuente abierta Android) y lanzarlo como el proceso del espacio de usuario inicial.

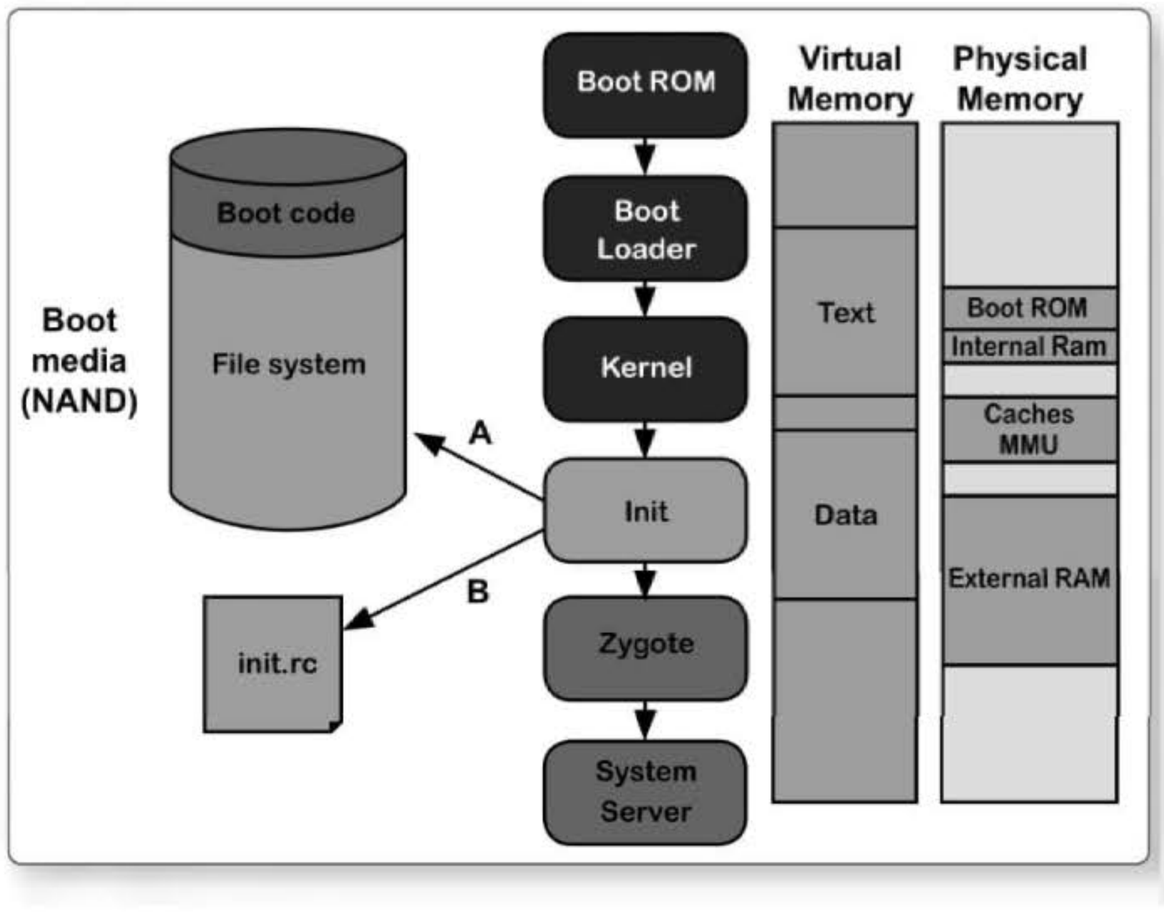
```
on init
sysclktz 0
loglevel 3
# setup the global environment
export PATH /sbin:/system/sbin:/system/bin:/system/xbin
export LD_LIBRARY_PATH /system/lib
export ANDROID_BOOTLOGO 1
export ANDROID_ROOT /system
export ANDROID_ASSETS /system/app
export ANDROID_DATA /data
export EXTERNAL_STORAGE /mnt/sdcard
export ASEC_MOUNTPOINT /mnt/asec
# Backward compatibility
```

```
symlink /system/etc /etc
symlink /sys/kernel/debug /d
# create mountpoints
mkdir /mnt 0775 root system
mkdir /mnt/sdcard 0000 system system
# Backwards Compat - XXX: Going away in G*
symlink /mnt/sdcard /sdcard
mkdir /system
mkdir /data 0771 system system
mkdir /cache 0770 system cache
mkdir /config 0500 root root
# Directory for putting things only root should see.
mkdir /mnt/secure 0700 root root
# Directory for staging bindmounts
mkdir /mnt/secure/staging 0700 root root
# Directory-target for where the secure container
# imagefile directory will be bind-mounted
mkdir /mnt/secure/asec 0700 root root
# Secure container public mount points.
mkdir /mnt/asec 0700 root system
mount tmpfs tmpfs /mnt/asec mode=0755,gid=1000
mount rootfs rootfs / ro remount
write /proc/sys/kernel/panic_on_oops 1
write /proc/sys/kernel/hung_task_timeout_secs 0
write /proc/cpu/alignment 4
write /proc/sys/kernel/sched_latency_ns 5000000
write /proc/sys/kernel/sched_wakeup_granularity_ns 100000
write /proc/sys/kernel/sched_min_granularity_ns 100000
write /proc/sys/kernel/sched_compat_yield 1
write /proc/sys/kernel/sched_child_runs_first 0
# mount mtd partitions
# Mount /system rw first to give the filesystem a chance to save
a checkpoint
mount yaffs2 mtd@system /system
mount yaffs2 mtd@system /system ro remount
Rom and boot loaders 53
```

13.4.4. Proceso init

El proceso init es el padre de todos los procesos del sistema. Cada proceso en el sistema se puso en marcha a partir de este proceso o uno de sus descendientes.

(A) El proceso init en Android buscará un archivo llamado init.rc. Este es un script que describe los servicios del sistema, sistema de archivos y otros parámetros que se deben configurar. El init.rc se coloca en system / core / rootdir en el proyecto de código abierto Android. (B) El proceso init analizará el inicio y pondrá en marcha los procesos de servicio del sistema.



Como se puede notar en su código fuente, aparte de crear algunos directorios, interpreta los ficheros:

```
...
parse_config_file("/init.rc");
...
get_hardware_name();
  snprintf(tmp, sizeof(tmp), "/init.%s.rc", hardware);
  parse_config_file(tmp);
```

El nombre del hardware se obtiene de "/proc/cpuinfo", en una N1.

```
# cat /proc/cpuinfo | grep Hardware
Hardware    : mahimahi
```

Los primeros pasos se encuentran en "init.rc":

```
head -30 init.rc | grep " "
on init
sysclktz 0
loglevel 3
# setup the global environment
  export PATH /sbin:/system/sbin:/system/bin:/system/xbin
  export LD_LIBRARY_PATH /system/lib
  export ANDROID_BOOTLOGO 1
  export ANDROID_CACHE /cache
  export ANDROID_ROOT /system
  export ANDROID_ASSETS /system/app
  export ANDROID_DATA /data
  export EXTERNAL_STORAGE /mnt/sdcard
  export ASEC_MOUNTPOINT /mnt/asec
  export
BOOTCLASSPATH/system/framework/core.jar:/system/framework/ext.jar:/system/framework/
framework.jar:/system/framework/android.policy.jar:/system/framework/services.jar
...
exec /system/bin/sysinit
class_start default
```

La importancia de Java en la plataforma es que desde el principio se define el CLASSPATH para el arranque. En este fichero, se configuran parámetros del núcleo, se crean sistemas de ficheros, se configuran los permisos, se hace una configuración inicial de la red, se definen varios servicios y se inician algunos "adbd".

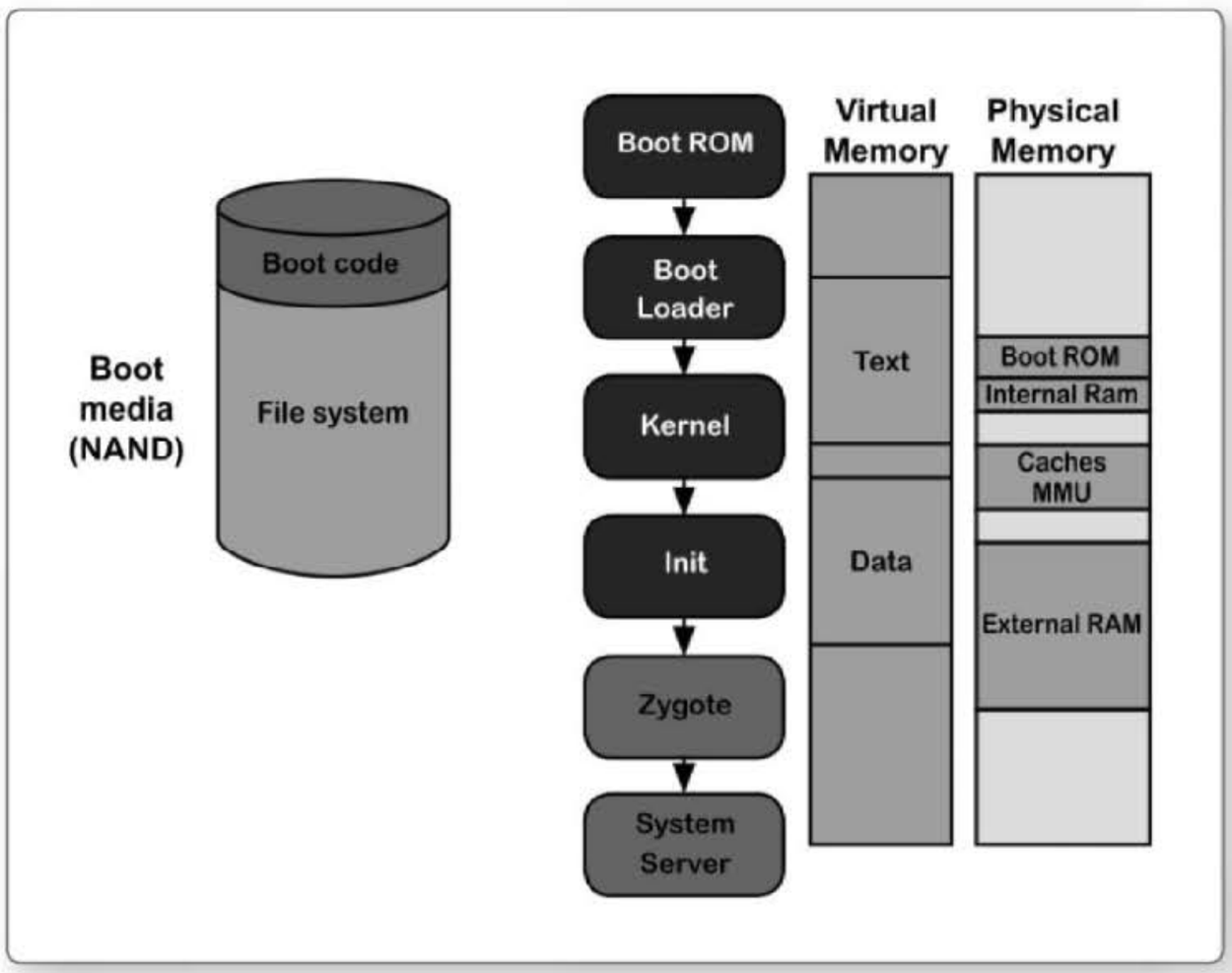
En "init.rc", ocurren también las particiones con el sistema completo como "/system" y se da paso al inicio habitual con los scripts dentro de "/system/etc/init.d" al ejecutar "/system/bin/sysinit".
Dentro de "init.mahimahi.rc", se inicializa todo el hardware y servicios asociados específicos del dispositivo y se intenta configurar la Wi-Fi.

Con "class_start default", se inician todos los servicios relacionados con la clase "default", salvo aquellos que tienen la opción "disabled". La gran mayoría de los servicios están en la clase "default". De entre todos los servicios que se inician el clave es Zygote y System Manager. Para ver todos los servicios disponibles, ingrese lo siguiente:

```
# /system/bin/service list
Found 52 services:
0    phone: [com.android.internal.telephony.ITelephony]
1    iphonesubinfo: [com.android.internal.telephony.IPhoneSubInfo]
...
50   media.player: [android.media.IMediaPlayerService]
51   media.audio_flinger: [android.media.IAudioFlinger]
```

13.4.5. Zygote y Dalvik

Zygote se encarga de lanzar la máquina virtual Dalvik; por lo que se tiene la plataforma Java en marcha.



Asimismo, Zygote es el proceso principal en la máquina virtual, responsable por el inicio y administración de componentes en que se basa Dalvik y sus privilegios. Precarga las bibliotecas necesarias y realiza un fork() de sí mismo para iniciar otros procesos.

El agujero se encuentra en el socket Zygote y afecta a dispositivos móviles tanto con versiones Android antiguas como con "Ice Cream Sandwich".

30/03/2012



Un equipo de investigadores italianos ha descubierto una vulnerabilidad en el sistema operativo Android ***desde versiones antiguas hasta la liberación 4.0.3*** *que deja los dispositivo móviles patrocinados por Google potencilamente (sic) abiertos a un ataque por denegación de servicio, lo que se conoce como DoS, y a la consecuente inhabilitación de sus recursos.*

Según han explicado Alessandro Armando, Alessio Merlo y Luca Verderame, de la Universidad de Génova, y Mauro Migliardi, de la Universidad de Padua, ***el agujero se encuentra en el socket Zygote de Linux,*** *que pusieron a prueba en varios productos como la Galaxy Tab 7.1 y los smartphones HTC Desire HD y LG Optimus One.*

Con la application DoSChecker, dispositivos con poca memoria como este último fueron "violados" en menos de un minuto. La tableta de Samsung, por su parte, tan sólo permaneció en pie un par de minutos. Junto a los niveles de seguridad, el equipo de investigación ha explicado que, mientras el ataque de denegación se producía, ***"los usuarios experimentan una reducción progresiva de la capacidad de respuesta del sistema, que termina con su caída y reinicio"****.*

Una vez que el dispositivo está intentando recuperarse, los ciberdelincuentes podrían emplear software malicioso para ejecutar DoSChecker como servicio de arranque, forzando el bloqueo continuo del sistema.

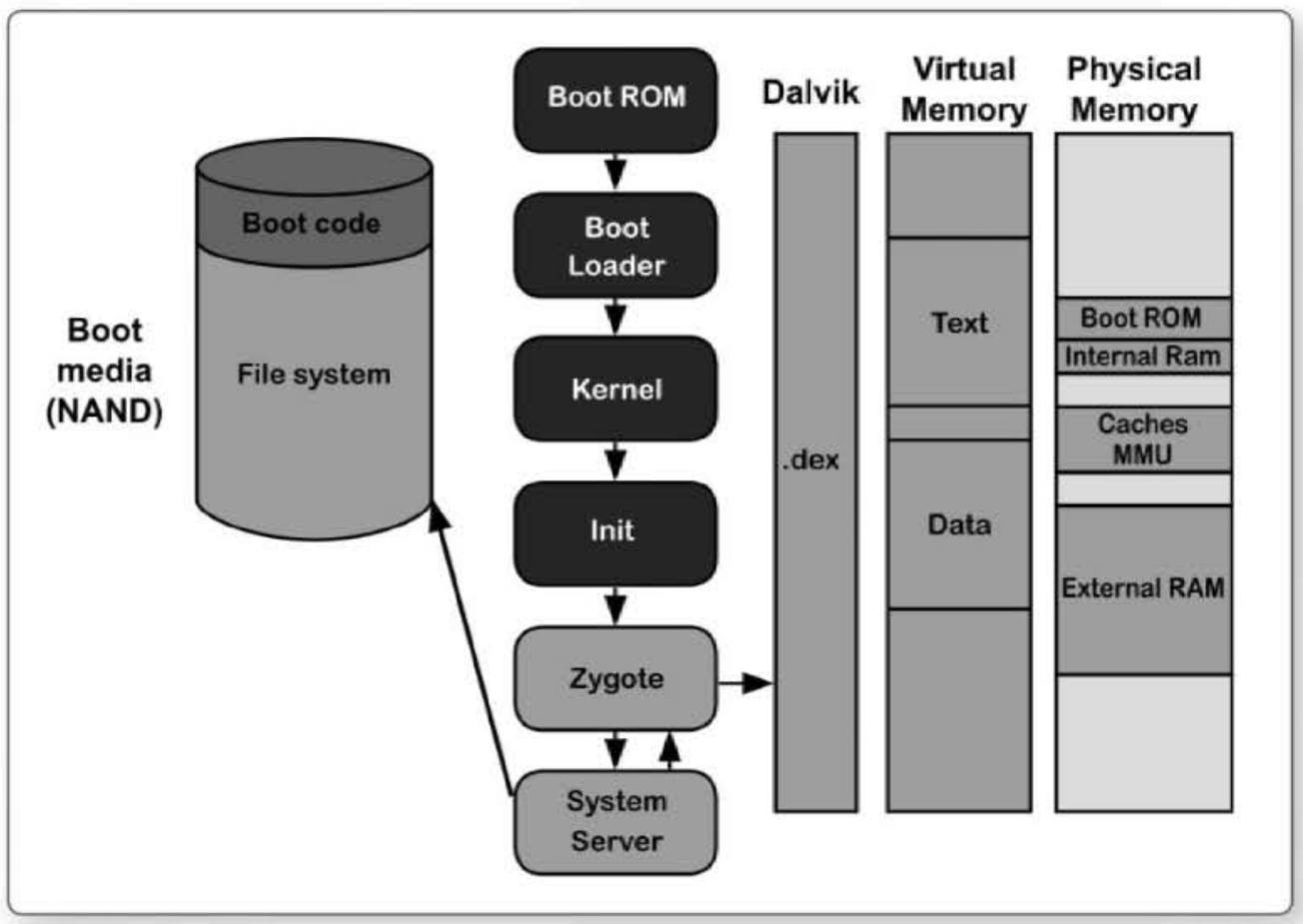
Para atajar este problema, los expertos sugieren bien revisar el proceso de verificación de solicitudes de Zygote, bien restringir los permisos del socket. Desde Google ya han comunicado que parchearán el exploit en la próxima actualización de software utilizando una de estas dos posibilidades.

Fuente: eWeek Europe (obtenido de www.csirtcv.gva.es).

13.4.6. El servidor del sistema

El servidor del sistema es el primer componente de Java que se ejecuta en el sistema. Se iniciará todos los servicios de Android, como gerente de telefonía y Bluetooth. Puesta en marcha de cada servicio está escrita directamente en el método de ejecución del servidor del sistema. La fuente del servidor del sistema se puede encontrar en el proyecto de código abierto:

frameworks/base/services/java/com/android/server/SystemServer.java



13.4.7. Arranque completado

Una vez agregada a esta parte del mensaje el "20090831", ya que es útil y es algo que no debería haber salido desde el principio. Si el servidor del sistema está en funcionamiento y el sistema de arranque ha terminado, hay una acción de difusión estándar llamado ACTION_BOOT_COMPLETED. Para iniciar su propio servicio y registrar una alarma o poner a su aplicación, realice alguna acción después del arranque y debe registrarse para recibir esta intención difusión.

13.5. Uso del Linux dentro del dispositivo móvil Android con terminal Emulador



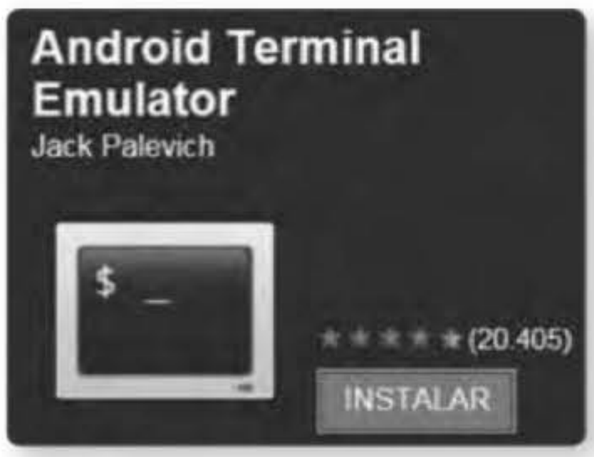
Android *Terminal Emulator* es una aplicación que le permite acceder a Android incorporado en el shell de Linux, línea de comandos. Emula un Digital Equipment Corporation terminal VT-100 que puede utilizarse para ejecutar programas de utilidad en un dispositivo.

Esta aplicación le permite ejecutar utilidades de línea de comandos Linux en su teléfono Android. Es sobre todo útil para los programadores de computadoras. También podría ser útil para las personas que quieran explorar el funcionamiento interno de Linux de su teléfono Android.

El **terminal** es una herramienta útil para el sistema operativo, principalmente para los usuarios de Linux, ya que permite realizar diferentes tareas, actualizar todo el sistema (programas, librerías, repositorios, etc.); del mismo modo, se puede (des)instalar programas, crear directorios o eliminarlos, copiar, cortar y mover archivos, cambiarle los permisos, ejecutar programas, etc.

Se debe conocer los comandos básicos para navegar por los archivos internos del Android y hacer una copia de seguridad de la carpeta más importante del teléfono (la carpeta EFS) que contiene datos importantes como el IMEI. La copia siempre se hace dentro de SD interna. Para poder desempeñar esta operación con éxito, tiene que ser usuarios root, lo que significa, que tiene que tener "roteado" su dispositivo. Asimismo, puede descargar la aplicación gratuita desde el Market llamada Terminal Emulator.

https://play.google.com/store/apps/details?id=jackpal.androidterm&feature=search_result



13.5.1. Comandos básicos

Son los comandos básicos que se va a usar o simplemente que se debe conocer.

ls	Lista los archivos y directorios.
cd	Cambio de directorios.
cd -	Vuelve al directorio inmediatamente anterior.
mkdir	Crea un directorio.
rmdir	Borra un directorio.
cp	Copia un archivo.
rm	Borra un archivo.
mv	Mueve un archivo.



Estas sencillas órdenes sirven para moverse por el sistema de archivos del teléfono y crear la copia de seguridad de la carpeta EFS.

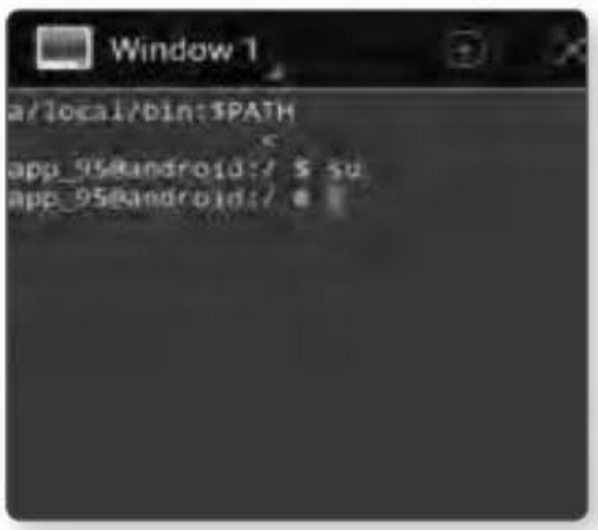


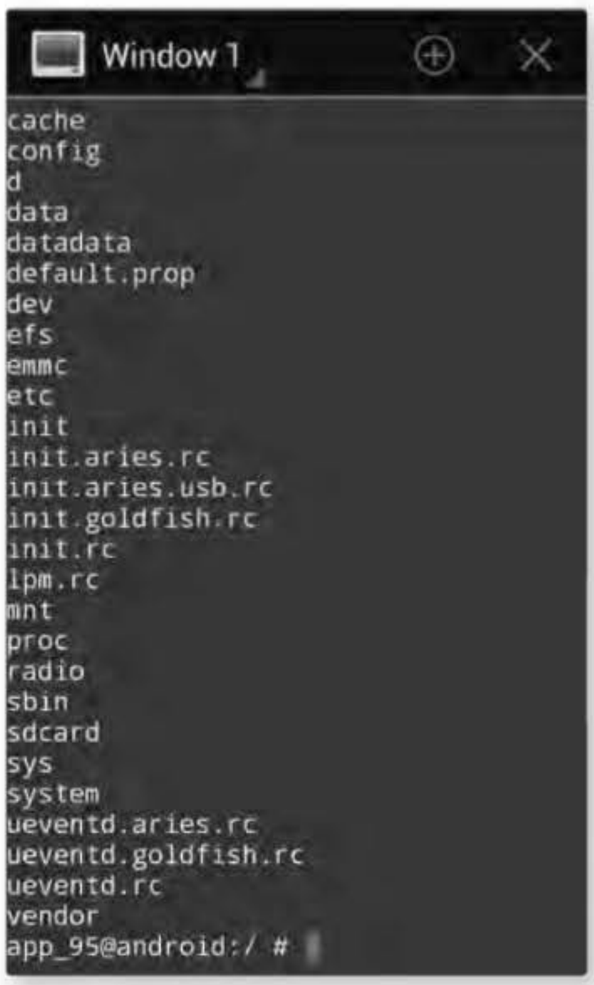
Paso 1: Abra la aplicación **Terminal Emulator.**

Paso 2: Digite "su" para trabajar con permisos de superusuario.

> ***Nota:***
>
> *Cabe advertir que con este permiso, se accede a todo el sistema y se puede modificar o alterar sino se maneja apropiadamente; ya que se podría eliminar algún archivo esencial para el funcionamiento del dispositivo.*

Paso 3: Se cambiará del símbolo **Dólar** al símbolo **Cuadradillo**; esto significa que se tiene el permiso para modificar el sistema.





Paso 4: Teclee "ls" para ver la lista de archivos y directorios desde donde se inicia.

En la ventana de arriba, se observa la carpeta "efs" para copiar y el directorio "sdcard", que es el sitio donde se guardará la copia de seguridad.

Paso 5: Escriba "cd sdcard" para entrar dentro de la memoria interna.

Paso 6: Cree una carpeta nueva llamada "copiaefs", digitando "mkdir copiaefs":

```
cd
sdcard
mkdir
copiaefs
```



Paso 7: Con esto ya se obtiene la carpeta copiaefs dentro del sdcard, aquí se copia el directorio efs.

Paso 8: Regrese al directorio principal donde está la carpeta efs que quiere copiar. Para ello, escriba "cd -" y presione **Enter**.

Paso 9: Digite "cp -r efs /sdcard/copiaefs" y se observará lo siguiente:

```
cd cp -r efs /sdcard/copiaefs
```

Nota:

El comando cp copia archivos, como efs es una carpeta con archivos dentro, después del comando cp se adjunta "-r" para indicar que es una carpeta para copiarla con todo su contenido.



Paso 10: Para verificar que ya se cuenta con una copia de seguridad en la sdcard y poder ingresar dentro de la carpeta copiaefs, escriba lo siguiente:

```
cd sdcard/copiaefs
```

El comando Ls que dentro de la carpeta copiaefs se ha creado otro directorio llamado «efs».

Paso 11: Si ahora se teclea lo siguiente:

```
cd efs
ls
```

estará dentro de la carpeta efs que se acaba de copiar y, al haber puesto el comando ls, se podrá comprobar que dentro de ella existen varios archivos que son los que se acaban de copiar anteriormente.



Estos pasos sirven para salvar la carpeta efs a la sdcard, ahora solo queda conectar el dispositivo al ordenador y sacar la carpeta copiaefs y guardarla bien segura.

13.6. INGENIERIA INVERSA BÁSICA

«Ingeniería inversa



El objetivo de la ingeniería inversa es obtener información o un diseño a partir de un producto accesible al público, con el fin de determinar de qué está hecho, qué lo hace funcionar y cómo fue fabricado.

Hoy en día (principios del siglo XXI), los productos más comúnmente sometidos a ingeniería inversa son los programas de computadoras y los componentes electrónicos, pero, en realidad, cualquier producto puede ser objeto de un análisis de Ingeniería Inversa.

El método se denomina así porque avanza en dirección opuesta a las tareas habituales de ingeniería, que consisten en utilizar datos técnicos para elaborar un producto determinado. En general, si el producto u otro material que fue sometido a la ingeniería inversa fue obtenido en forma apropiada, entonces el proceso es legítimo y legal. De la misma forma, pueden fabricarse y distribuirse, legalmente, los productos genéricos creados a partir de la información obtenida de la ingeniería inversa, como es el caso de algunos proyectos de Software libre ampliamente conocidos.

El programa Samba es un claro ejemplo de ingeniería inversa, dado que permite a sistemas operativos UNIX compartir archivos con sistemas Microsoft Windows. El proyecto Samba tuvo que investigar información confidencial (no liberada al público en general por Microsoft) sobre los aspectos técnicos relacionados con el sistema de archivos Windows. Lo mismo realiza el proyecto WINE para el conjunto de API de Windows y OpenOffice.org con los formatos propios de Microsoft Office, o se hace para entender la estructura del sistema de archivos NTFS y así poder desarrollar drivers para la lectura-escritura sobre el mismo (principalmente para sistemas basados en GNU/Linux).

La ingeniería inversa es un método de resolución. Aplicar ingeniería inversa a algo supone profundizar en el estudio de su funcionamiento, hasta el punto de que podamos llegar a entender, modificar y mejorar dicho modo de funcionamiento.

Pero este término no sólo se aplica al software, sino que también se considera ingeniería inversa el estudio de todo tipo de elementos (por ejemplo, equipos electrónicos, microcontroladores, u objeto fabril de cualquier clase). Diríamos, más bien, que la ingeniería inversa antecede al nacimiento del software, tratándose de una posibilidad a disposición de las empresas para la producción de bienes mediante copiado [...] desde el mismo surgimiento de la ingeniería.

En el caso concreto del software, se conoce por ingeniería inversa a la actividad que se ocupa de descubrir cómo funciona un programa, función o característica de cuyo código fuente no se dispone, hasta el punto de poder modificar ese código o generar código propio que cumpla las mismas funciones. La gran mayoría del software de pago incluye en su licencia una prohibición expresa de aplicar ingeniería inversa a su código, con el intento de evitar que se pueda modificar su código y que así los usuarios tengan que pagar si quieren usarlo.

La ingeniería inversa nace en el transcurso de la Segunda Guerra Mundial, cuando los ejércitos enemigos incautaban insumos de guerra como aviones u otra maquinaria de guerra para mejorar las suyas mediante un exhaustivo análisis».

Fuente: http://es.wikipedia.org

13.6.1. Herramientas a utilizar

Para la preparación del entorno, debe tener en cuenta desde los requisitos que son la lógica de un lenguaje de programación (es recomienda Java) y los siguientes:

- *La plataforma Java* (advierta que no basta con el entorno de ejecución JRE), todas las herramientas para desarrolladores (JDK, *Java Development Kit*) y un IDE cualquiera (Eclipse, por ejemplo). Para ello, visite la siguiente página:

  http://www.oracle.com/technetwork/java/javase/downloads/index.html

- Las herramientas de desarrollo de Android o *SDK*, en el portal oficial hay una guía de instalación (http://developer.android.com/sdk/installing.html).
- Un *emulador* o el AVD configurado permitirá ejecutar una máquina virtual de Android en los ordenadores (Fuente: http://developer.android.com/sdk/adding-components.html).
- *Smali/Baksmali* que se trata de un (des)ensamblador de ficheros APK y sirve para generar las fuentes en lenguaje de bajo nivel (aunque como tal no se usará en esta guía). Descárguelo de:

  http://code.google.com/p/smali/

- *APKTool* es una herramienta que sustituirá el Smali/Baksmali; ya que los integra dentro de su paquete. Como su nombre lo indica es un kit de herramientas para trabajar con ficheros APK. Descárguelo de:

  http://code.google.com/p/android-apktool/

  Con este se genera el código de bajo nivel y luego se recompila.

- T*estsign.jar* es una herramienta de terceros, desarrollada para "firmar" o generar certificados aleatorios e indicarle al sistema que el fichero modificado es una aplicación Android. Se descarga desde:

  http://code.google.com/p/zen-droid/downloads/list (buscar el fichero testsign.jar)

- *Zipalign* es una herramienta que viene incluida en el SDK (PATH\android-sdk-windows\tools\ zipalign.exe) y que cumple la función de preparar el APK para el sistema Android; es decir, para que su rendimiento sea exacto para dicho sistema ya que si no está comprimida, con esta herramienta podría afectar el rendimiento de la misma en el dispositivo.

13.6.2. Formatos de archivo .DEX y .JAR

Como se sabe la mayoría de aplicaciones para Android están programadas en Java. Los ejecutables de este sistema operativo tienen una extensión de tipo .APK, Es preciso indicar que esta extensión no es más que una variación de la extensión .JAR, por lo que si alguna vez se programa en Java, el .JAR funcionará como una especie de contenedor con varios archivos adentro, por lo que con cualquier compresor se puede extraer su contenido y ver una serie de archivos que son los que componen la aplicación.

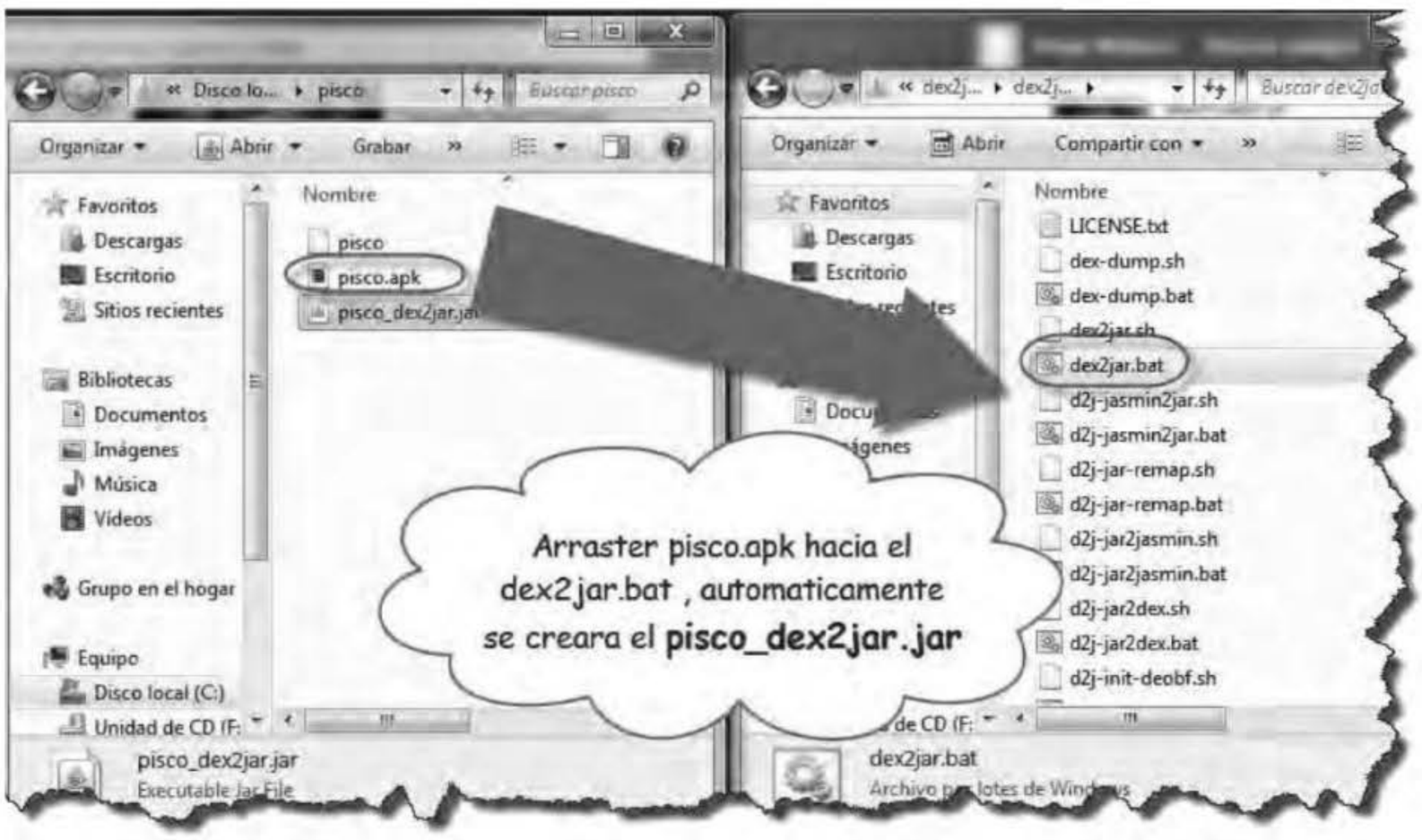
Por lo tanto, si a una aplicación Android (.APK) se le extrae los ficheros con un compresor cualquiera, se verán todos los ficheros que componen la aplicación.

13.6.3. Contenido del archivo pisco.PAK



Paso 1: Use el dex2jar.

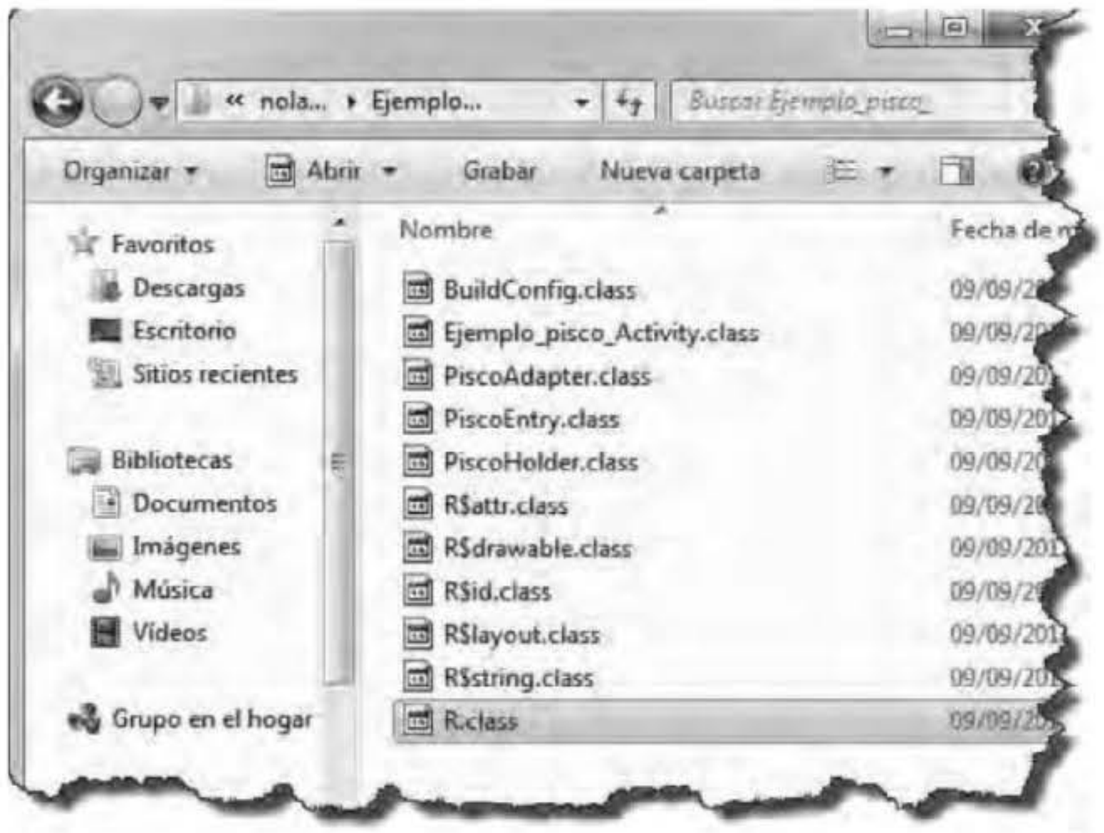
Paso 2: Convierta el archivo APK a JAR.



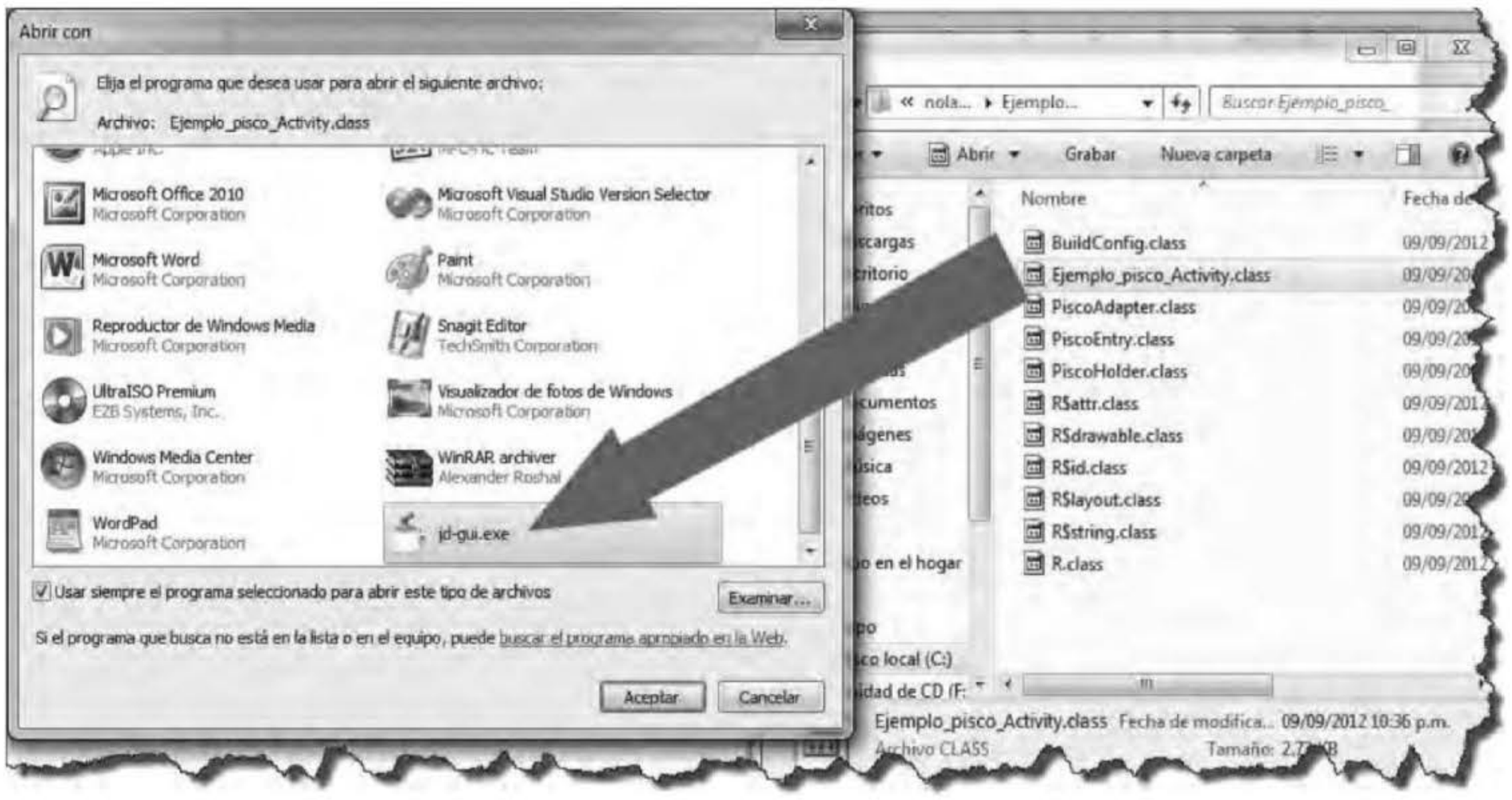
Paso 3: Renombre la extensión del Jar por Zip, obteniendo lo siguiente:



Paso 4: Desempaque el archivo pisco_dex2jar.zip.



Paso 5: Utilice el decompilador jd-gui.exe.



13.7. SANTOKU

Santoku es una distribución Linux basada en OWASP's MobiSec especializada en pruebas de seguridad, análisis de malware y análisis forenses para teléfonos móviles; válida para dispositivos con Android, BlackBerry, iOS y Windows Phone.

La versión *Santoku Community Edition* es un proyecto colaborativo para proveer un entorno Linux preconfigurado con utilidades, drivers y guías para este campo. La versión Alpha puede descargarse desde:

https://santoku-linux.com/download

Santoku incluye las siguientes herramientas:

HERRAMIENTAS DE DESARROLLO	PENETRATION TESTING:	INGENIERÍA INVERSA:	ANALIZADORES WIRELESS:	FORENSE DE DISPOSITIVOS:	INFRAESTRUCTURA MÓVIL:
• Android 2.3.3, 3.2 y 4.0.3 Emulators • Android SDK Manager • Apple Xcode IDE • BlackBerry JDE • BlackBerry Ripple • BlackBerry Simulators • BlackBerry Tablet OS SDK • BlackBerry WebWorks • DroidBox • Eclipse IDE • SecurityCompass Lab Server (HTTP y HTTPS) • Windows Phone SDK	• BeEF • Burp Suite • CeWL • DirBuster • Ettercap • Fierce • iSniff • Mallory • Metasploit Console • Metasploit GUI • NetSed • Nikto • nmap • SET • SQLMap • SSLStrip • w3af Console • w3af GUI • ZAP	• APK Tool • Dex2Jar • Flawfinder • Java Decompiler • Strace	• Aircrack-ng • Kismet • Ubertooth Kismet • Ubertooth Spectrum Analyzer • Wireshark	• AFLogical Open Source Edition • Android Encryption Brute Force • BitPim • BlackBerry Desktop Manager • Foremost • iPhone Backup Analyzer • MIAT • Paraben Device Seizure • Sift Workstation • Sleuth Kit • SQLiteSpy	• BES Express • Google Mobile Management • iPhone Configuration Tool

13.7.1. Requerimientos para la instalación

- Santoku–Alpha 0.1 (o posteriores)
- Virtual Box or VMWare Player
- Una máquina con un mínimo de procesador de doble núcleo, 2 Gb de RAM y 40 Gb de espacio libre en el disco duro o más recomendado

13.7.2. Configurar su máquina virtual

Para ejecutar Santoku, debe instalar el software de la máquina virtual. En esta sección, se va a usar la versión más reciente de VirtualBox, versión 4.1.18.

Paso 1: Descárguela desde:

http://www.virtualbox.org/wiki/Downloads

Paso 2: Instale el software de la máquina virtual en la estación de trabajo forense y siga las instrucciones para iniciar la máquina virtual (VM).

Paso 3: Busque la instalación de VirtualBox.

Paso 4: Seleccione Nuevo para crear una nueva máquina virtual y siga el paso por el asistente.



Paso 5: Cree un nombre para la máquina virtual y seleccione el sistema operativo.



Paso 6: Seleccione una cantidad adecuada de memoria para la máquina virtual.

- 512, por ejemplo; sin embargo el aumento de la capacidad de memoria por lo general hará que VM sea más rápido (pero el equipo más lento).
- Si usted va a utilizar el administrador de dispositivos Android Virtual (AVD) y el dispositivo Android emulador con frecuencia, se recomienda seleccionar al menos 4 Gb de memoria.

Paso 7: En el disco duro virtual de pantalla, asegúrese de que Start-up del disco esté marcado

Paso 8: Seleccione Crear nuevo disco duro. Para crear un disco duro nuevo, seleccione la opción VDI (imagen de VirtualBox Disk).



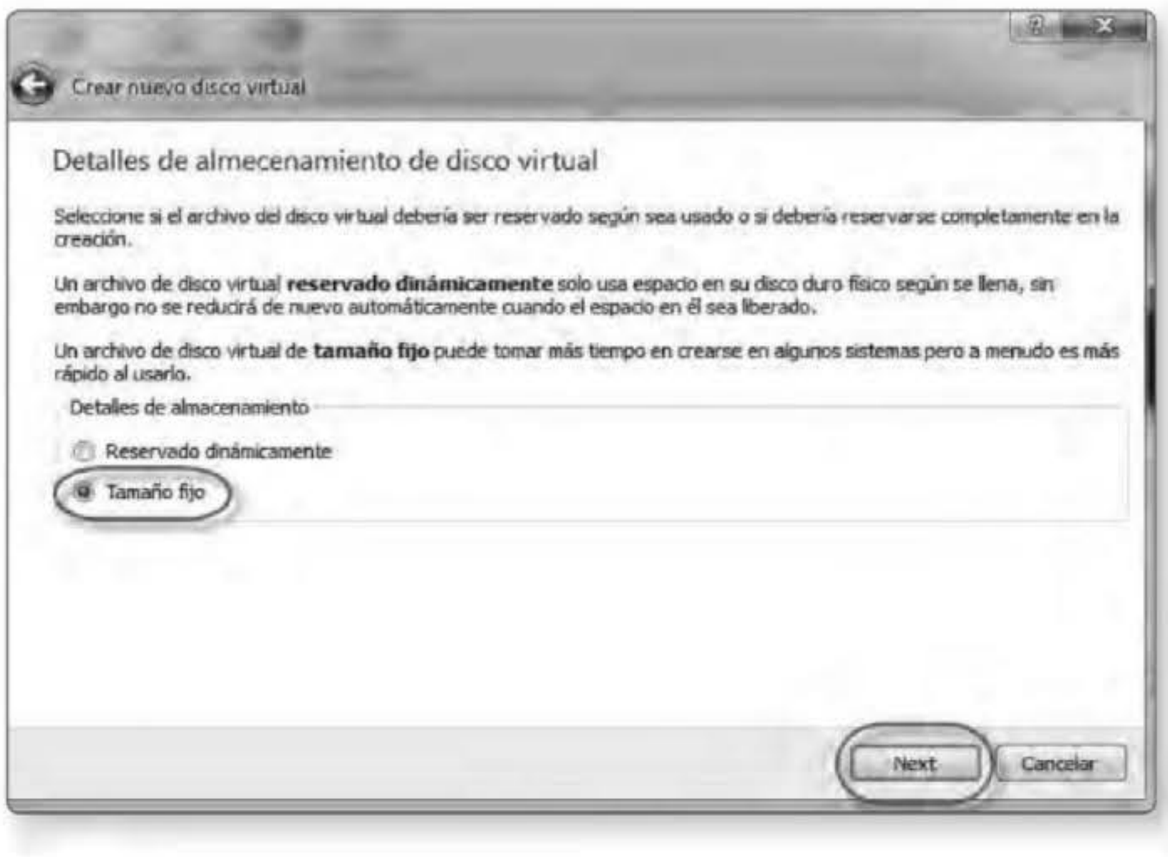
Paso 9: Seleccione *Crear un disco virtual nuevo*.



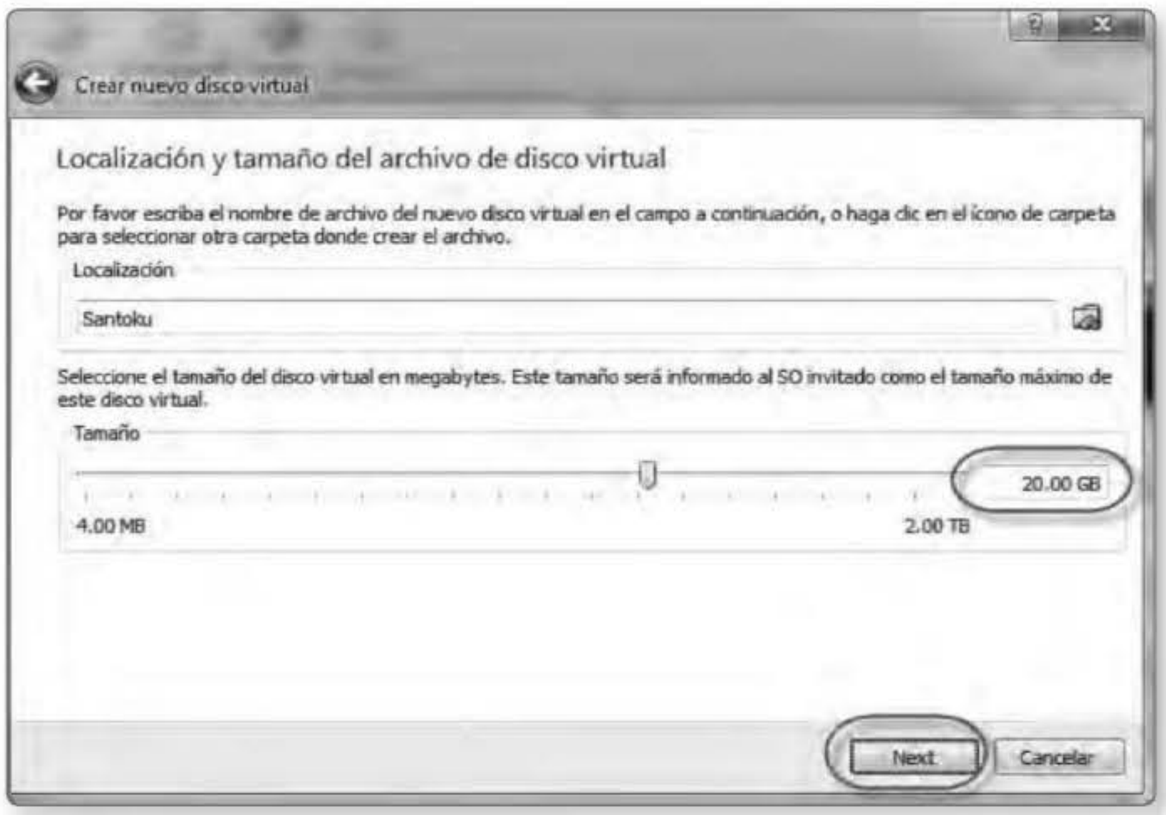
Paso 10: Seleccione VDI.



Paso 11: Seleccione *Tamaño fijo*.



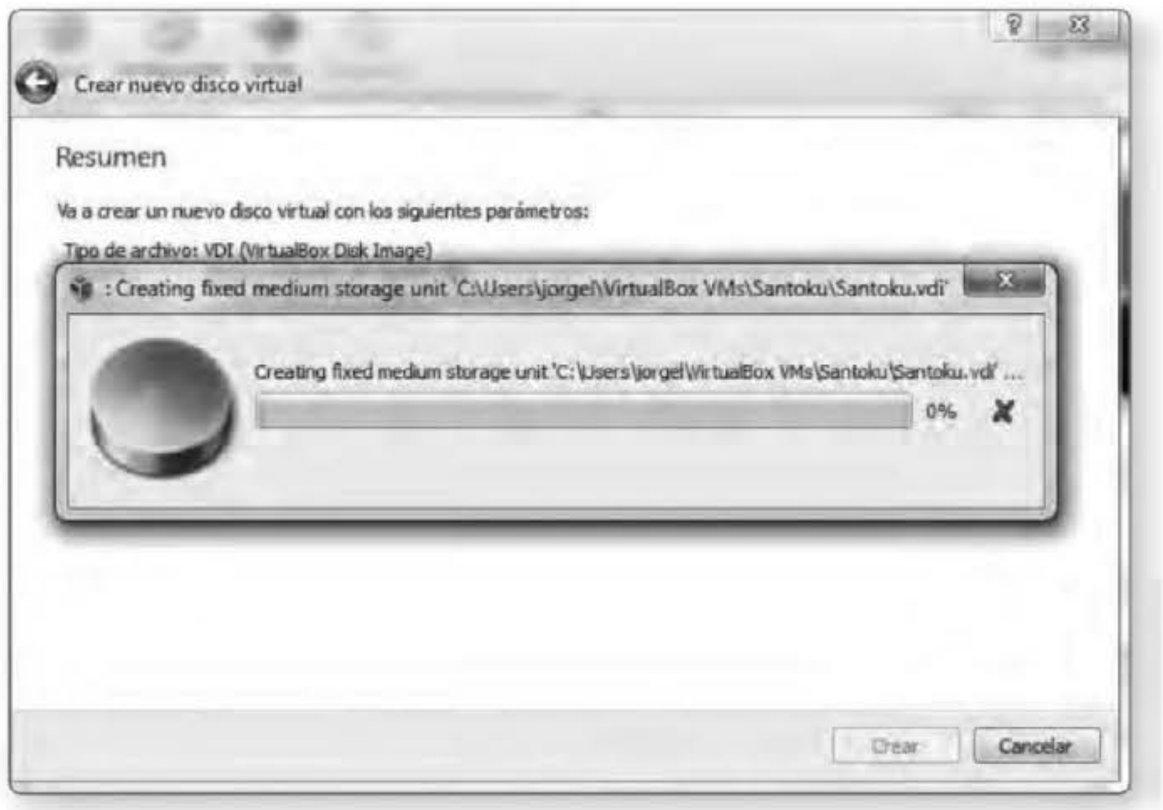
Paso 12: Seleccione el tamaño del disco virtual a 20 Gb.



Paso 13: Presione el botón *Crear.*



Espere mientras se crea el disco virtual.



Paso 14: Inicie la máquina virtual.



13.7.3. Instalar Santoku

Una vez instalada haga clic en *Inicio* en la pantalla principal de VirtualBox para cargar la máquina virtual.

Paso 1: Seleccione arrancar desde el DVD en directo o instalar. Si ya lo ha creado, estará en una máquina virtual.

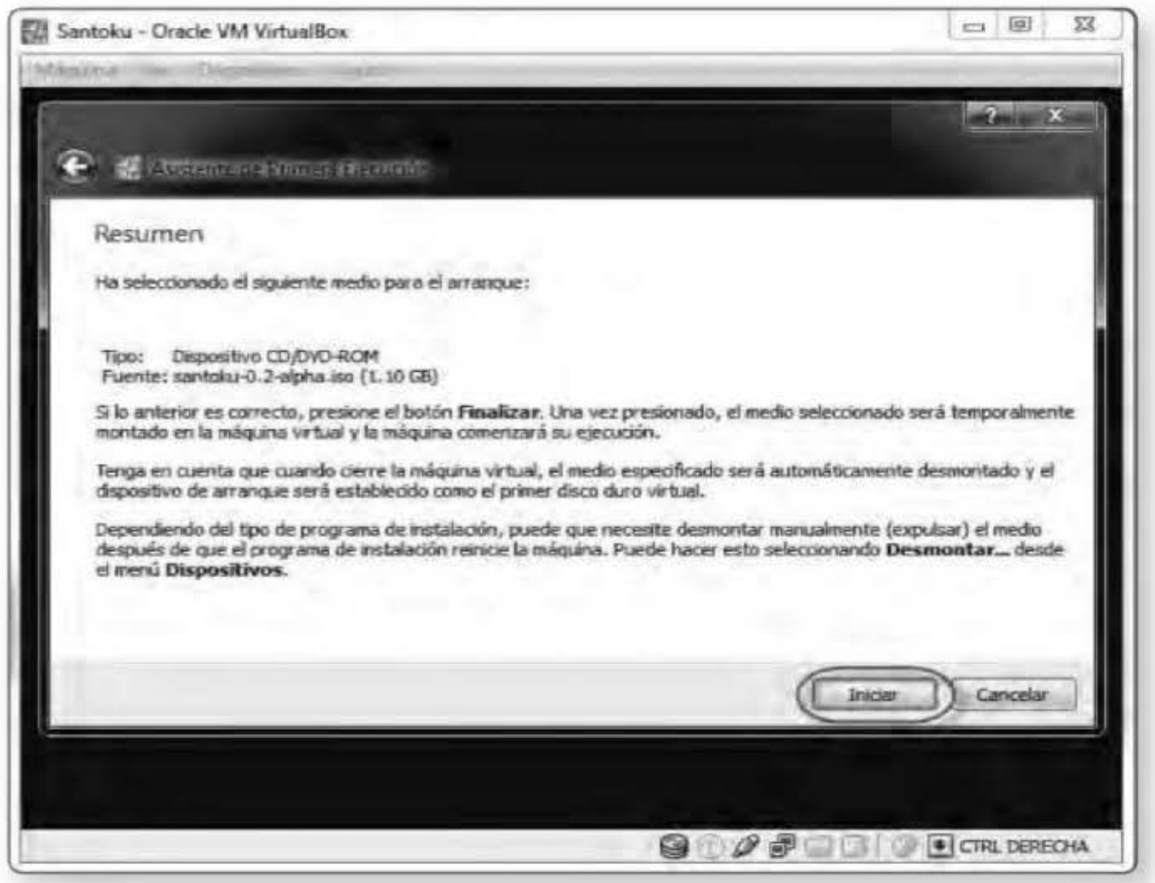
Paso 2: Seleccione *Install-iniciar el instalador directamente*.



Paso 3: Seleccione el medio que contiene el ISO.



Paso 4: Presione el botón *Instalar.*



Paso 5: Seleccione Live–boot the Live System.



Paso 6: Haga clic en Instalar.



Paso 7: Seleccione el idioma y presione el botón Continuar.



Paso 8: Presione nuevamente el botón Continuar.



Paso 9: Presione el botón *Instalar ahora*.



Paso 10: Verifique la configuración y presione el botón *Continuar.*



Paso 11: Seleccione el idioma y presione el botón *Continuar*.



Paso 12: Ingrese las credenciales a utilizar y presione el botón *Continuar.*



Paso 13: Instale.



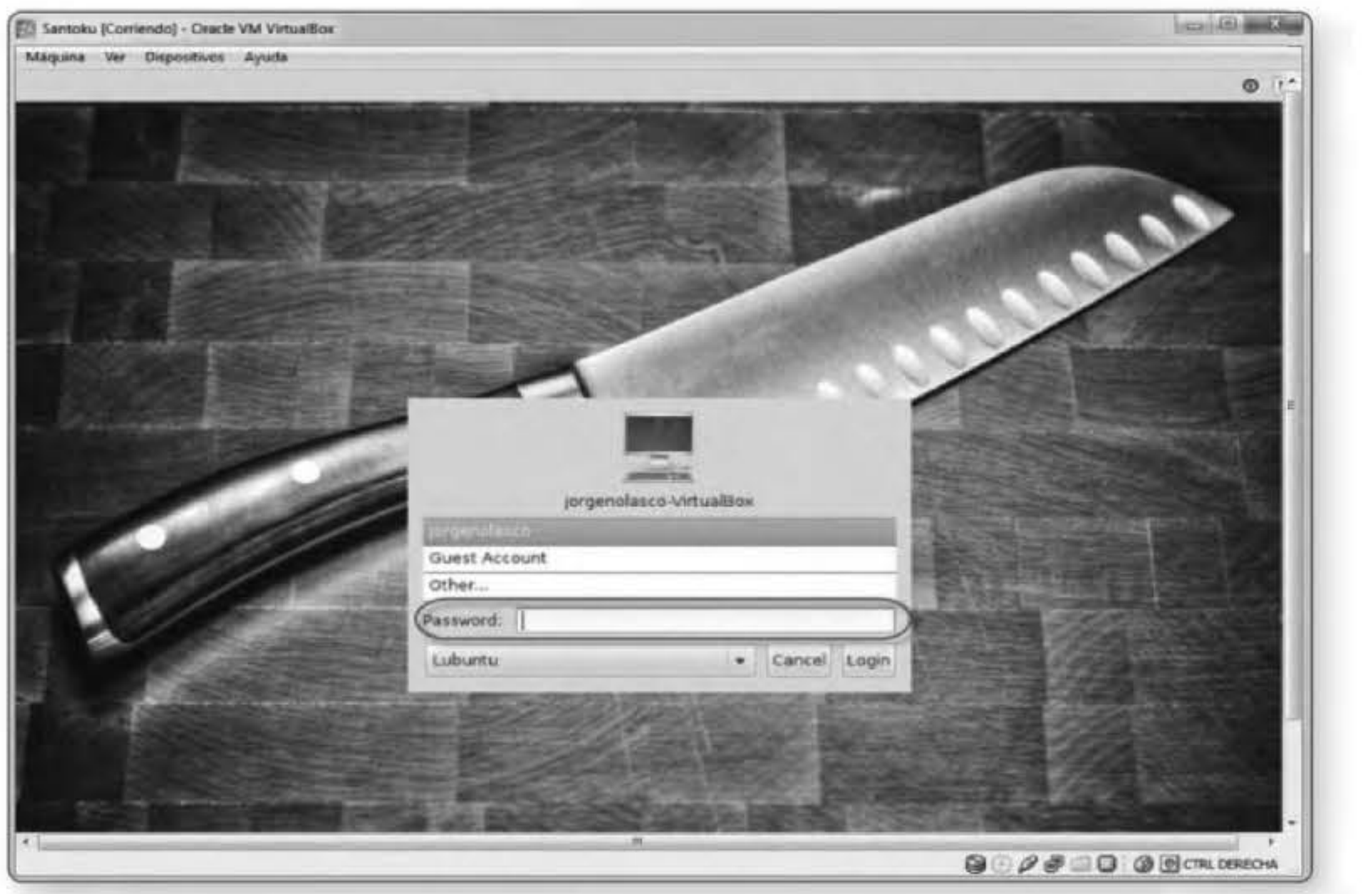
Paso 14: Presione el botón *Reiniciar ahora*.



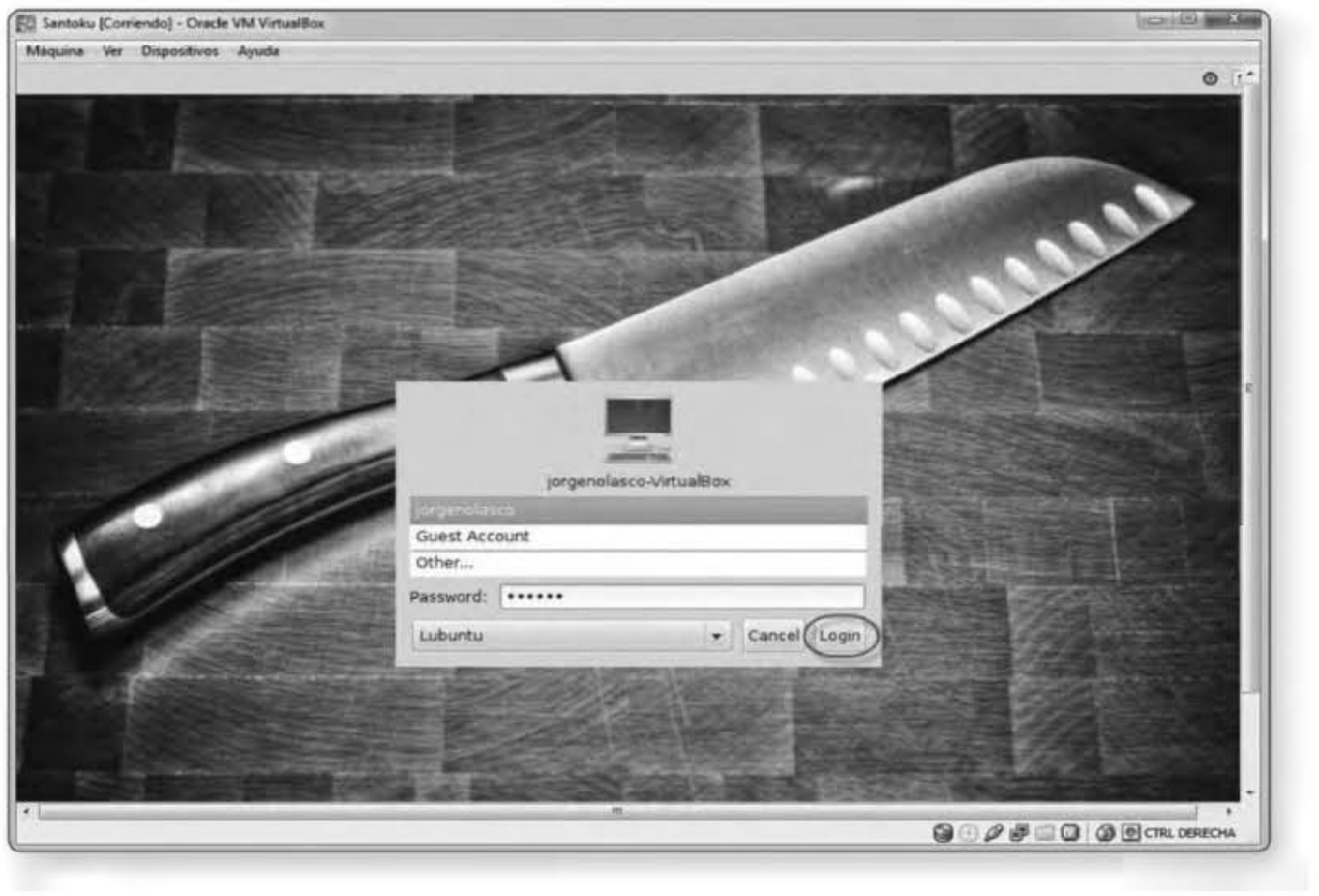
Paso 15: Inicie Lubuntu.



Paso 16: Ingrese la clave.



Paso 17: Presione el botón *Login*.



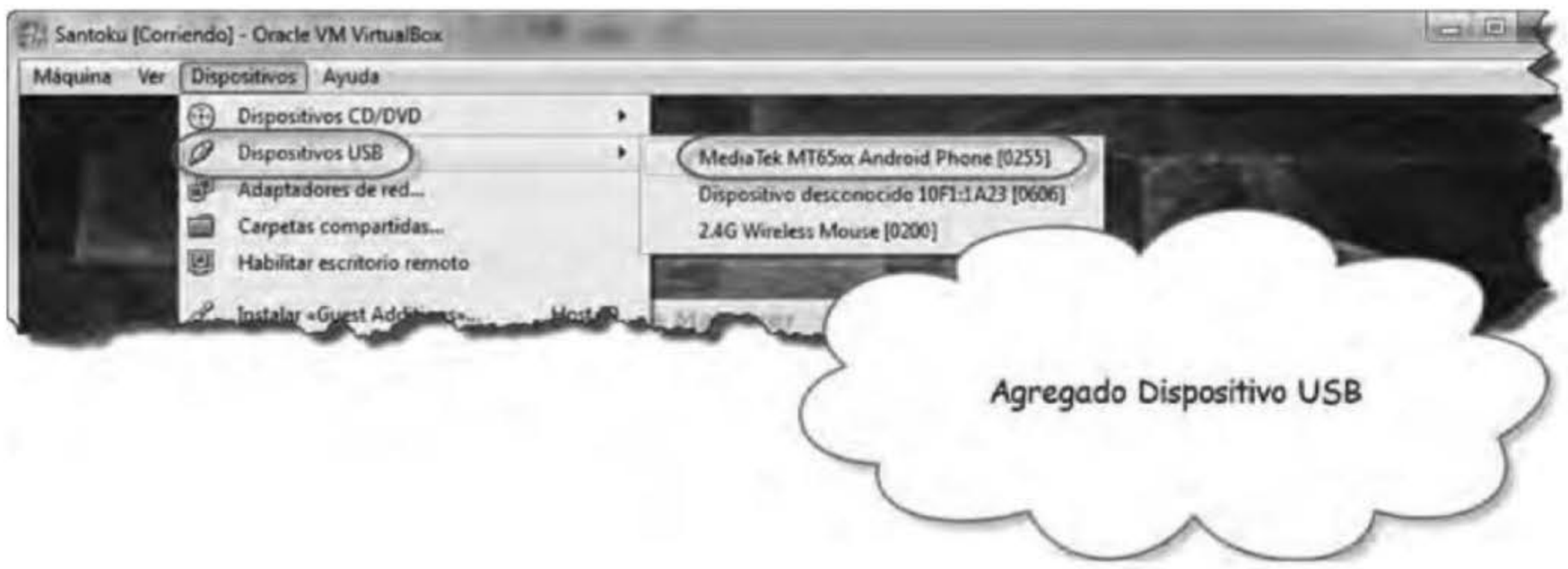
Paso 18: Es importante tener en cuenta, cómo un dispositivo Android se conecta a una máquina virtual. Los dispositivos Android tienen una interfaz USB físico que les permite conectar, compartir datos y recursos y, por lo general, recargar desde un ordenador o una estación de trabajo.

- Si solo está ejecutando un sistema operativo, el dispositivo usb deberá tener detectado y accesible. sin embargo, la configuración adicional o controladores podrían estar requeridos.
- Si está ejecutando una máquina virtual y desea que el sistema operativo anfitrión pase a la conexión a través de la máquina virtual.

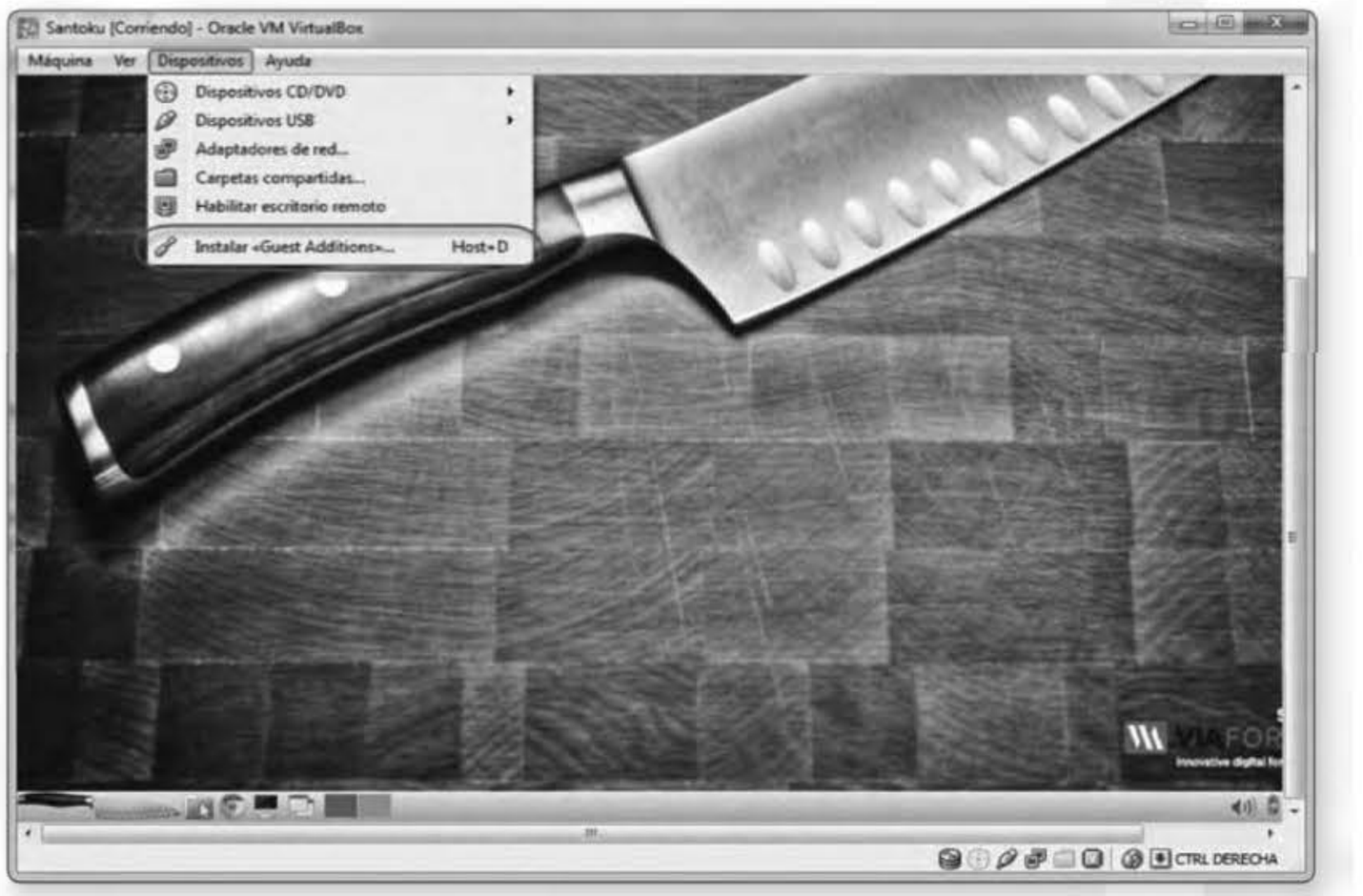
Por ejemplo, si el sistema operativo anfitrión OS X y se está ejecutando VMware Fusion, se seleccionan los menús de Virtual Machine/USB y luego se conecta el dispositivo (High teléfono Android, en este caso).

Del mismo modo, cuando el sistema operativo anfitrión es Linux y se está ejecutando la máquina virtual mediante VirtualBox Oracle, primero asegúrese de que el usuario sea miembro de las usbusers grupo.

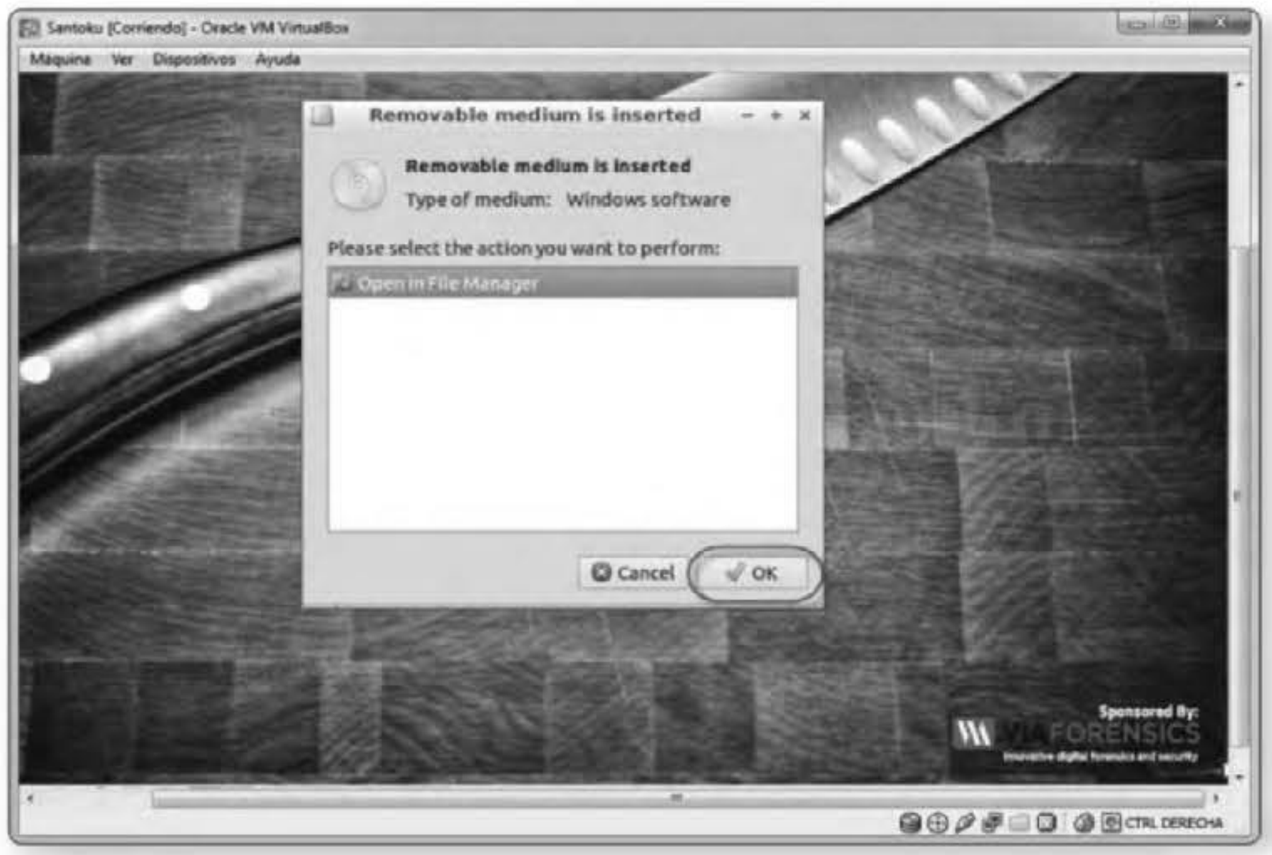
Por lo tanto, desde una sesión de terminal, ejecute lo siguiente:



Paso 19: Instale.



Paso 20: Presione el botón OK.



Paso 21: Presione nuevamente el botón OK.



Paso 22: Haga doble clic en autorun.sh y elija *Execute in Terminal*.

Paso 23: Presione la tecla F4.

13.7.4. Comando wget

El descargador de archivos wget es considerado como el descargador (downloader) más potente que existe en el mercado; ya que soporta HTTP, HTTPS y FTP (ver fig.).



Comenzará la descarga:



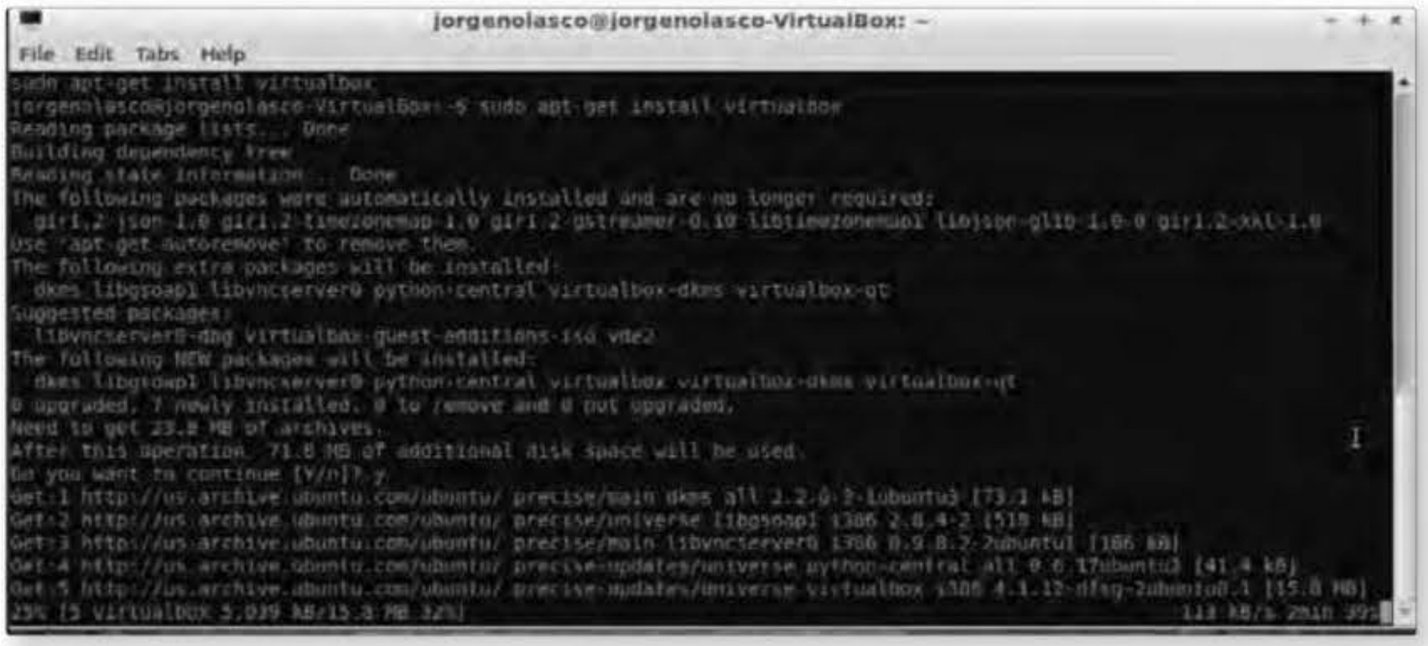
Finalizando la descarga:



Empiece a instalar con el siguiente comando:



A la espera de que concluya:



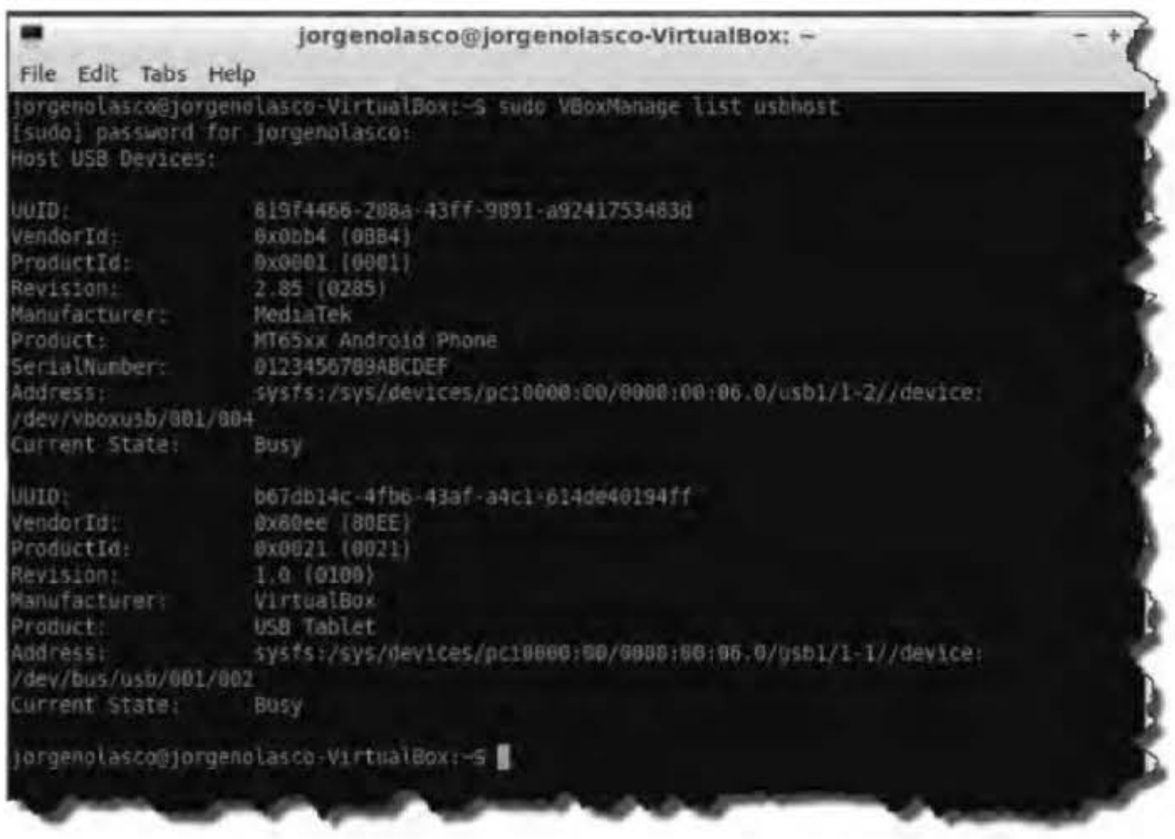
Instalación completada:



13.7.5. Comando VBoxManage list

El comando list proporciona información relevante acerca del sistema; para ello, suministra información sobre los dispositivos USB conectados al host, en particular información útil relacionada a la construcción de filtros USB y si se encuentran actualmente en uso por el anfitrión.

A. Sudo VBoxManage list usbhost



> ***Nota:***
>
> *No olvide conectar su dispositivo móvil.*

B. Comando VBoxManage usbfilter

Coloque un filtro por defectos en el dispositivo móvil, mediante el comando:

```
VBoxManage usbfilter add <index,0-N>
-target <uuid>|<name>|global
-name <string>
-action ignore|hold (global filters only)
[-active yes|no] (yes)
[-vendorid <XXXX>] (null)
[-productid <XXXX>] (null)
[-revision <IIFF>] (null)
[-manufacturer <string>] (null)
[-product <string>] (null)
[-remote yes|no] (null, VM filters only)
[-serialnumber <string>] (null)
[-maskedinterfaces <XXXXXXXX>]
```

Al identificar los parámetros anteriormente descritos:

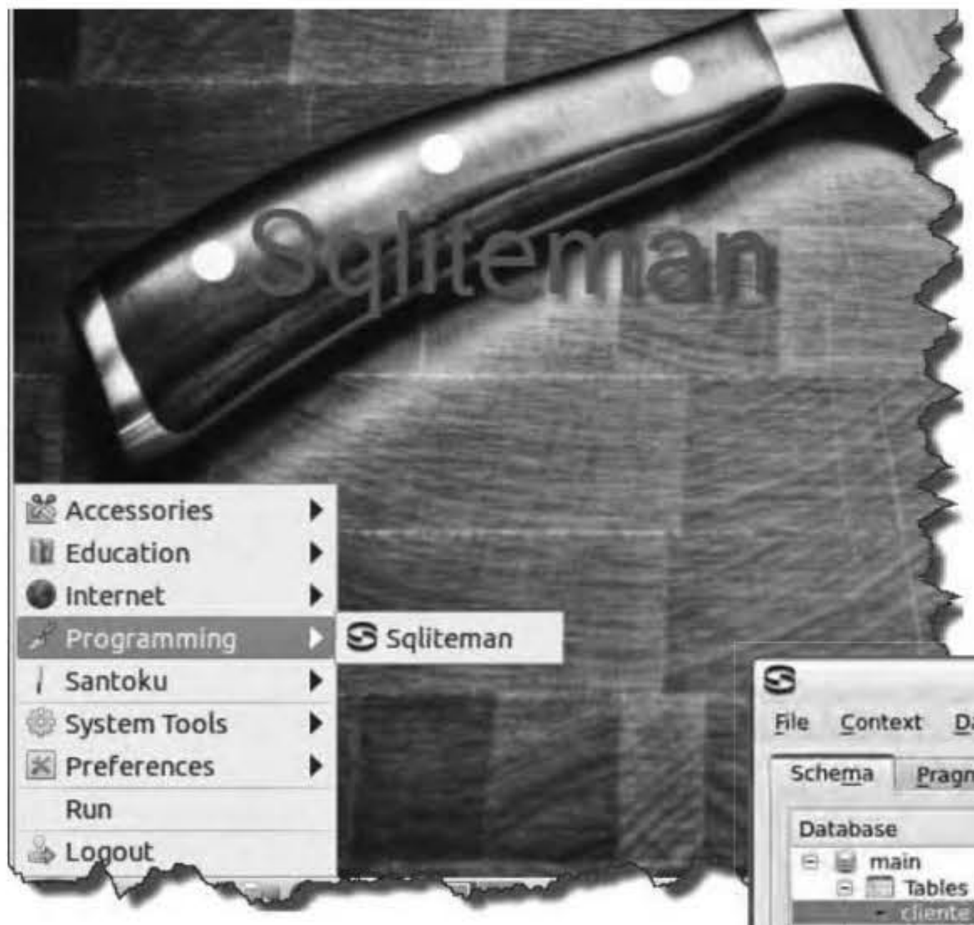
```
jorgenolasco@jorgenolasco-VirtualBox: ~
File  Edit  Tabs  Help
jorgenolasco@jorgenolasco-VirtualBox:~$ sudo VBoxManage list usbhost
[sudo] password for jorgenolasco:
Host USB Devices:

UUID:               819f4466-208a-43ff-9091-a9241753463d
VendorId:           0x0bb4 (0BB4)
ProductId:          0x0001 (0001)
Revision:           2.85 (0285)
Manufacturer:       MediaTek
Product:            MT65xx Android Phone
SerialNumber:       0123456789ABCDEF
Address:            sysfs:/sys/devices/pci0000:00/0000:00:06.0/usb1/1-2//device:
/dev/vboxusb/001/004
Current State:      Busy

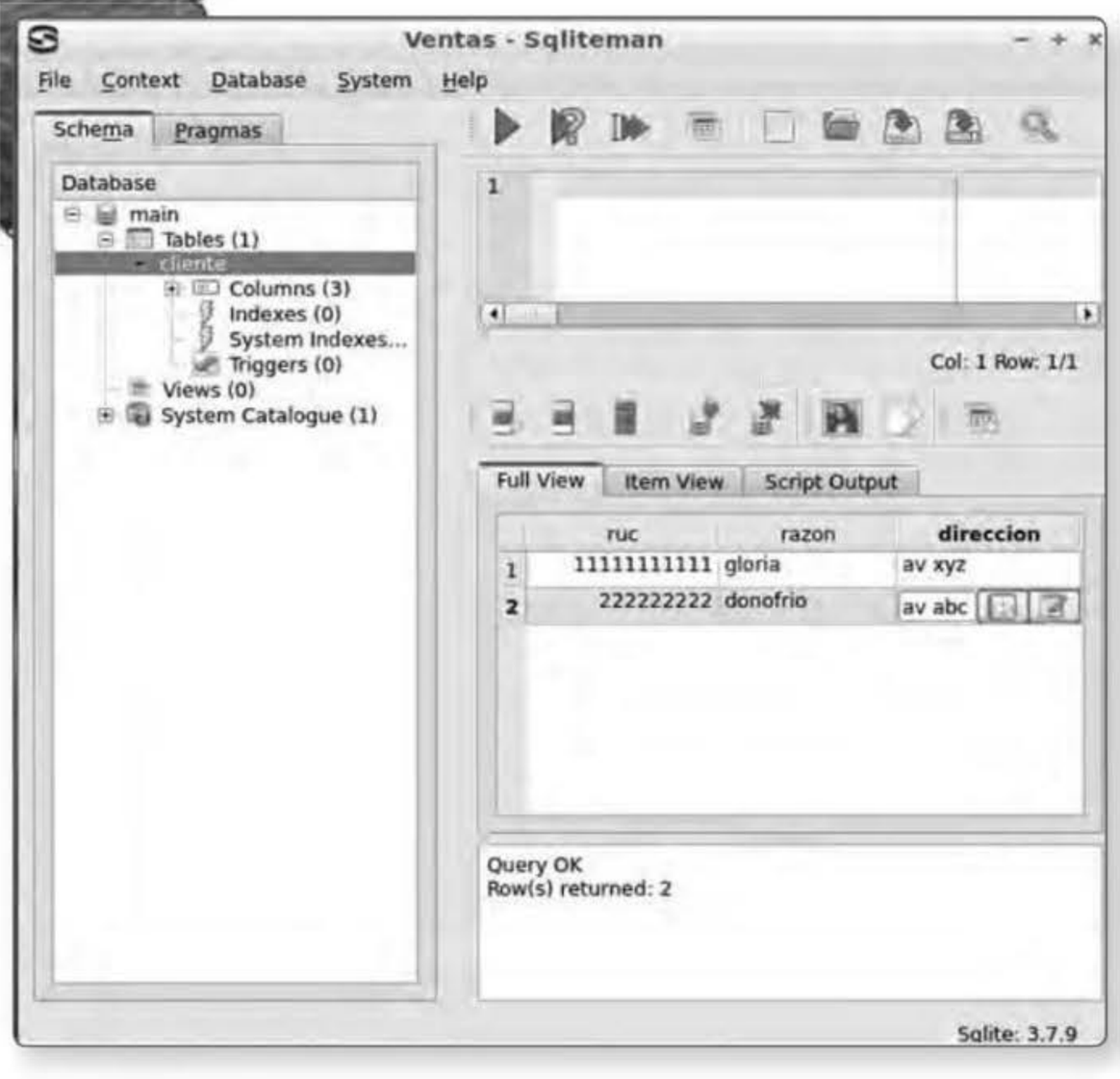
UUID:               b67db14c-4fb6-43af-a4c1-614de40194ff
VendorId:           0x80ee (80EE)
ProductId:          0x0021 (0021)
Revision:           1.0 (0100)
Manufacturer:       VirtualBox
Product:            USB Tablet
Address:            sysfs:/sys/devices/pci0000:00/0000:00:06.0/usb1/1-1//device:
/dev/bus/usb/001/002
Current State:      Busy

jorgenolasco@jorgenolasco-VirtualBox:~$
```

13.8. SQLITEMAN



Sqliteman es una herramienta útil diseñada para ofrecer a los desarrolladores una herramienta completa de GUI del desarrollador o del administrador para Sqlite3; por lo que el usuario ideal es un desarrollador de software, principalmente. Contiene uno de los más completos conjuntos de características de las herramientas disponible; además utiliza bases de datos Sqlite3 en su código. En la siguiente figura, se observa una interfaz amigable para desarrolladores:



13.9. HERRAMIENTAS DE DESARROLLO

13.9.1. Android SDK Manager

Android SDK Manager es el Kit de desarrollo de software Android y el emulador.

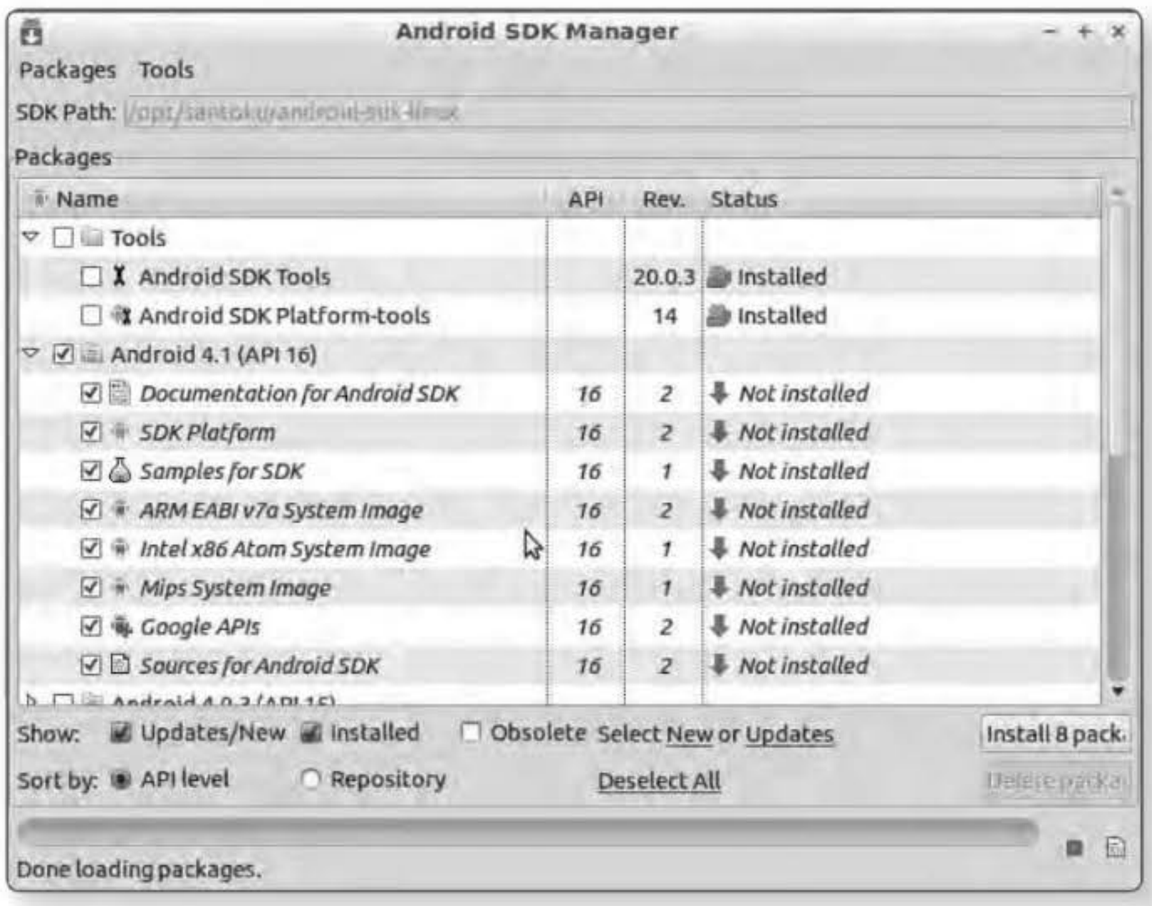
El emulador puede ser una herramienta valiosa para el analista, ya que permite el pleno acceso al sistema de archivos, configuraciones de hardware, características de funcionamiento, y más de cada versión del sistema operativo Android.



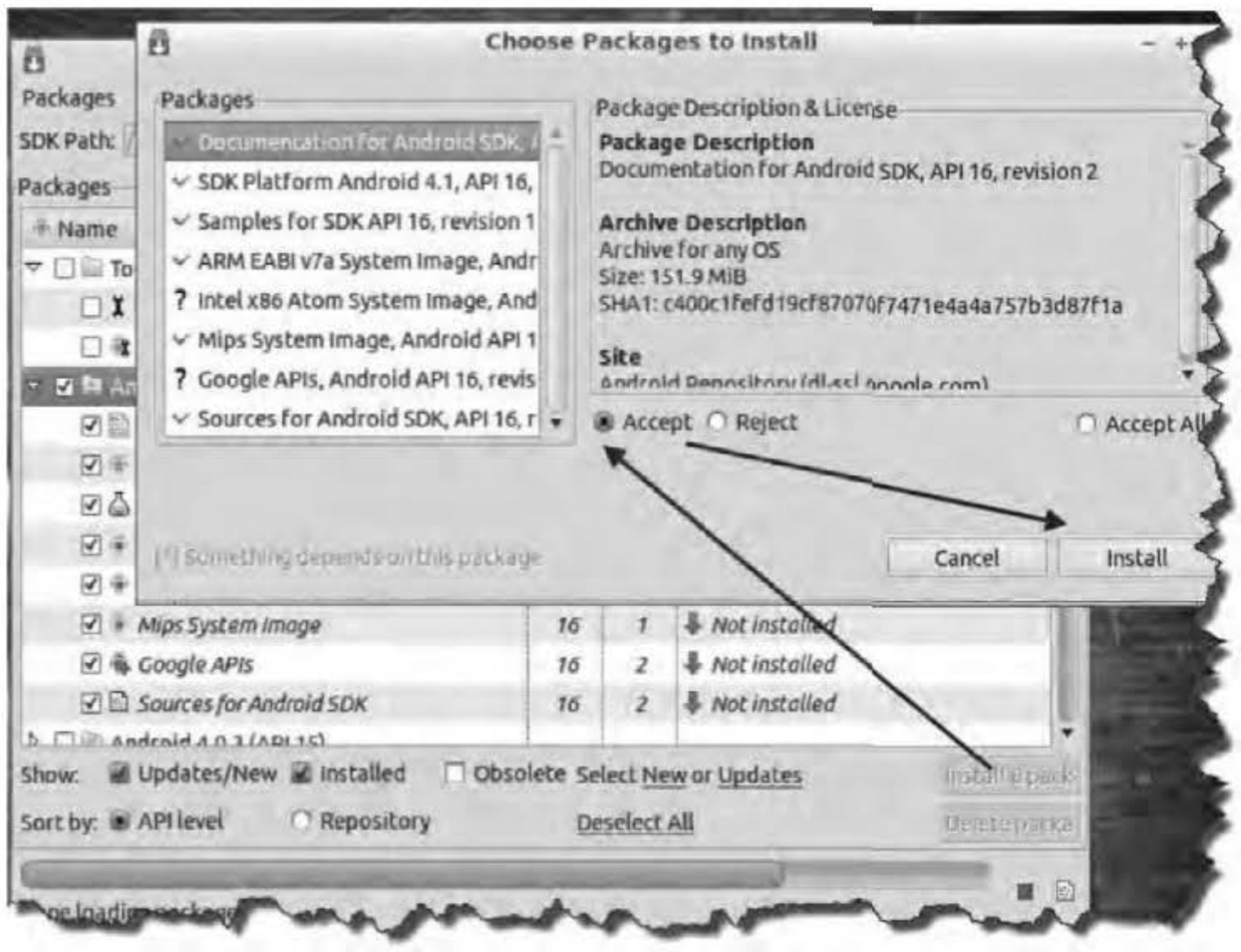
Nota

El emulador del SDK de Android requiere una gran cantidad de recursos, por lo que se recomienda proporcionar a la máquina virtual con memoria la cantidad de procesadores como sea posible.

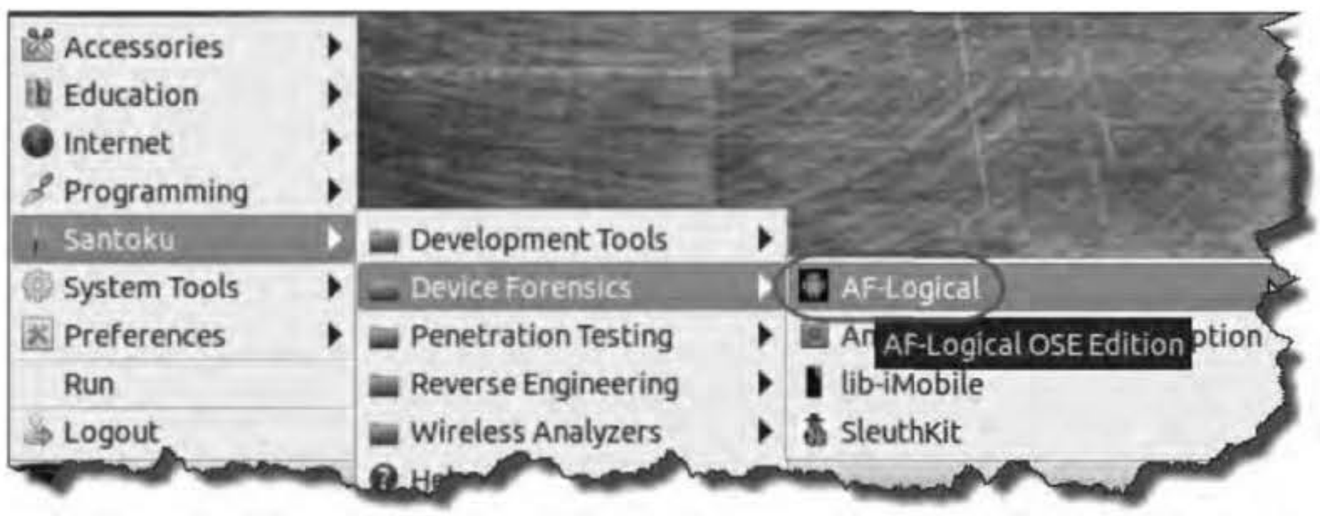
Paso 1: Para este ejemplo, se eligió instalar con la versión 4.1 API 16 luego presione el botón *Install*.



Paso 2: Presione el botón Install (o Instalar).



13.9.2. AFLogical OSE

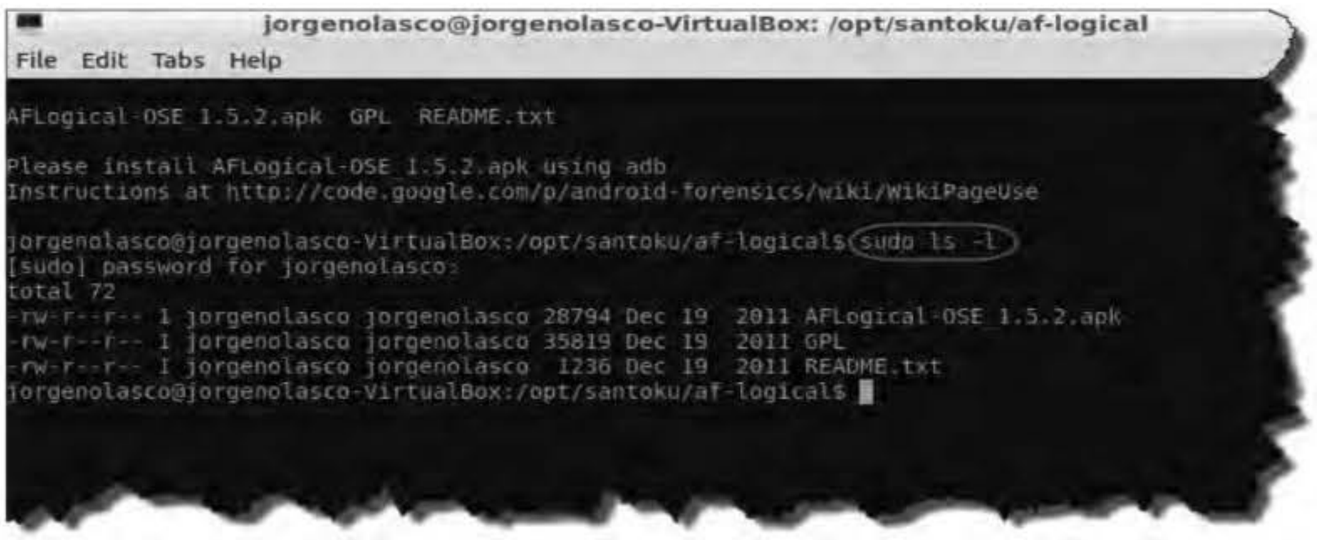


Active *USB debugging* en su dispositivo:

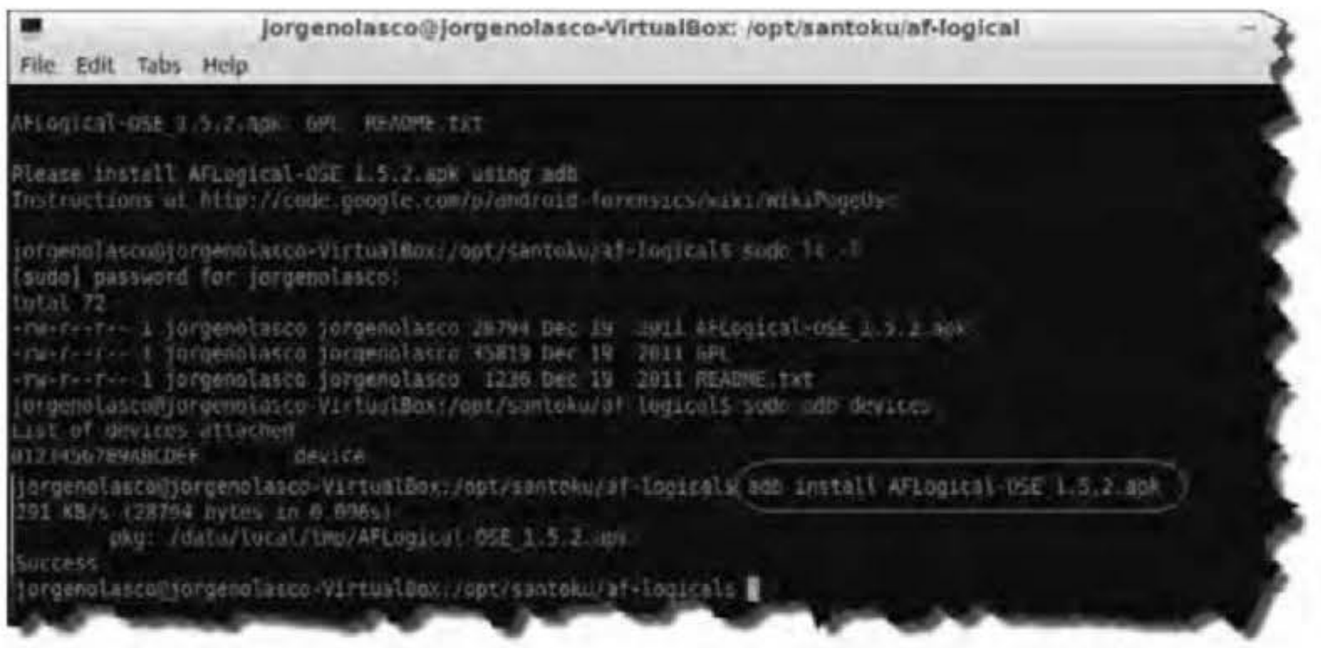
- Para Android 3.x (o versiones anteriores), diríjase a *Settings > Applications > Development* y active *USB debugging.*
- Para Android 4.x (o versiones posteriores), diríjase a *Settings > Developer Options* y active con un check en *USB debugging*.



Al extraer la información del dispositivo:



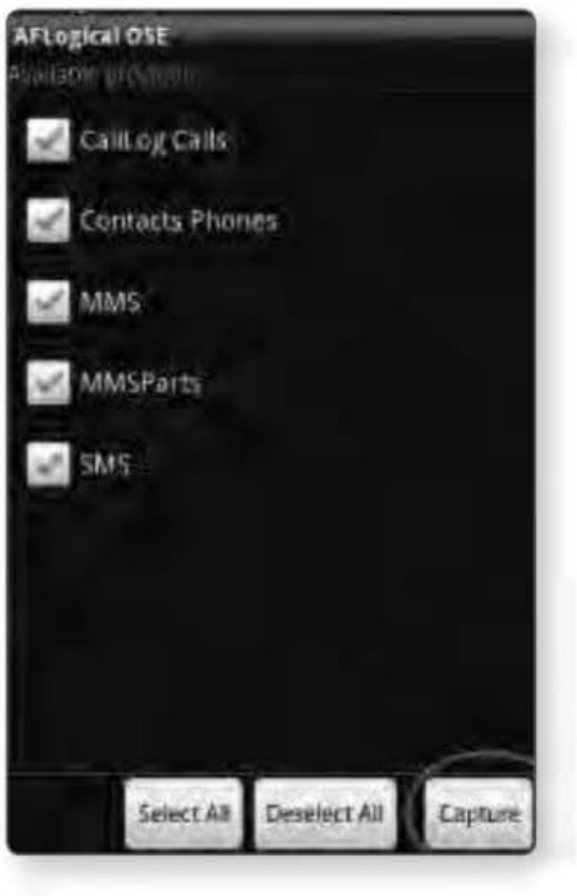




13.10. CARACTERÍSTICAS DE UN DISPOSITIVO MÓVIL

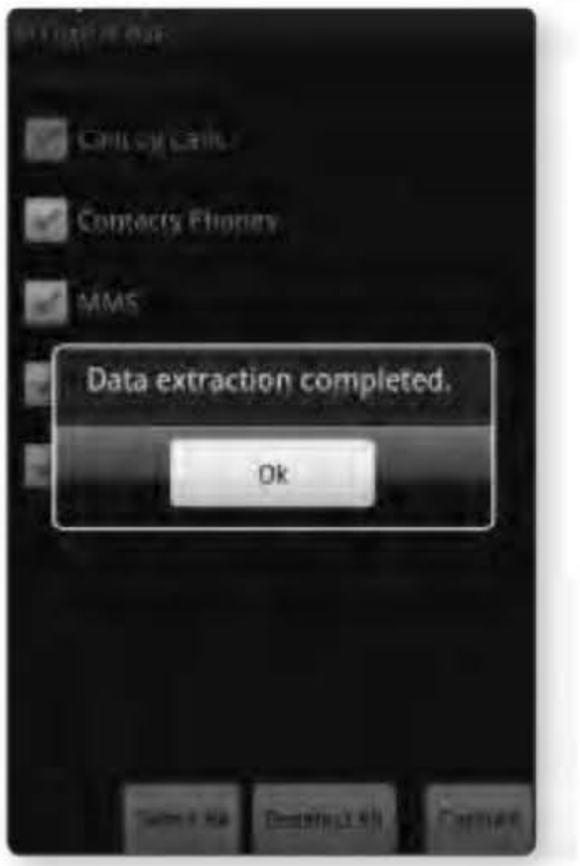


Paso 1: Ingrese a la aplicacion AFLogical OSE en su dispositivo Android.



Paso 2: Seleccione la información que desea extraer y presione el botón Capture.

Paso 3: Presione el botón OK.



13.11. Estrategia de Google

La misión de Google es organizar la información mundial y hacerla accesible y útil. Los teléfonos celulares son el dispositivo de consumo más popular, con más de 4 mil millones, Google cree que está cumpliendo con la parte de acceso universal de su misión. Pero todavía tiene que existir algún beneficio para Google. Cuando hay más personas en línea, más personas utilizan la búsqueda, lo que conduce en última instancia la principal fuente de ingresos de anuncios de Google. En una entrevista de marzo 2009, Andy Rubin explica:

> *Google tiene un gran modelo de negocio alrededor de la publicidad, y hay una conexión natural entre el código abierto y el modelo de negocio de publicidad. El código abierto es básicamente una estrategia de distribución, es eliminar por completo la barrera de entrada para la adopción.*

Una de las críticas de Android es que el mercado está fragmentado con diferentes versiones y variaciones de Android, un resultado directo de cómo Google lanza Android a los fabricantes. Esto está en contraste con otros dispositivos, como el iPhone que Apple tiene un control total sobre el hardware y el sistema operativo y una influencia significativa en una aplicación de terceros. A pesar de que Rubin defiende este modelo, mencionó en la misma entrevista lo siguiente:

> *El control de todo el dispositivo es grande, (pero) estamos hablando de alrededor de 4 mil millones de aparatos. Cuando el control de todo el dispositivo la capacidad de innovar con rapidez es bastante limitada en lo que viene de un solo proveedor. Usted puede tener arranques de la innovación. Usted puede clavar la empresa, las uñas de ciertas técnicas de interfaz o usted puede clavar la Web en el teléfono-negocio, pero usted no puede hacerlo todo. Siempre vas a estar en algún lugar. Lo que estamos hablando es de salir de un nicho de mercado y dar a las personas el acceso a Internet en la forma en que esperan que el Internet para acceder a ellos. No quiero crear algún derivado de la Internet, no quiero tomar solo un trozo de la Internet, no quiero estar en* la esquina en algún lugar con alguna de Internet versión simplificada del quiero estar en Internet.

Por lo tanto mediante la creación de un sistema operativo móvil que satisface las demandas de los consumidores, así como las necesidades de los fabricantes y proveedores de servicios inalámbricos, Google dispone de una plataforma excelente distribución de los ingresos de generación de negocio de búsqueda y publicidad.

13.11.1. Aplicaciones

Es una forma importante por la cual Android apoya la innovación, permitiendo el desarrollo y la distribución de aplicaciones de terceros en Android.

A partir de enero de 2011, más de 200 000 aplicaciones de Android se han desarrollado. Esto, por supuesto, es similar a la estrategia de Apple. Sin embargo, existen diferencias fundamentales en su enfoque. Apple mantiene un estricto control sobre su App Store, exigiendo algunas pruebas largas donde Apple poseería la decisión final.

Google, por el contrario, requiere una revisión limitada para publicar una aplicación en el Android Market. Mientras que Google tiene la capacidad de prohibir a un desarrollador, eliminar una aplicación del Android Market y remotamente desinstalar aplicaciones de los dispositivos Android.

Droid Dream Light: un nuevo virus infecta 25 'apps' de Android

Entre 30.000 y 120.000 usuarios de Android pueden haberse visto afectados por una nueva variante de un 'malware' anterior, conocido como Droid Dream.

Según Lookout Security, este 'software' malicioso ha infectado a 25 aplicaciones. Todas ellas ya han sido eliminadas por Google de sumercado de 'apps' oficial.

El nuevo 'malware' para Android se llama Droid Dream Light. Es una versión modificada de Droid Dream y proviene de los creadores del virus que recientemente infectó a 50 aplicaciones. En este caso se han identificado 25 'apps' y entre 30.000 y 120.000 usuarios podrían haberse visto afectados.

Sus componentes maliciosos aparecen cuando se activa un movimiento "android.intent.action. PHONE_STATE intent", como puede ser una llamada entrante. Esto quiere decir que este 'malware' no depende de la ejecución manual de la aplicación infectada para activarse. Google ya ha eliminado todas las 'apps' infectadas, aunque los que ya están afectados requieren otra solución, como el borrado remoto.

Según se desprende del blog de la firma Lookuot Security, pese a la designación de 'Light' este 'software' es incluso más malicioso que su predecesor, ya que no requiere ninguna acción por parte del usuario para que el código se ejecute.

Lookout ha identificado el 'malware' gracias a una notificación de un desarrollador que informó de unas versiones modificadas de su aplicación que estaban siendo distribuidas en el Android Market.

ReadWriteWeb.es |

Fuente: www.elmundo.es

13.11.2. Seguridad en Android: Reactivo-proactivo

Cuando una aplicación se instala por primera vez, Android comprueba el archivo. Apk para asegurarse de que tiene una firma digital válida para identificar el desarrollador. A diferencia de SSL, las aplicaciones Android no tienen que ser firmado por una Autoridad de Certificación. Sin embargo, el desarrollador debe tener la llave de seguridad; de lo contrario, alguien podría firmar una aplicación maliciosa y distribuirlo como el desarrollador.

Por ejemplo, si la firma digital de una entidad financiera se ha visto comprometida, un desarrollador malintencionado podría publicar una actualización de la aplicación de banca, que posibilite robar los datos críticos.

Google luchará de manera proactiva contra el malware del Android Market

Uno de los puntos más grises de Android es el de la seguridad de las aplicaciones del Android Market (y los otros markets de aplicaciones) y la presencia de malware. Si bien es cierto que el usuario es una de las mejores barreras para esquivar este tipo de problemas de seguridad, por ejemplo, no instalando aplicaciones "sospechosas" porque, además, hasta ahora Google solamente actuaba de manera reactiva y retiraba las aplicaciones si éstas presentaban problemas pero no verificaba la ausencia de malware antes de ponerlas a disposición de los usuarios. Al final, Google ha decidido intervenir y se sumará de manera más proactiva a la lucha contra el malware en el Android Market y escaneará las aplicaciones en busca de código malicioso.

El anuncio lo ha realizado Hiroshi Lockheimer, vicepresidente de ingeniería del equipo de Android en una entrevista en CNET en la que presentó este servicio, de nombre Bouncer, que escaneará las aplicaciones en busca de malware conocido, spyware y troyanos; una tarea que realizará buscando comportamientos sospechosos y comparando el código de la aplicación con el de versiones previas. Todas las aplicaciones se ejecutarán en un entorno virtual para simular el funcionamiento de la aplicación sobre un dispositivo Android, algo que también se realizará con las aplicaciones ya publicadas en el market y no únicamente con las nuevas.

> Este sistema toma una aplicación que se ha subido y la ejecuta en una infraestructura en la nube que monitoriza lo que hace la aplicación en un entorno virtual

Si el sistema cataloga una aplicación como sospechosa, se marca para una confirmación manual de que se trata de malware, en cuyo caso se bloquearía o se eliminaría directamente. La verdad es que se echaba en falta un control así en el Android Market y, bajo mi punto de vista, esta apuesta además de ser algo demandado por los usuarios va a servir para despejar cualquier duda sobre la seguridad de la plataforma. Además, Google analizará las cuentas de los nuevos desarrolladores para prevenir "malas prácticas" y controlar casos reincidentes.

Según han comentado, Bouncer lleva ya varios meses funcionando en silencio y en el segundo semestre de 2011 las descargas de aplicaciones maliciosas han descendido en un 40%, según comenta Google, un dato más que interesante y que, teóricamente, debería seguir creciendo y limpiando el market de aplicaciones potencialmente peligrosas.

Fuente: www.celularis.com

13.11.3. Tecnicas de seguridad

Investigaciones internas en empresas encuentran los problemas más comunes:
El robo de propiedad intelectual o datos.

- El uso inapropiado de los recursos de la empresa.
- Intento de ataque o de éxito contra los sistemas informáticos.
- Las investigaciones relacionadas con el empleo como la discriminación, el acoso sexual, etc.
- Auditoría de seguridad (aleatorio o específico).
- Análisis forense en los casos de asuntos de familia. Los casos más comunes incluyen los siguientes:
  - Divorcio
  - Custodia de los hijos
  - Los conflictos de bienes

Una última área donde la investigación forense puede generar un importante valor es para la seguridad y el funcionamiento de un gobierno.

13.12. COMPELSON MOBILEDIT!

MOBILedit! Forense recoge todos los datos posibles de los teléfonos móviles y genera informes detallados en una PC que se pueden guardar o imprimir. Es la solución más universal de teléfonos móviles con el software de apoyo a la mayoría de teléfonos GSM y una arquitectura abierta que permite el apoyo de cualquier teléfono.

MOBILedit! Forense hace un análisis completo del teléfono, incluida su libreta de teléfonos, los últimos números marcados, llamadas perdidas, llamadas recibidas, mensajes MMS, mensajes SMS, fotos, videos, archivos, detalles del teléfono, calendario, notas, tareas, etc.

13.12.1. Uso del MOBILedit!

Paso 1: Conecte el dispositivo con Android a la estación de trabajo forense a través de USB.

Paso 2: Haga clic en *Start > Connect your phone*.



Paso 3: Seleccione *Phone connected to PC > Next*.



Paso 4: Seleccione *Cable (Usb of serial) connection > Next.*







13.12.2. Pantalla principal

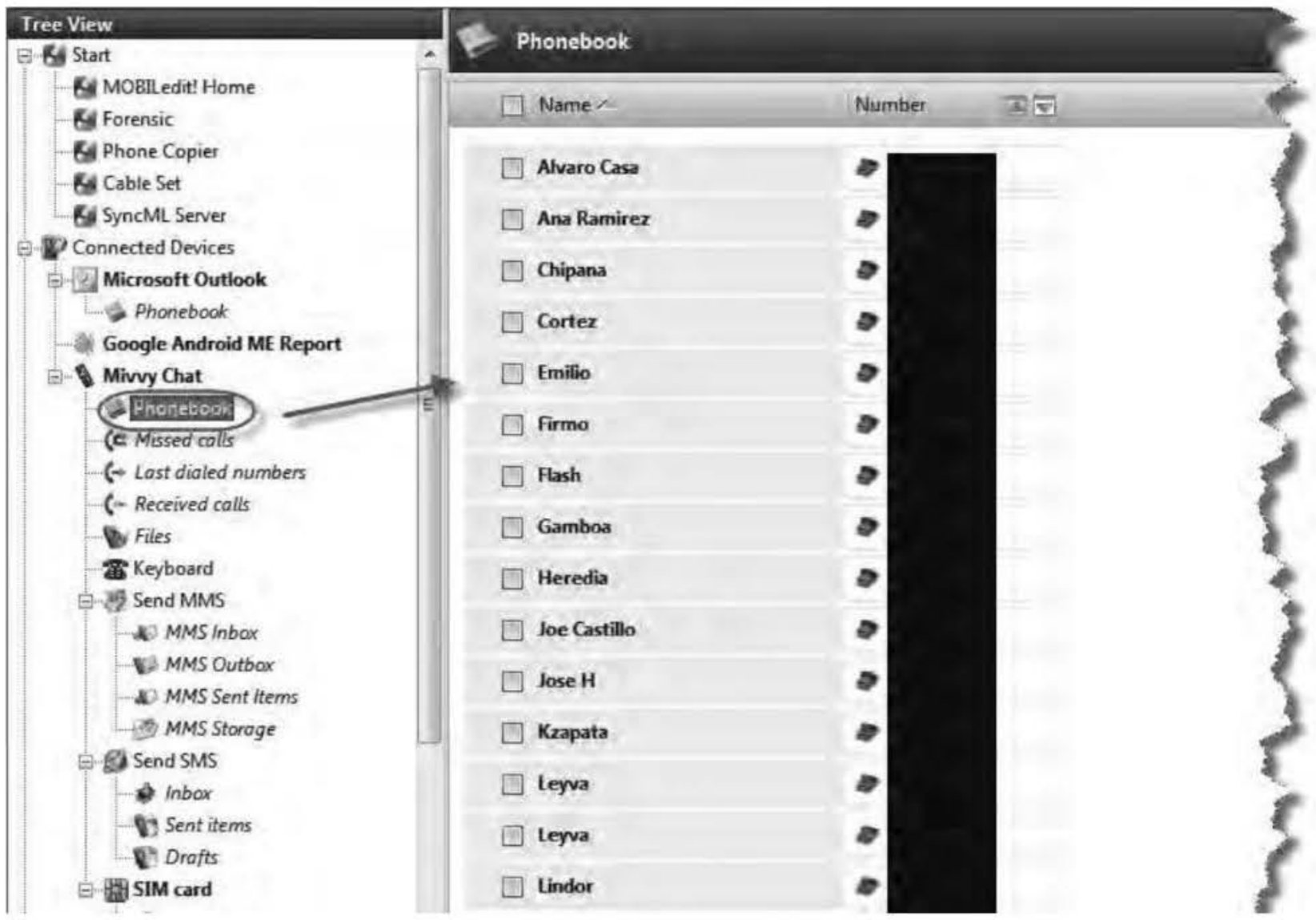
MOBILedit muestra las estadísticas de los dispositivos que fueron detectados, así como una vista de los datos de las aplicaciones disponibles para el análisis.
Se muestra la pantalla principal lo siguiente:

A. Información general

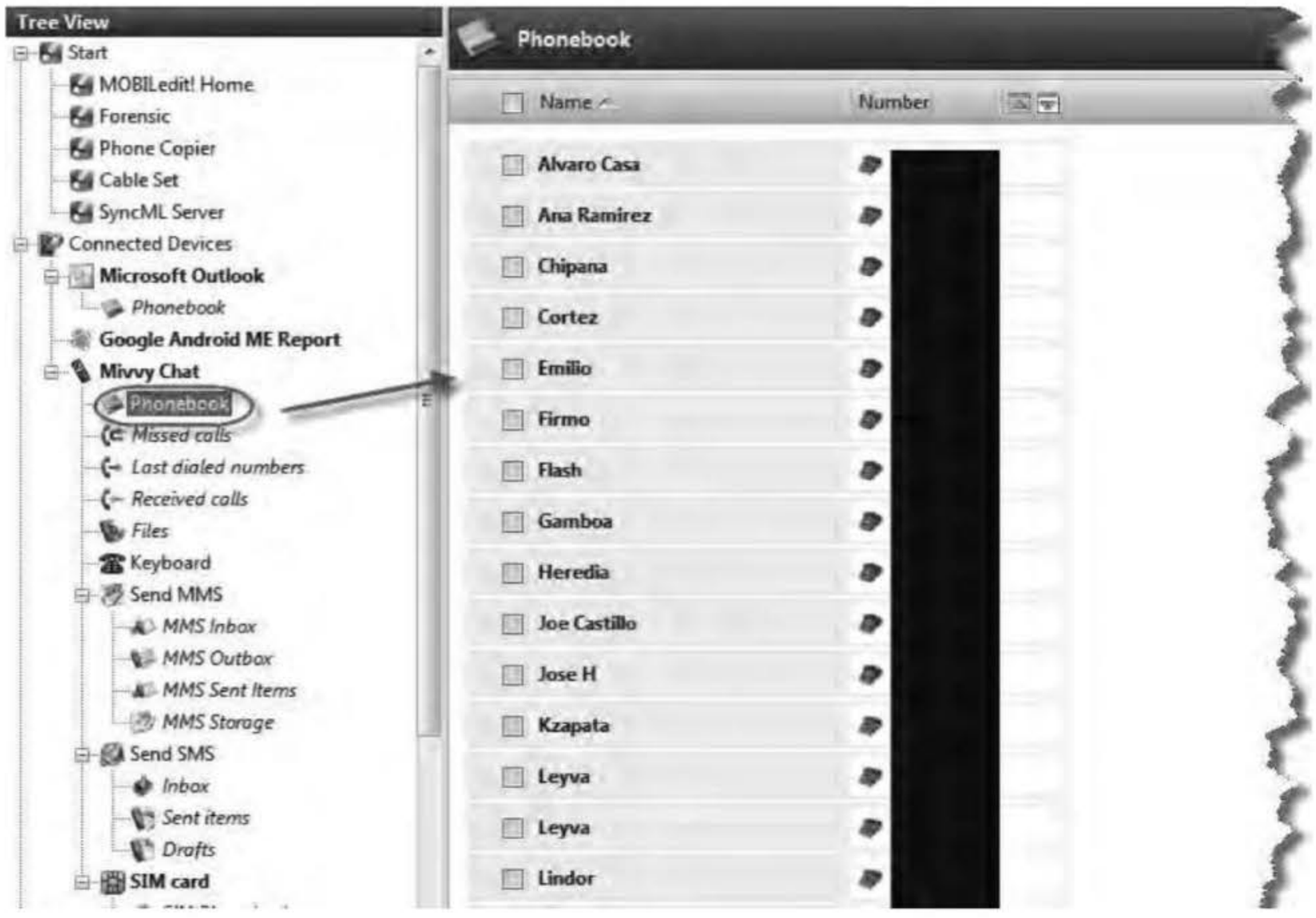
Donde puede ver información específica del dispositivo, incluyendo el número de IMEI, número de serie y detalles sobre la cantidad de memoria del teléfono, la señal de la batería, señal de la red.



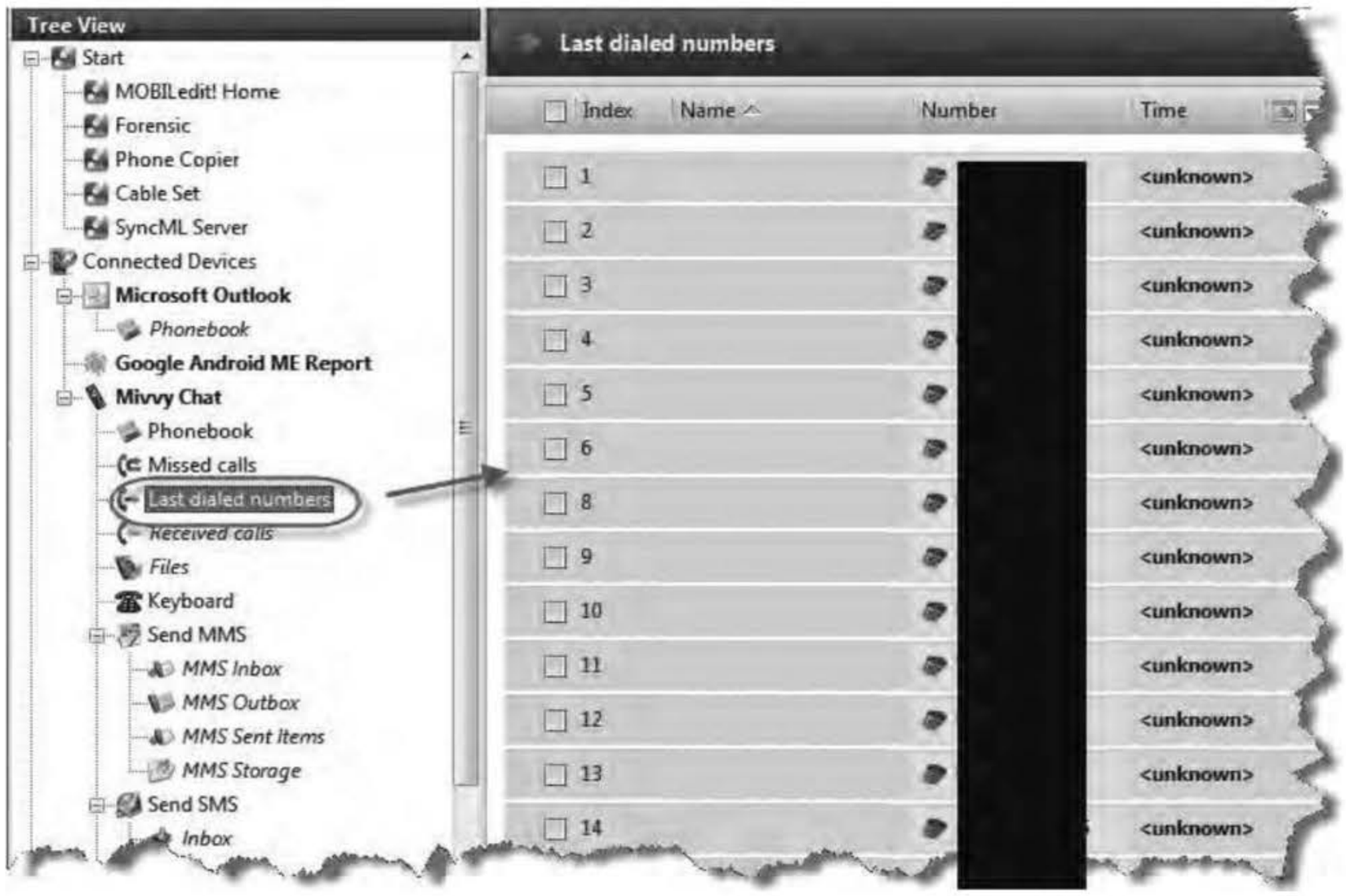
B. Los contactos



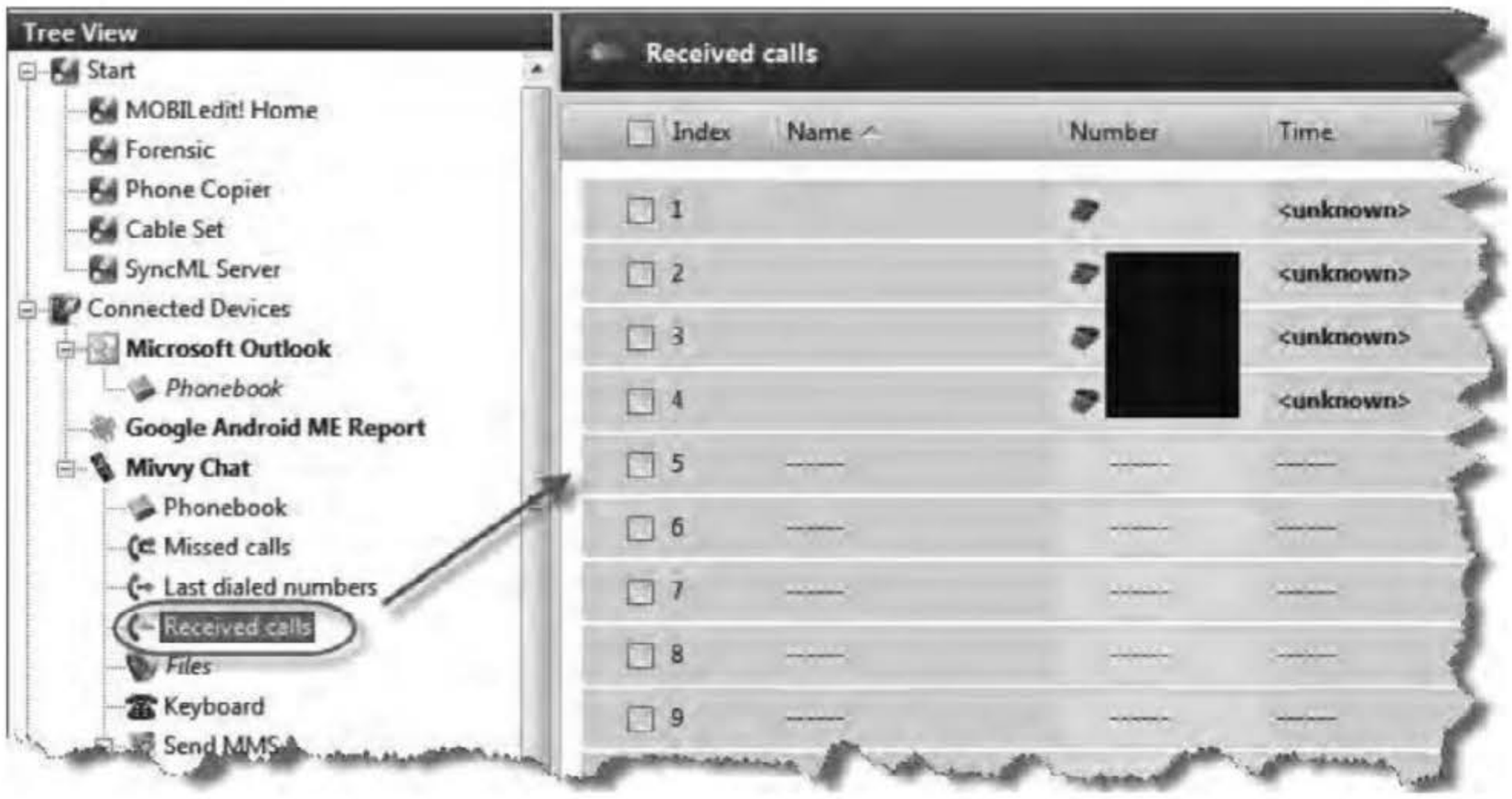
C. Llamadas perdidas



D. Últimos números marcados



E. Llamadas recibidas



13.13. FILE EXPERT



Descargar File Expert v4.2.4 (APK)

File Expert es un explorador de archivos para usuarios ROOT que te permite realizar todas las operaciones básicas con archivos y carpetas como copiar, pegar, mover, crear, renombrar y cambiar permisos.

Ademas, te permite convertir tu dispositivo en un servidor de archivos FTP/HTTP, de modo que puedas compartir y administrar tus archivos sin cables USB utilizando clientes como Web Sharing o WIFI File Explorer.

File Expert también funciona como cliente SMB, SFTP, FTP, FTPS, lo que te permite acceder a una PC anfitrión desde tu dispositivo para poder gestionarla mediante una WIFI en los protocolos SMB, SFTP, FTP, FTPS.

Fuente: androidzone.org

Gestión de archivos completa y con opciones avanzadas

File Expert es una completa herramienta para organizar tus archivos y aplicaciones y para acceder al PC o incluso a servidores FTP.

Dirigido a usuarios que requieran opciones de gestión avanzada, File Expert destaca por posibilidades como el acceso a archivos de PC mediante SMB y la conexión WiFi, con la que podrás ver tus fotos o escuchar música almacenada en el ordenador. También permite la subida de archivos usando FTP. File Expert cuenta con opciones más básicas como copiar, renombrar, mover o descomprimir archivos. A ello se suma una práctica utilidad de copias de seguridad para las aplicaciones que tengas instaladas en tu Android.

Visor de imágenes, búsqueda de archivos en la tarjeta de memoria y muchas más opciones convierten a File Expert en uno de los mejores gestores de archivos avanzados para Android.

Fuente: file-expert.softonic.com

CAP.

14

Juegos



14.1. ASPECTOS GENERALES

Canvas

Canvas (lienzo) es una interfaz que permite pintar en un área rectangular, que refleja las acciones sobre un mapa de bits (Bitmap) asociado:

```
Bitmap b = Bitmap.createBitmap
    (100, 100, Bitmap.Config.ARGB_8888);
Canvas c = new Canvas(b);
```

Para escribir en la pantalla, no será necesario instanciar Canvas directamente.

Canvas-Métodos

Principales métodos de Canvas:

***drawBitmap**(Bitmap bitmap, float left, float top, Paint paint). Dibuja un Bitmap con la esquina superior izquierda en la posicion indicada del lienzo.*

***drawBitmap**(Bitmap bitmap, Rect src, Rect dst, Paint paint). Escala/ajusta un Bitmap al area (sic) de destino.*

***drawCircle(**float cx, float cy, float radius, Paint paint). Dibuja un circulo.*

***drawColor**(int color). Llena el lienzo con un color.*

***drawline**(float startX, float startY, float stopX, float stopY, Paint paint). Dibuja una linea de (startX, startY) a (stopX, stopY).*

***drawRect**(float left, float top, float right, float bottom, Paint paint). Dibuja el rectangulo especificado.*

***drawText**(String text, float x, float y, Paint paint). Dibuja texto en (x, y) con el Paint especificado.*

***drawLine**(float startX, float startY, float stopX, float stopY, Paint paint). Dibuja una linea de (startX, startY) a (stopX, stopY).*

***drawRect(**float left, float top, float right, float bottom, Paint paint). Dibuja el rectangulo especificado.*

***drawText**(String text, float x, float y, Paint paint) Dibuja texto en (x, y) con el Paint especificado.*

*[**drawARGB/drawRGB/drawColor**. Llena el canvas con un solo color.].*

Rafael Morón Abad

Fuente: recursos.crfptic.es:9080

14.2. APLICACIÓN DE CANVAS: LÍNEAS, RECTÁNGULO Y CÍRCULOS

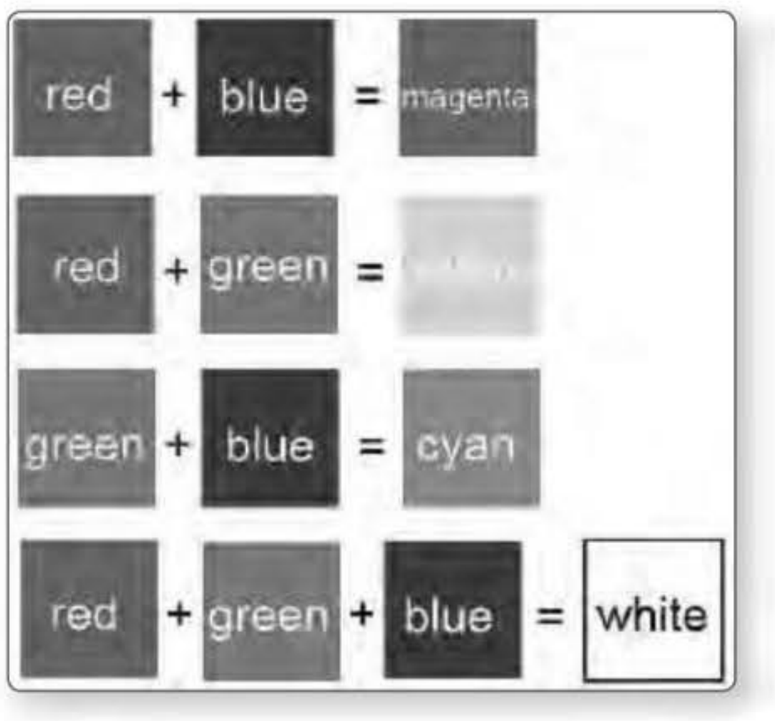
Cree una aplicación que permita demostrar el uso de canvas dibujando algunas figuras.

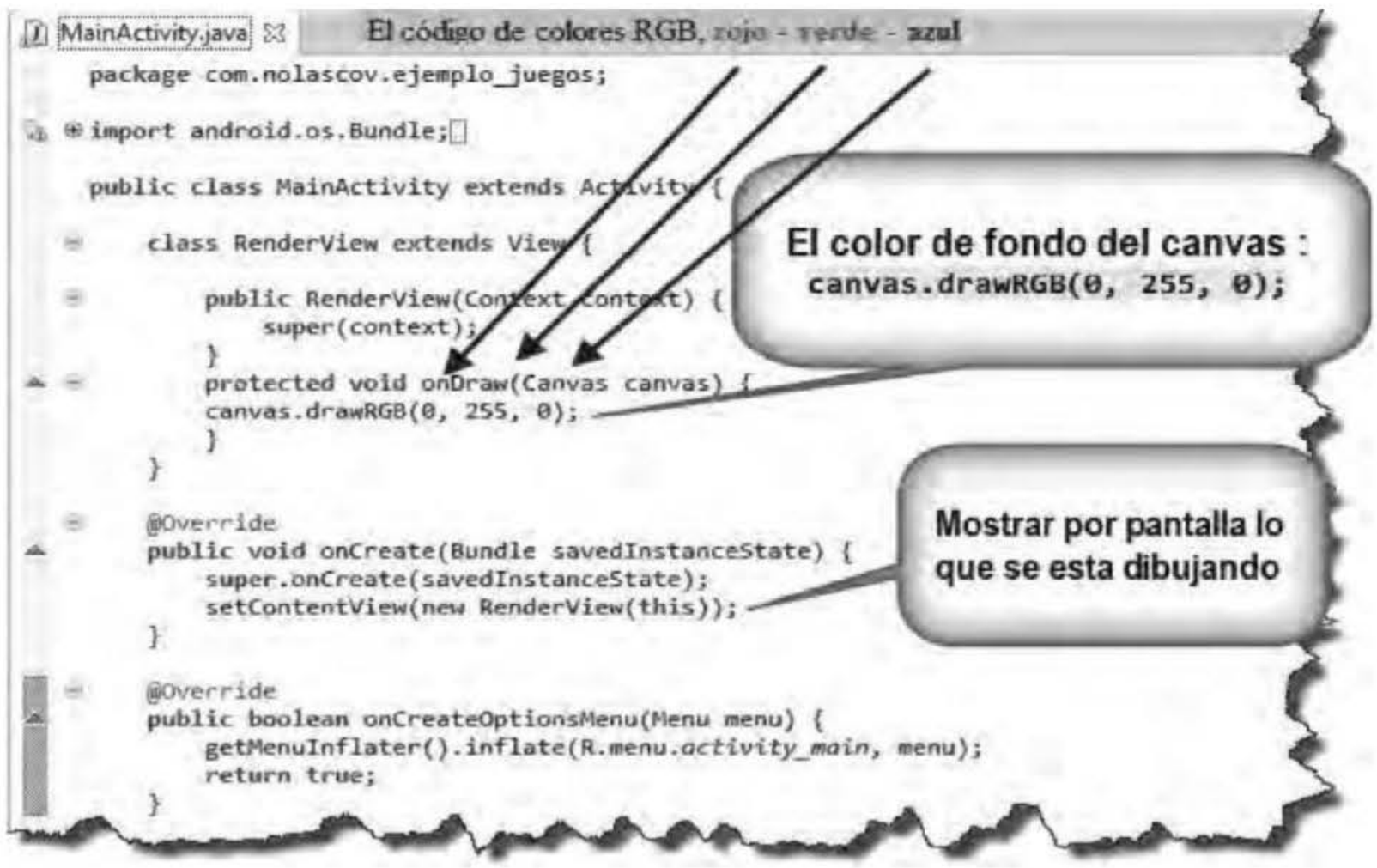
Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto*.

Paso 2: Ingrese el nombre del proyecto: Ejemplo_juegos (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 3: Modifique la actividad principal.







EJECUTANDO LA APLICACIÓN

Paso 1: *Guarde los cambios realizados, haciendo clic en el ícono*



o presionando la combinación Ctrl+S.

Paso 2: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:*



Paso 3: *Haga clic y mueva el ícono del candado en la zona superior para tocar un círculo que aparece al mover el mouse. Esto abre el emulador.*

La siguiente figura muestra la ventana principal de la pantalla de Android 4.1.



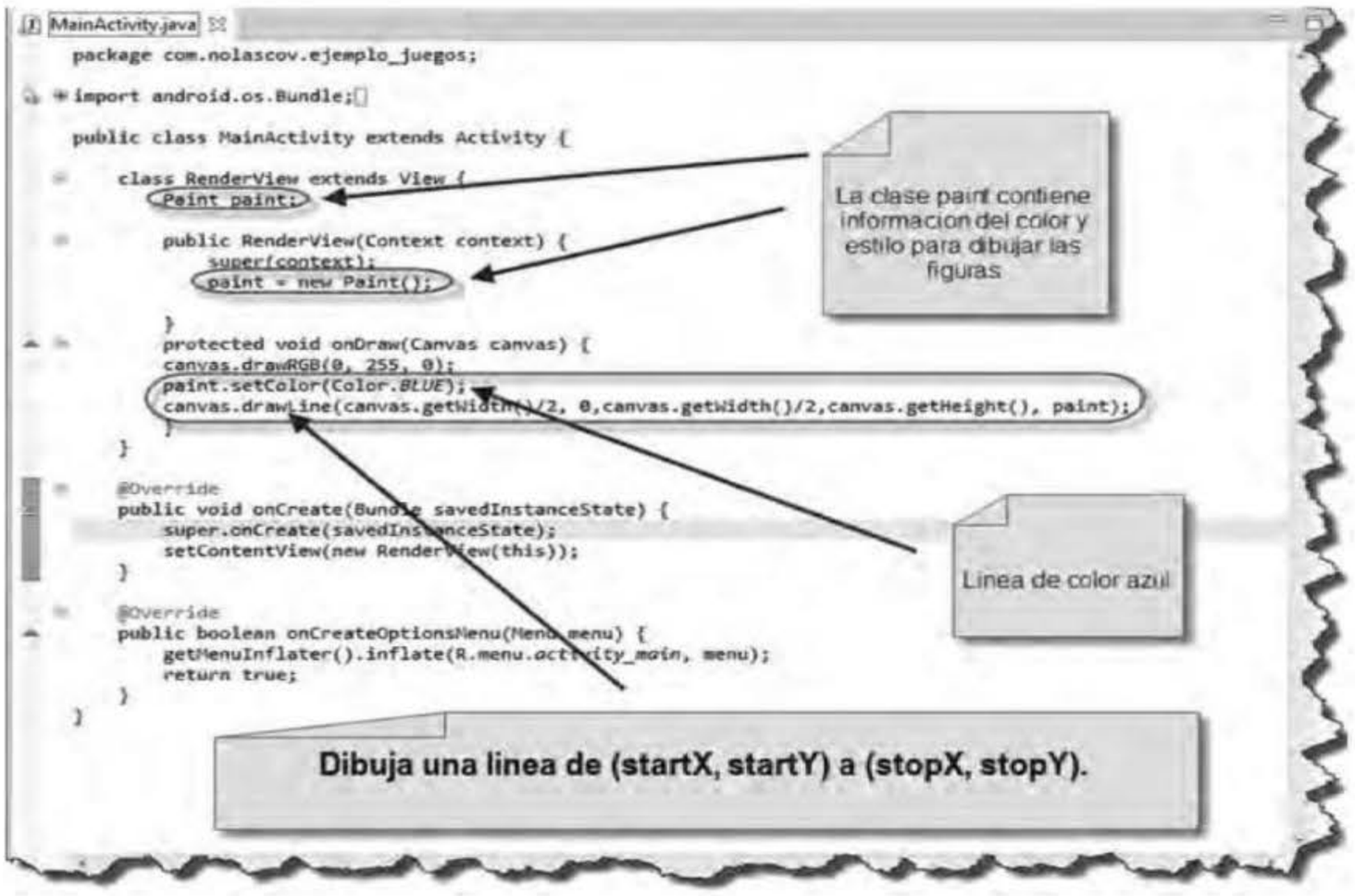
Paso 4: *Si el emulador se bloquea, deslice el botón de desbloqueo para contrarrestar su función.*

La figura muestra la aplicación que se ejecuta en el emulador de Android.



14.2.1. Dibujar una línea vertical

Para dibujar una línea vertical de color azul en medio de la pantalla, realice los siguientes pasos:





14.2.2. Dibujar rectángulos

Parat dibujar algunos rectángulos, realice los siguientes pasos:

Paso 1

```
*MainActivity.java
package com.nolascov.ejemplo_juegos;

import android.os.Bundle;

public class MainActivity extends Activity {

    class RenderView extends View {
        Paint paint;

        public RenderView(Context context) {
            super(context);
            paint = new Paint();

        }
        protected void onDraw(Canvas canvas) {
        canvas.drawRGB(0, 255, 0);
        int ancho=canvas.getWidth();
        paint.setColor(Color.BLUE);
        canvas.drawLine(canvas.getWidth()/2, 0,canvas.getWidth()/2,canvas.getHeight(), paint

        paint.setARGB(255,255,0,0);
        canvas.drawRect(10,20,ancho-10,50,paint);
```

Calculamos el ancho de la pantalla

Rectangulo de color Rojo

Dibujamos un rectángulo desde la coordenada columna:10 y fila 20 hasta la columna que coincide con el ancho de la pantalla menos 10 pixeles y la fila 50. Además le pasamos el pincel a utilizar



Paso 2:

```
MainActivity.java
package com.nolascov.ejemplo_juegos;

import android.os.Bundle;

public class MainActivity extends Activity {

    class RenderView extends View {
        Paint paint;
        public RenderView(Context context) {
            super(context);
            paint = new Paint();
        }
        protected void onDraw(Canvas canvas) {
        canvas.drawRGB(0, 255, 0);
        int ancho=canvas.getWidth();
        paint.setColor(Color.BLUE);
        canvas.drawLine(canvas.getWidth()/2, 0,canvas.getWidth()/2,canvas.getHeight(), pai

        paint.setARGB(255,255,0,0);
        canvas.drawRect(10,20,ancho-10,50,paint);

        paint.setStyle(Style.STROKE);
        canvas.drawRect(10,80,ancho-10,100,paint);
```

El siguiente rectángulo configuramos el pincel para que solo pinte el perimetro llamando al método setStyle y pasando la constante STROKE

Dibujamos un rectángulo desde la coordenada columna:10 y fila 80 hasta la columna que coincide con el ancho de la pantalla menos 10 pixeles y la fila 100. Además le



14.2.3. Dibujar un círculo

Para dibujar un círculo de color blanco, realice los siguientes pasos:

MainActivity.java

```
class RenderView extends View {
    Paint paint;
    public RenderView(Context context) {
        super(context);
        paint = new Paint();
    }
    protected void onDraw(Canvas canvas) {
    canvas.drawRGB(0, 255, 0);
    int ancho=canvas.getWidth();

    paint.setColor(Color.BLUE);
    canvas.drawLine(canvas.getWidth()/2, 0,canvas.getWidth()
                                                    t(), paint);

    paint.setARGB(255,255,0,0);
    canvas.drawRect(10,20,ancho-10,50,paint);

    paint.setStyle(Style.STROKE);
    canvas.drawRect(10,80,ancho-10,100,paint);

    paint.setColor(Color.WHITE);
    canvas.drawCircle(ancho/2, 220, 100,paint);

    }
}
```

circulo de color blanco

dibuja un circulo , los dos primeros argumentos especifican las coordenadas del punto central el segundo el radio



14.3. APLICACIÓN DE CANVAS: TEXTO

```
package com.nolascov.ejemplo_juegos;
import android.os.Bundle;
import android.app.Activity;
import android.content.Context;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Paint.Style;
import android.graphics.Path;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.support.v4.app.NavUtils;

public class MainActivity extends Activity {

    class RenderView extends View {
            Paint paint;
            public RenderView(Context context) {
                    super(context);
                    paint = new Paint();
            }
            protected void onDraw(Canvas canvas) {
            canvas.drawRGB(0, 255, 0);
            int ancho=canvas.getWidth();

            paint.setColor(Color.BLUE);
                canvas.drawLine(canvas.getWidth()/2,        0,canvas.getWidth()/2,canvas.
getHeight(), paint);

                paint.setARGB(255,255,0,0);
          canvas.drawRect(10,20,ancho-10,50,paint);

          paint.setStyle(Style.STROKE);
          canvas.drawRect(10,80,ancho-10,100,paint);

                paint.setColor(Color.WHITE);
          canvas.drawCircle(ancho/2, 220, 100,paint);

          paint.setColor(Color.RED);
          paint.setTextSize(38);
          paint.setShadowLayer(5, 0, 5, Color.BLACK);
          canvas.drawText("Este es un Texto", 60, 400, paint);

                }
        }

        @Override
        public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(new RenderView(this));
        }

        @Override
        public boolean onCreateOptionsMenu(Menu menu) {
            getMenuInflater().inflate(R.menu.activity_main, menu);
            return true;
        }
    }
```

EJECUTANDO LA APLICACIÓN



14.4. APLICACIÓN DE FUENTES EXTERNAS: TEXTO

Para confeccionar aplicaciones e incorporar fuentes True Type externas, implemente una aplicación que muestre un texto con una fuente externa.
Para crear una aplicación que permita demostrar el uso de fuentes externas, realice los siguientes pasos:

Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto*.

Paso 2: Ingrese el nombre del proyecto: Ejemplo_Fuentes (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 3: Descargue las fuentes a utilizar y cópielas en la carpeta assets (ver fig.).



Paso 4: Modifique la actividad principal.

```
package com.nolasco.ejemplo_fuentes;

import android.os.Bundle;
import android.app.Activity;
import android.content.Context;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Rect;
import android.graphics.Typeface;
import android.graphics.Paint.Style;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.support.v4.app.NavUtils;

public class MainActivity extends Activity {

    class RenderView extends View {
            Paint paint;
            Typeface face;
            public RenderView(Context context) {
                    super(context);
                    paint = new Paint();
            }
            protected void onDraw(Canvas canvas) {


    canvas.drawRGB(255,255,255);

        paint.setARGB(255,0,0,0);
        paint.setTextSize(30);

        face = Typeface.createFromAsset(getAssets(),"A Drink For All Ages.TTF");
        paint.setTypeface(face);
        canvas.drawText("Tipo de Fuente : A Drink For All Ages", 0, 220, paint);

        face= Typeface.createFromAsset(getAssets(),"ABITE.ttf");
        paint.setTypeface(face);
        canvas.drawText("Tipo de Fuente : ABITE", 0, 320, paint);

        face= Typeface.createFromAsset(getAssets(),"ACAPPELLA.ttf");
        paint.setTypeface(face);
        canvas.drawText("Tipo de Fuente : ACAPPELLA", 0, 420, paint);

        face= Typeface.createFromAsset(getAssets(),"ADAMN.TTF");
        paint.setTypeface(face);
        canvas.drawText("Tipo de Fuente : ADAMN", 0, 520, paint);

        face= Typeface.createFromAsset(getAssets(),"adrip1.ttf");
        paint.setTypeface(face);
        canvas.drawText("Tipo de Fuente : adrip1", 0, 620, paint);

        face = Typeface.createFromAsset(getAssets(),"AGothiqueTime.ttf");
        paint.setTypeface(face);
        canvas.drawText("Tipo de Fuente : aGothiqueTime", 0, 720, paint);
```

```
        face= Typeface.createFromAsset(getAssets(),"alittlepot.ttf");
        paint.setTypeface(face);
        canvas.drawText("Tipo de Fuente : alittlepot", 0, 820, paint);

            }
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(new RenderView(this));
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.activity_main, menu);
        return true;
    }


}
```

14.5. DATOS



Una manera muy sencilla y cómoda de guardar; por ejemplo, la información de un juego como es el usuario y la puntuación es por medio del uso de las preferencias.

14.5.1. Persistencia y preferencia

Son almacenes clave valor que permite crear varios almacenes de preferencias; sin embargo, normalmente bastará con usar un almacén por defecto para cada aplicación.

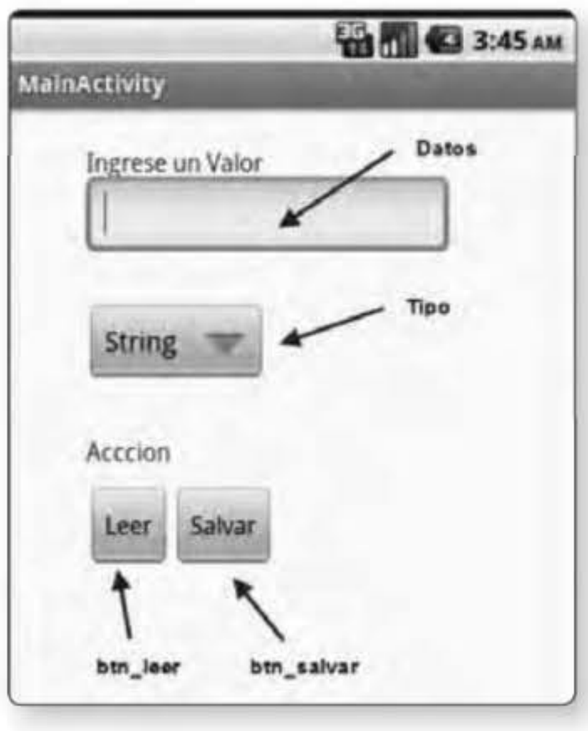
14.5.2. Aplicacion sobre el uso de Preferencias

Cree una aplicación que permita demostrar el uso de Preferencias.

Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto.*

Paso 2: Ingrese el nombre del proyecto: Ejemplo_Preferencias (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 3: Modifique el archivo: activity_main.xml.



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” >

    <TextView
        android:id=”@+id/textView1”
        android:layout_width=”wrap_content”
        android:layout_height=”wrap_content”
        android:layout_alignParentLeft=”true”
        android:layout_alignParentTop=”true”
        android:layout_marginLeft=”43dp”
        android:layout_marginTop=”22dp”
        android:text=”Ingrese un Valor” />

    <EditText
        android:id=”@+id/Datos”
        android:layout_width=”wrap_content”
        android:layout_height=”wrap_content”
        android:layout_alignLeft=”@+id/textView1”
        android:layout_marginTop=”39dp”
        android:ems=”10” />

    <Spinner
        android:id=”@+id/Tipo”
        android:layout_width=”wrap_content”
        android:layout_height=”wrap_content”
        android:layout_alignLeft=”@+id/Datos”
        android:layout_below=”@+id/Datos”
        android:layout_marginTop=”24dp”
        android:entries=”@array/etiquetas”
        android:entryValues=”@array/valores”
        />

    <TextView
        android:id=”@+id/textView2”
        android:layout_width=”wrap_content”
        android:layout_height=”wrap_content”
        android:layout_alignLeft=”@+id/Tipo”
        android:layout_below=”@+id/Tipo”
        android:layout_marginTop=”27dp”
        android:text=”Acccion” />

    <TableLayout
        android:layout_width=”wrap_content”
        android:layout_height=”wrap_content”
        android:layout_alignLeft=”@+id/textView2”
        android:layout_centerVertical=”true” >

        <TableRow
            android:id=”@+id/tableRow1”
            android:layout_width=”wrap_content”
            android:layout_height=”wrap_content” >

            <Button
                android:id=”@+id/btn_Leer”
```

```
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Leer" />

        <Button
            android:id="@+id/btn_Salvar"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Salvar" />

    </TableRow>
    </TableLayout>

</RelativeLayout>
```

El Spinner utilizará los valores del archivo: strings.xml , desde un arreglo de cadenas.

strings.xml

```
<resources>

    <string name="app_name">Ejemplo_Preferencias </string>
    <string name="hello_world">Hello world!</string>
    <string name="menu_settings">Settings</string>
    <string name="title_activity_main">MainActivity</string>

    <string-array name="etiquetas">
        <item >String</item>
        <item >Boolean</item>
        <item >Integer</item>
        <item >Float</item>
    </string-array>

      <array name="valores">
        <item >0</item>
        <item >1</item>
        <item >2</item>
        <item >3</item>

    </array>

</resources>
```

MainActivity.java

```
package com.nolascov.ejemplo_preferencias;

import android.os.Bundle;
import android.app.Activity;
import android.content.SharedPreferences;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.TextView;
```

```
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Leer" />

        <Button
            android:id="@+id/btn_Salvar"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Salvar" />

    </TableRow>
    </TableLayout>

</RelativeLayout>
```

El Spinner utilizará los valores del archivo: strings.xml , desde un arreglo de cadenas.

strings.xml

```
<resources>

    <string name="app_name">Ejemplo_Preferencias </string>
    <string name="hello_world">Hello world!</string>
    <string name="menu_settings">Settings</string>
    <string name="title_activity_main">MainActivity</string>

    <string-array name="etiquetas">
        <item >String</item>
        <item >Boolean</item>
        <item >Integer</item>
        <item >Float</item>
    </string-array>

      <array name="valores">
        <item >0</item>
        <item >1</item>
        <item >2</item>
        <item >3</item>

    </array>

</resources>
```

MainActivity.java

```
package com.nolascov.ejemplo_preferencias;

import android.os.Bundle;
import android.app.Activity;
import android.content.SharedPreferences;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.TextView;
```

```
public class MainActivity extends Activity {

    private EditText Datos;
    private Spinner Tipo;
    private Button btn_Leer;
    private Button btn_Salvar;
    private TextView Salida;


    // variable para las prefrencias
    private static String PREFERENCIA=”MisPrefrencias”;
    //modo de la preferencia privado solo acceso para la actividad
    private static int mode =Activity.MODE_PRIVATE;
    //variable de preferencia compartida
    private SharedPreferences MisPreferenciasCompartidas;



    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Datos = (EditText) findViewById(R.id.Datos);
        Tipo = (Spinner) findViewById(R.id.Tipo);
        btn_Leer = (Button) findViewById(R.id.btn_Leer);
        btn_Salvar = (Button) findViewById(R.id.btn_Salvar);
        Salida = (TextView) findViewById(R.id.Salida);
        //creando nuestra preferencia
        MisPreferenciasCompartidas= getSharedPreferences(PREFERENCIA, mode);

        this.btn_Leer.setOnClickListener(new OnClickListener() {

                public void onClick(View v) {
                        //datos leidos de la preferencia
                        String cadena = “”;
                        //el tipo de la preferencia , el item seleccioando
                        Integer i = Tipo.getSelectedItemPosition();
                        switch(i)
                        {
                        case 0:
                                cadena              =MisPreferenciasCompartidas.
getString(“cadena”,”Hola”);
                                break;
                        case 1:
                                cadena  =new  Boolean(MisPreferenciasCompartidas.
getBoolean(“Logico”,true)).toString();
                                break;
                        case 2:
                                cadena  =new  Integer(MisPreferenciasCompartidas.
getInt(“Entero”,11)).toString();

                                break;
                        case 3:
                                cadena =new Float(MisPreferenciasCompartidas.get
Float(“flotante”,(float)3.5)).toString();

                                break;
                        default:
```

```
                                cadena =””;
                                break;
                        }

                        Salida.setText(cadena);

                }
        });


this.btn_Salvar.setOnClickListener(new OnClickListener() {

                        public void onClick(View v) {
                                //datos leidos de la preferencia
                                String Datos2 = Datos.getText().toString();
                                //el tipo de la preferencia , el item seleccioando
                                Integer i = Tipo.getSelectedItemPosition();

                                SharedPreferences.Editor                    editor
=MisPreferenciasCompartidas.edit();
                                switch(i)
                                {
                                case 0:
                                        editor.putString(“cadena”, Datos2);
                                        break;
                                case 1:
                                        editor.putBoolean(“Logico”,Boolean.
parseBoolean(Datos2));
                                        break;
                                case 2:
                                        editor.putInt(“Entero”,Integer.
parseInt(Datos2));
                                        break;
                                case 3:
                                        editor.putFloat(“Flotante”,Float.
parseFloat(Datos2));

                                        break;
                                default:
                                        break;
                                }

                                editor.commit();
                        }
                });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.activity_main, menu);
        return true;
    }
}
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Guarde los cambios realizados, haciendo clic en el ícono*



o presionando la combinación Ctrl+S.

Paso 2: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:*



14.6. DISPLAYMETRICS

Devuelve una estructura que describe la información general sobre la pantalla del dispositivo móvil, tal como :

- tamaño
- densidad
- escala de la fuente

Ejemplo:

```
DisplayMetrics metrics = new DisplayMetrics();
 getWindowManager().getDefaultDisplay().getMetrics(metrics);
```

Constantes

int	DENSITY_DEFAULT	La densidad de referencia utilizada en todo el sistema.
int	DENSITY_HIGH	Estándar cuantificado DPI para pantalla de alta resolucion
int	DENSITY_LOW	Estándar cuantificado DPI para pantalla de baja resolucion
int	DENSITY_MEDIUM	Estándar cuantificado DPI para pantalla de mediana resolucion
int	DENSITY_TV	Densidad secundaria
int	DENSITY_XHIGH	Estándar cuantificado DPI para pantallas de extra alta densidad
int	DENSITY_XXHIGH	Estándar cuantificado DPI para pantallas de extra extra alta densidad

Propiedades

public float	density	La densidad de la pantalla.
public int	densityDpi	La densidad de la pantalla expresada en pulgadas
public int	**heightPixels**	**La altura de la pantalla en pixeles**
public float	scaledDensity	Un factor de escala para fuentes que se muestran en la pantalla.
public int	**widthPixels**	**El ancho de la pantalla en pixeles**
public float	xdpi	Los píxeles por pulgada de la pantalla en la dimensión X.
public float	ydpi	Los píxeles por pulgada de la pantalla en la dimensión Y.



Ahora diseñe una aplicación que permita realizar algunos trazos.

Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto*.

Paso 2: Ingrese el nombre del proyecto: EjemploTrazo (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

MainActivity.java

```
package com.nolascov.ejemplopintado;

import android.app.Activity;
import android.content.Context;
import android.content.res.AssetManager;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Path;
import android.os.Bundle;
import android.util.DisplayMetrics;
import android.view.Display;
import android.view.KeyEvent;
import android.view.MotionEvent;
import android.view.Window;
import android.view.WindowManager;
import java.io.InputStream;

public class MainActivity extends Activity {

    protected Display  display;
    protected MainView view;
    protected Paint  paint;
    protected Path   path;
    protected Canvas scr_canvas;
    protected Canvas dib_canvas;
    protected Bitmap scr_bitmap;
    protected Bitmap dib_bitmap;
    protected Bitmap icon_accept;
    protected Bitmap icon_accept_sel;
    protected Bitmap icon_cancel;
    protected Bitmap icon_cancel_sel;

    protected int width;
    protected int height;
    protected int color;
    protected int size;

    protected float brush_radius;
    protected boolean anti_alias;
    protected int[] palette;

    protected MainScreen    mainScreen;
    protected PaletteScreen paletteScreen;
    protected Screen        currScreen;

    @Override
```

```
    public void onCreate(Bundle icicle) {
        super.onCreate(icicle);
        //pantalla sin titulo
        requestWindowFeature(Window.FEATURE_NO_TITLE);
     getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,   WindowManager.
LayoutParams.FLAG_FULLSCREEN);
             // dm objetos que posee ancho , alto , densidad de la pantalla
        DisplayMetrics dm = new DisplayMetrics();
        // rellena el objetos con los datos
        getWindowManager().getDefaultDisplay().getMetrics(dm);

        //obtienes las imagenes desde los recursos
        icon_accept = BitmapFactory.decodeResource(getResources(), R.drawable.button_
accept);
          icon_accept_sel = BitmapFactory.decodeResource(getResources(), R.drawable.
button_accept_sel);
        icon_cancel = BitmapFactory.decodeResource(getResources(), R.drawable.button_
cancel);
          icon_cancel_sel = BitmapFactory.decodeResource(getResources(), R.drawable.
button_cancel_sel);


        //obteniendo datos para el dm
        //ancho de la pantalla en pixeles
        width = dm.widthPixels;
      //alto de la pantalla en pixeles
        height = dm.heightPixels;
        //radio de la paleta
        brush_radius = 5;
        view = new MainView(this);
        //color negro
        color = Color.BLACK;
        //brocha para pintar
        paint = new Paint();
        //color a la brocha
        paint.setColor(color);
        //guardar puntos
        path = new Path();
      //crear una imagen en 32 bits del ancho y alto de la pantalla , pantalla de fondo
        scr_bitmap = Bitmap.createBitmap(width, height, Bitmap.Config.ARGB_8888);
        //es el objetos para pintar en el bitmap
        scr_canvas = new Canvas(scr_bitmap);
        //carga una paleta de 256 colores
        palette = loadPalette(this, "HEXEN.PAL");
        // instanciar la clase mainscreen
        mainScreen = new MainScreen(this);
        //no tiene uso
        paletteScreen = new PaletteScreen(this);
        // hace referencia a la pantalla actual
        currScreen = mainScreen;
        // muestra la pantalla
        mainScreen.show();

        setContentView(view);
    }

    @Override
    public boolean onTouchEvent(MotionEvent event) {
        return currScreen.onTouchEvent(event);
    }
```

```
    public void onCreate(Bundle icicle) {
        super.onCreate(icicle);
        //pantalla sin titulo
        requestWindowFeature(Window.FEATURE_NO_TITLE);
     getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,   WindowManager.
LayoutParams.FLAG_FULLSCREEN);
             // dm objetos que posee ancho , alto , densidad de la pantalla
        DisplayMetrics dm = new DisplayMetrics();
        // rellena el objetos con los datos
        getWindowManager().getDefaultDisplay().getMetrics(dm);

        //obtienes las imagenes desde los recursos
        icon_accept = BitmapFactory.decodeResource(getResources(), R.drawable.button_
accept);
          icon_accept_sel = BitmapFactory.decodeResource(getResources(), R.drawable.
button_accept_sel);
        icon_cancel = BitmapFactory.decodeResource(getResources(), R.drawable.button_
cancel);
          icon_cancel_sel = BitmapFactory.decodeResource(getResources(), R.drawable.
button_cancel_sel);


        //obteniendo datos para el dm
        //ancho de la pantalla en pixeles
        width = dm.widthPixels;
      //alto de la pantalla en pixeles
        height = dm.heightPixels;
        //radio de la paleta
        brush_radius = 5;
        view = new MainView(this);
        //color negro
        color = Color.BLACK;
        //brocha para pintar
        paint = new Paint();
        //color a la brocha
        paint.setColor(color);
        //guardar puntos
        path = new Path();
      //crear una imagen en 32 bits del ancho y alto de la pantalla , pantalla de fondo
        scr_bitmap = Bitmap.createBitmap(width, height, Bitmap.Config.ARGB_8888);
        //es el objetos para pintar en el bitmap
        scr_canvas = new Canvas(scr_bitmap);
        //carga una paleta de 256 colores
        palette = loadPalette(this, "HEXEN.PAL");
        // instanciar la clase mainscreen
        mainScreen = new MainScreen(this);
        //no tiene uso
        paletteScreen = new PaletteScreen(this);
        // hace referencia a la pantalla actual
        currScreen = mainScreen;
        // muestra la pantalla
        mainScreen.show();

        setContentView(view);
    }

    @Override
    public boolean onTouchEvent(MotionEvent event) {
        return currScreen.onTouchEvent(event);
    }
```

```
    @Override
    public final boolean onKeyDown(final int keyCode, final KeyEvent event) {
        return currScreen.onKeyDown(keyCode, event);
    }


    //manejo de la paleta de 256 colores
    public static int[] loadPalette(Context context, String name) {
        int palette[] = new int[256];
        //lista de byte ( flujo de entrada )
        InputStream is = null;
        //tener acceso a la carpeta assets
        AssetManager am = context.getAssets();
        //leer la paleta de colores
        try {
            //apertura
            is = am.open(name);
            for(int idx=0; idx<256; idx++) {
            //formato RGB 4 byte , alfa - Rojo - Verde - Azul
                palette[idx] = 0xFF000000 | (is.read() << 16) | (is.read() << 8) |
is.read();
            }
        } catch(Exception ex) {
        } finally {
            if(is != null) {
                try {
                    is.close();
                } catch(Exception ex) {}
            }
            return palette;
        }

    }
```

MainScreen.java

```
package com.nolascov.ejemplopintado;

import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Color;
import android.net.Uri;
import android.provider.MediaStore;
import android.view.KeyEvent;
import android.view.MotionEvent;
import android.widget.Toast;

public class MainScreen extends Screen {

    private   AlertDialog clearDialog;
    private   AlertDialog brushDialog;
    private   AlertDialog colorDialog;
    private   AlertDialog exitDialog;
```

```
    protected Button btClear;
    protected Button btColor;
    protected Button btBrush;
    protected Button btShare;
    protected Button btExit;

    protected Bitmap icon_clear;
    protected Bitmap icon_clear_sel;
    protected Bitmap icon_brush;
    protected Bitmap icon_brush_sel;
    protected Bitmap icon_color;
    protected Bitmap icon_color_sel;
    protected Bitmap icon_share;
    protected Bitmap icon_share_sel;
    protected Bitmap icon_exit;
    protected Bitmap icon_exit_sel;

    protected int toolbar_height;
    protected int icon_width;
    protected int icon_height;
    protected int icon_x;
    protected int icon_y;

    protected float lastX;
    protected float lastY;
    protected float currX;
    protected float currY;

    public MainScreen(Context context) {
        super(context);
      //carga los iconos en la barra de herramientas
        loadBitmaps();
      //inicializa la barra herramientas
        initToolBox();
      //canvas donde se dibujara
        main.dib_bitmap = Bitmap.createBitmap(main.width, main.height-toolbar_height,
Bitmap.Config.ARGB_8888);
        //objeto para dibujar
        main.dib_canvas = new Canvas(main.dib_bitmap);
        //borra el canvas con el color blanco
        main.dib_canvas.drawColor(Color.WHITE);
        //la brocha a color negro
        main.paint.setColor(0xFF000000);

    }

    @Override
    protected void show() {
      //pinta color gris
        main.scr_canvas.drawColor(Color.GRAY);
        //dibuja la barra de herramientas
        drawToolBar();
        main.scr_canvas.drawBitmap(main.dib_bitmap, 0, toolbar_height, null);
        main.view.invalidate();

    }

    @Override
    protected boolean onTouchEvent(MotionEvent event) {
        boolean res = false;
        int action = event.getActionMasked();
         if(action == MotionEvent.ACTION_DOWN || action == MotionEvent.ACTION_MOVE ||
action == MotionEvent.ACTION_UP) {
```

```
                res = true;
                float x = event.getX();
                float y = event.getY();
                if(y > toolbar_height ) {
                    y -= toolbar_height;
                    switch(action) {
                        case MotionEvent.ACTION_DOWN:
                            currX = x;
                            currY = y;
                            lastX = currX;
                            lastY = currY;
                             //dibuja un punto o circulo
                             drawPoint(currX, currY);
                             //actualizar la pantalla
                            main.scr_canvas.drawBitmap(main.dib_bitmap, 0, toolbar_height,
null);
                          //para repinte la pantalla
                             main.view.invalidate();
                             break;
                        case MotionEvent.ACTION_MOVE:
                            currX = x;
                            currY = y;
                          //dibujo una linea desde el punto actual , al punto anterior
                             drawLine();
                          //actualizar la pantalla
                            main.scr_canvas.drawBitmap(main.dib_bitmap, 0, toolbar_height,
null);
                          //para repinte la pantlla
                             main.view.invalidate();
                             lastX = currX;
                             lastY = currY;
                             break;
                    }
                } else {
                 // los botones presionado
                    if( btClear.isTouched(x,y) ) {
                        btClear.onTouchEvent(event);
                    } else
                    if( btBrush.isTouched(x,y) ) {
                        btBrush.onTouchEvent(event);
                    } else
                    if( btColor.isTouched(x,y) ) {
                        btColor.onTouchEvent(event);
                    } else
                    if( btShare.isTouched(x,y) ) {
                        btShare.onTouchEvent(event);
                    } else
                    if( btExit.isTouched(x,y) ) {
                        btExit.onTouchEvent(event);
                    }
                    main.view.invalidate();
                }
            }
            return res;
        }

        @Override
        protected boolean onKeyDown(int keyCode, KeyEvent event) {
            boolean res = false;
            if(event.getAction() == KeyEvent.ACTION_DOWN) {
                if(keyCode == KeyEvent.KEYCODE_BACK) {
                    res = true;
                    getExitDialog();
```

```
            }
        }
        return res;
    }

    private void loadBitmaps() {
          icon_clear = BitmapFactory.decodeResource(main.getResources(), R.drawable.
icon_menu);
       icon_clear_sel = BitmapFactory.decodeResource(main.getResources(), R.drawable.
icon_menu_sel);
          icon_brush = BitmapFactory.decodeResource(main.getResources(), R.drawable.
icon_brush);
       icon_brush_sel = BitmapFactory.decodeResource(main.getResources(), R.drawable.
icon_brush_sel);
          icon_color = BitmapFactory.decodeResource(main.getResources(), R.drawable.
icon_color);
       icon_color_sel = BitmapFactory.decodeResource(main.getResources(), R.drawable.
icon_color_sel);
          icon_share = BitmapFactory.decodeResource(main.getResources(), R.drawable.
icon_share);
       icon_share_sel = BitmapFactory.decodeResource(main.getResources(), R.drawable.
icon_share_sel);
          icon_exit = BitmapFactory.decodeResource(main.getResources(), R.drawable.
icon_tool);
       icon_exit_sel = BitmapFactory.decodeResource(main.getResources(), R.drawable.
icon_tool_sel);
    }

    private void initToolBox() {
        icon_width = icon_clear.getWidth();
        icon_height = icon_clear.getHeight();
        main.size = icon_height;
        icon_y = 0;
        toolbar_height = icon_height + (icon_height >> 3);
        int sp = (main.width - (icon_width*5)) / 6;
        icon_x = sp;
        sp += icon_width;

        btClear = new Button(main, icon_x , icon_y, icon_clear, icon_clear_sel) {
            @Override
            protected void action() {
                getClearDialog().show();
            }
        };
        icon_x += sp;
        btBrush = new Button(main, icon_x , icon_y, icon_brush, icon_brush_sel) {
            @Override
            protected void action() {
                getBrushDialog().show();
            }
        };
        icon_x += sp;
        btColor = new Button(main, icon_x , icon_y, icon_color, icon_color_sel) {
            @Override
            protected void action() {
                getColorDialog().show();
            }
        };
        icon_x += sp;
        btShare = new Button(main, icon_x , icon_y, icon_share, icon_share_sel) {
            @Override
            protected void action() {
                shareBitmap();
```

```
            }
        };
        icon_x += sp;
        btExit = new Button(main, icon_x , icon_y, icon_exit, icon_exit_sel) {
            @Override
            protected void action() {
                getExitDialog().show();
            }
        };
    }

    public AlertDialog getClearDialog() {
        if(!(clearDialog instanceof AlertDialog)) {
            clearDialog = new AlertDialog.Builder(main).create();
            clearDialog.setTitle("Clear Canvas?");
        clearDialog.setButton(AlertDialog.BUTTON_POSITIVE, "Yes", new DialogInterface.
OnClickListener() {
             public void onClick(DialogInterface paramDialogInterface, int paramInt) {
                    main.dib_canvas.drawColor(Color.WHITE);
                main.scr_canvas.drawBitmap(main.dib_bitmap, 0, toolbar_height, null);
                    main.view.invalidate();
                }
            });
           clearDialog.setButton(AlertDialog.BUTTON_NEGATIVE, "No", (DialogInterface.
OnClickListener)null);
        }
        return clearDialog;
      }

    public AlertDialog getBrushDialog() {
        if(!(brushDialog instanceof AlertDialog)) {
            brushDialog = new BrushDialog(main);
        }
        return brushDialog;
    }

    public AlertDialog getColorDialog() {
        if(!(colorDialog instanceof AlertDialog)) {
            colorDialog = new ColorDialog(main);
        }
        return colorDialog;
    }

    public AlertDialog getExitDialog() {
        if(!(exitDialog instanceof AlertDialog)) {
            exitDialog = new AlertDialog.Builder(main).create();
            exitDialog.setTitle("Exit MicroPaint?");
         exitDialog.setButton(AlertDialog.BUTTON_POSITIVE, "Yes", new DialogInterface.
OnClickListener() {
             public void onClick(DialogInterface paramDialogInterface, int paramInt) {
                    SplashActivity.killed = true;
                    main.finish();
                }
            });
            exitDialog.setButton(AlertDialog.BUTTON_NEGATIVE, "No", (DialogInterface.
OnClickListener)null);
        }
        return exitDialog;
      }

    private void drawToolBar() {
        main.paint.setColor(0xFF202020);
        main.scr_canvas.drawRect(0, 0, main.width, main.size, main.paint);
```

```
            main.paint.setColor(main.color);
            btClear.paint(false);
            btColor.paint(false);
            btBrush.paint(false);
            btShare.paint(false);
            btExit.paint(false);
        }

        private void shareBitmap() {
            try {
             String fpath = MediaStore.Images.Media.insertImage(main.getContentResolver(),
main.dib_bitmap, “dib_bitmap”, null);
                Uri screenshotUri = Uri.parse(fpath);
                final Intent emailIntent = new Intent(android.content.Intent.ACTION_SEND);
                emailIntent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
                emailIntent.putExtra(Intent.EXTRA_STREAM, screenshotUri);
                emailIntent.setType(“image/png”);
                main.startActivity(Intent.createChooser(emailIntent, “Share image...”));
            } catch(Exception ex) {
                Toast.makeText(main, ex.getMessage(), Toast.LENGTH_LONG).show();
            }
        }

        private void drawLine() {
        float lenX = currX - lastX;
        float lenY = currY - lastY;
        float vx = 0;
        if(lenX > 0) {
                vx = 1;
        } else if(lenX < 0) {
                vx = -1;
        }
        float vy = 0;
        if(lenY > 0) {
                vy = 1;
        } else if(lenY < 0) {
                vy = -1;
        }
        float len;
        lenX = Math.abs(lenX);
        lenY = Math.abs(lenY);
        if(lenX > lenY) {
                vy = (lenY / lenX) * vy;
                len = lenX;
        } else {
                vx = (lenX / lenY) * vx;
                len = lenY;
        }
        float x = lastX;
        float y = lastY;
        for(int i=0; i < len; i++) {
                drawPoint((int)x, (int)y);
                x += vx;
                y += vy;
        }
        }

        private void drawPoint(final float x, final float y) {
            main.dib_canvas.drawCircle(x, y, main.brush_radius, main.paint);
        }

}
```

BrushDialog.java

```
package com.nolascov.ejemplopintado;

import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.view.View;
import android.widget.CheckBox;
import android.widget.LinearLayout;
import android.widget.LinearLayout.LayoutParams;
import android.widget.SeekBar;
import android.widget.TextView;

public class BrushDialog extends AlertDialog implements SeekBar.OnSeekBarChangeListener,
DialogInterface.OnClickListener, View.OnClickListener {

    private MainActivity main;
    private TextView     text;
    private CheckBox     check;
    private int          brush_radius;

    public BrushDialog(Context context) {
        super(context);
        main = (MainActivity)context;
        brush_radius = (int)main.brush_radius;
       LayoutParams params = new LayoutParams(LayoutParams.MATCH_PARENT, LayoutParams.
MATCH_PARENT);
        LinearLayout layout = new LinearLayout(main);
        layout.setOrientation(LinearLayout.VERTICAL);
        layout.setPadding(8,8,8,8);

        text = new TextView(main);
        text.setText("size: "+(brush_radius << 1));
        layout.addView(text);
        SeekBar seekbar = new SeekBar(main);
        seekbar.setLayoutParams(params);
        seekbar.setOnSeekBarChangeListener(this);
        seekbar.setMax(100);
        seekbar.setProgress(brush_radius << 1);
        layout.addView(seekbar);

        check = new CheckBox(main);
        check.setText("Antialias");
        check.setChecked(main.anti_alias);
        check.setOnClickListener(this);
        layout.addView(check);

        setView(layout);
        setTitle("Brush Settings");
        setIcon(0);
        setButton(DialogInterface.BUTTON_POSITIVE, "Accept", this);
        setButton(DialogInterface.BUTTON_NEGATIVE, "Cancel", this);
    }

    @Override
    protected void onStart() {
        super.onStart();
        brush_radius = (int)main.brush_radius;
       LayoutParams params = new LayoutParams(LayoutParams.MATCH_PARENT, LayoutParams.
```

```
MATCH_PARENT);
        LinearLayout layout = new LinearLayout(main);
        layout.setOrientation(LinearLayout.VERTICAL);
        layout.setPadding(8,8,8,8);

        text = new TextView(main);
        text.setText("size: "+(brush_radius << 1));
        layout.addView(text);
        SeekBar seekbar = new SeekBar(main);
        seekbar.setLayoutParams(params);
        seekbar.setOnSeekBarChangeListener(this);
        seekbar.setMax(100);
        seekbar.setProgress(brush_radius << 1);
        layout.addView(seekbar);
    }

    public void onProgressChanged(SeekBar seekbar, int progress, boolean user) {
        brush_radius = progress >> 1;
        text.setText("size: "+progress);
    }

    public void onStartTrackingTouch(SeekBar seekbar) {}

    public void onStopTrackingTouch(SeekBar seekbar) {}

    public void onClick(DialogInterface dialog, int button) {
        if( button == DialogInterface.BUTTON_POSITIVE ) {
            main.brush_radius = brush_radius;
        }
    }

    public void onClick(View arg0) {
        if(main.anti_alias) {
            main.anti_alias = false;
        } else {
            main.anti_alias = true;
        }
        main.paint.setAntiAlias(main.anti_alias);
    }

}
```

Button.java

```
package com.nolascov.ejemplopintado;

import android.graphics.Bitmap;
import android.view.MotionEvent;

public class Button {

    protected float  x;
    protected float  y;
    protected float  xx;
    protected float  yy;
    protected int    width;
    protected int    height;
    protected Bitmap icon;
    protected Bitmap icon_sel;
```

```
    protected float  x1;
    protected float  y1;
    protected float  x2;
    protected float  y2;
    private MainActivity main;


    public Button(final MainActivity main,final float x, final float y, final Bitmap icon,
final Bitmap icon_sel) {
        this.main = main;
        this.x = x;
        this.y = y;
        width = icon.getWidth();
        height = icon.getHeight();
        xx = (x+width)-1;
        yy = (y+height)-1;
        this.icon = icon;
        this.icon_sel = icon_sel;
        x1 = x+2;
        y1 = y+2;
        x2 = xx-2;
        y2 = yy-2;
    }

    protected final boolean isTouched(final MotionEvent event) {
        return isTouched(event.getX(), event.getY());
    }

    protected final boolean isTouched(final float x, final float y) {
        return (x1 < x) && (x < x2) && (y1 < y) && (y < y2);
    }

    protected final boolean onTouchEvent(final MotionEvent event) {
        boolean res = false;
        int action = event.getActionMasked();
        if(action == MotionEvent.ACTION_DOWN) {
            paint(true);
            res = true;
        } else if (action == MotionEvent.ACTION_UP) {
            paint(false);
            res = true;
            action();
        }
        return res;
    }

    protected final void paint(final boolean hover) {
        if(hover) {
            main.scr_canvas.drawBitmap(icon_sel, x, y, null);
        } else {
            main.scr_canvas.drawBitmap(icon, x, y, null);
        }
        main.view.invalidate();
    }

    protected void action() {}
```

ColorDialog.java

```
package com.nolascov.ejemplopintado;

import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.graphics.Color;
import android.widget.LinearLayout;
import android.widget.LinearLayout.LayoutParams;
import android.widget.SeekBar;
import android.widget.TextView;

public class ColorDialog extends AlertDialog implements SeekBar.OnSeekBarChangeListener,
DialogInterface.OnClickListener {

    private MainActivity main;
    private TextView    text_color;
    private TextView    text_red;
    private TextView    text_green;
    private TextView    text_blue;
    private TextView    text_alpha;
    private SeekBar     bar_red;
    private SeekBar     bar_green;
    private SeekBar     bar_blue;
    private SeekBar     bar_alpha;
    private int color;
    private int red;
    private int green;
    private int blue;
    private int alpha;

    public ColorDialog(Context context) {
        super(context);
        main = (MainActivity)context;
        LayoutParams params = new LayoutParams(LayoutParams.MATCH_PARENT, LayoutParams.
MATCH_PARENT);
        LinearLayout layout = new LinearLayout(main);
        layout.setOrientation(LinearLayout.VERTICAL);
        layout.setPadding(8,8,8,8);

        text_color = new TextView(main);
        text_color.setText(" ");
        layout.addView(text_color);

        text_red = new TextView(main);
        layout.addView(text_red);
        bar_red = new SeekBar(main);
        bar_red.setId(Color.RED);
        bar_red.setLayoutParams(params);
        bar_red.setOnSeekBarChangeListener(this);
        bar_red.setMax(255);
        layout.addView(bar_red);

        text_green = new TextView(main);
        layout.addView(text_green);
        bar_green = new SeekBar(main);
        bar_green.setId(Color.GREEN);
        bar_green.setLayoutParams(params);
        bar_green.setOnSeekBarChangeListener(this);
        bar_green.setMax(255);
```

```
        layout.addView(bar_green);

        text_blue = new TextView(main);
        layout.addView(text_blue);
        bar_blue = new SeekBar(main);
        bar_blue.setId(Color.BLUE);
        bar_blue.setLayoutParams(params);
        bar_blue.setOnSeekBarChangeListener(this);
        bar_blue.setMax(255);
        layout.addView(bar_blue);

        text_alpha = new TextView(main);
        layout.addView(text_alpha);
        bar_alpha = new SeekBar(main);
        bar_alpha.setId(Color.GRAY);
        bar_alpha.setLayoutParams(params);
        bar_alpha.setOnSeekBarChangeListener(this);
        bar_alpha.setMax(255);
        layout.addView(bar_alpha);

        setView(layout);
        setTitle("Color Settings");
        setIcon(0);
        setButton(DialogInterface.BUTTON_POSITIVE, "Accept", this);
        setButton(DialogInterface.BUTTON_NEGATIVE, "Cancel", this);
    }

    @Override
    protected void onStart() {
        super.onStart();
        color = main.color;
        red = Color.red(color);
        green = Color.green(color);
        blue = Color.blue(color);
        alpha = Color.alpha(color);

        text_color.setBackgroundColor(color);

        text_red.setText("red: "+red);
        bar_red.setBackgroundColor(Color.rgb(red,0,0));
        bar_red.setProgress(red);

        text_green.setText("green: "+green);
        bar_green.setBackgroundColor(Color.rgb(0,green,0));
        bar_green.setProgress(green);

        text_blue.setText("blue: "+blue);
        bar_blue.setBackgroundColor(Color.rgb(0,0,blue));
        bar_blue.setProgress(blue);

        text_alpha.setText("alpha: "+alpha);
        bar_alpha.setBackgroundColor(Color.rgb(alpha,alpha,alpha));
        bar_alpha.setProgress(alpha);
    }

    public void onProgressChanged(SeekBar seekbar, int progress, boolean user) {
        switch(seekbar.getId()) {
            case Color.RED:
                red = progress;
                bar_red.setBackgroundColor(Color.rgb(red,0,0));
                text_red.setText("red: "+red);
```

```
                break;
            case Color.GREEN:
                green = progress;
                bar_green.setBackgroundColor(Color.rgb(0,green,0));
                text_green.setText("green: "+green);
                break;
            case Color.BLUE:
                blue = progress;
                bar_blue.setBackgroundColor(Color.rgb(0,0,blue));
                text_blue.setText("blue: "+blue);
                break;
            case Color.GRAY:
                alpha = progress;
                bar_alpha.setBackgroundColor(Color.rgb(alpha,alpha,alpha));
                text_alpha.setText("alpha: "+alpha);
                break;
        }
        color = Color.argb(alpha, red, green, blue);
        text_color.setBackgroundColor(color);
    }

    public void onStartTrackingTouch(SeekBar seekbar) {
    }

    public void onStopTrackingTouch(SeekBar seekbar) {
    }

    public void onClick(DialogInterface dialog, int button) {
        if( button == DialogInterface.BUTTON_POSITIVE ) {
            main.color = color;
            main.paint.setColor(main.color);
        }
    }

}
```

MainView.java

```
package com.nolascov.ejemplopintado;

import android.content.Context;
import android.graphics.Canvas;
import android.view.View;

public class MainView extends View {
    private MainActivity main;

    public MainView(Context context) {
        super(context);
        main = (MainActivity)context;
    }

    @Override
    protected void onDraw(Canvas canvas) {
        canvas.drawBitmap(main.scr_bitmap, 0, 0, null);
    }

}
```

PaletteScreen.java

```
package com.nolascov.ejemplopintado;

import android.content.Context;
import android.graphics.Color;
import android.view.KeyEvent;
import android.view.MotionEvent;

public class PaletteScreen extends Screen {
    private Button btAccept;
    private Button btCancel;

    private int box;
    private int palX1;
    private int palY1;
    private int palX2;
    private int palY2;
    private int palWdt;
    private int palHgt;
    private int color;

    public PaletteScreen(Context context) {
        super(context);
        box = main.size >> 2;
        palWdt = box << 4;
        palHgt = palWdt;
        palX1 = (main.width - palWdt) >> 1;
        palY1 = main.size+box;
        palX2 = palX1+palWdt;
        palY2 = palY1+palHgt;
        color = main.color;

       btAccept = new Button(main, 0 , main.height-main.icon_accept.getHeight(), main.
icon_accept, main.icon_accept_sel) {
            @Override
            protected void action() {
                main.color = color;
                goback();
            }
        };
          btCancel = new Button(main, main.width-main.icon_cancel.getWidth() , main.
height-main.icon_cancel.getHeight(), main.icon_cancel, main.icon_cancel_sel) {
            @Override
            protected void action() {
                goback();
            }
        };
    }

    @Override
    protected void show() {
        main.scr_canvas.drawColor(Color.DKGRAY);
        drawColorBox();
        drawPalette();
        btAccept.paint(false);
        btCancel.paint(false);
        main.view.invalidate();
    }

    private void drawColorBox() {
```

```
        main.paint.setColor(color);
        main.scr_canvas.drawRect(0, 0, main.width, main.size, main.paint);
    }

    private void drawPalette() {
        int col = 0;
        for(int i=0; i<16; i++) {
            for(int j=0; j<16; j++) {
                main.paint.setColor(main.palette[col]);
                col++;
             main.scr_canvas.drawRect((j*box)+palX1, (i*box)+palY1, (j*box)+box+palX1,
(i*box)+box+palY1, main.paint);
            }
        }
    }

    @Override
    protected final boolean onTouchEvent(final MotionEvent event) {
        boolean res = false;
        int action = event.getActionMasked();
         if(action == MotionEvent.ACTION_DOWN || action == MotionEvent.ACTION_MOVE ||
action == MotionEvent.ACTION_UP) {
            res = true;
            float x = event.getX();
            float y = event.getY();
            if((x > palX1) && (y > palY1) && (x < palX2) && (y < palY2)) {
                int px = (int)((x-palX1)/box);
                int py = (int)((y-palY1)/box);
                color = main.palette[(py << 4) | px];
                drawColorBox();
                main.view.invalidate();
            } else {
                if( btAccept.isTouched(x,y) ) {
                    btAccept.onTouchEvent(event);
                } else
                if( btCancel.isTouched(x,y) ) {
                    btCancel.onTouchEvent(event);
                }
            }
        }
        return res;
    }

    @Override
    protected boolean onKeyDown(int keyCode, KeyEvent event) {
        boolean res = false;
        if(event.getAction() == KeyEvent.ACTION_DOWN) {
            if(keyCode == KeyEvent.KEYCODE_BACK) {
                res = true;
                goback();
            }
        }
        return res;
    }

    private void goback() {
        main.paint.setColor(main.color);
        main.currScreen = main.mainScreen;
        main.mainScreen.show();
    }

}
```

Screen.java

```
package com.nolascov.ejemplopintado;

import android.content.Context;
import android.view.KeyEvent;
import android.view.MotionEvent;

public class Screen {
    protected MainActivity main;

    public Screen(Context context) {
        main = (MainActivity)context;
    }

    protected boolean onTouchEvent(final MotionEvent event) {
        return false;
    }

    protected boolean onKeyDown(final int keyCode, final KeyEvent event) {
        return false;
    }

    protected void show() {}

}
```

SplashActivity.java

```
package com.nolascov.ejemplopintado;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.Window;
import android.view.WindowManager;
import android.widget.ImageView;
import java.util.Timer;
import java.util.TimerTask;

public class SplashActivity extends Activity {
    protected static boolean killed;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //desactivar titulo
        requestWindowFeature(Window.FEATURE_NO_TITLE);
        //mostrar en toda la pantalla
      getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN, WindowManager.
LayoutParams.FLAG_FULLSCREEN);
        //la imagen a mostrar inicialmente
        ImageView image = new ImageView(this);
        //codigo imagen
```

```
        image.setImageResource(R.drawable.logo);
        //colocar la imagen por defecto
        setContentView(image);
    }

    @Override
    protected void onStart() {
        super.onStart();
        //preguntar salida completa de la app
        if(killed) {
            killed = false;
            //cerrar la actividad
            finish();
        } else {
             //timer de 2 segundos
            new Timer().schedule(new TimerTask() {
                @Override
                public void run() {
                  Intent intent = new Intent(SplashActivity.this, MainActivity.class);
                    // no borrar la acterior actividad
                    intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                    startActivity(intent);
                }
            }, 4000);
        }
    }

}
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:*



Puede descargar la aplicación desde el siguiente link:

https://play.google.com/store/search?q=Inkadroid&c=apps

CAP.

15

Introducción a la inteligencia artificial: Angentes Android y dispositivos móviles

15.1. INTRODUCCIÓN

Los agentes surgieron de la investigacion en inteligencia artificial (IA) y específicamente de la inteligencia artificial distribuida (DIA). Al principio, la IA hablaba de los agentes como programas especiales cuya naturaleza y construcción no se llegaba a detallar. Recientemente, se ha escrito acerca de este tema, explicando que se puede entender por un agente y que no, en general, atendiendo a caracteristicas que debe tener un agente: autonomia, reactividad, iniciativa, habilidad social, etc.

En este libro se toma la definicion de Newell [Newell 82], una definicion de agente influenciada por la IA. Newell asume que existen diferentes niveles de abstraccion en el diseno de sistemas, entre ellos el nivel de simbolos yel nivel de conocimiento.Del mismo modo que un programa convencional maneja simbolos (e.g. expresiones, variables) en el nivel de simbolos, un agente, que equivale a un programa en el nivel del conocimiento, maneja unicamente conocimiento y se comporta de acuerdo con el principio de racionalidad. Este principio estipula que un agente emprende acciones porque esta buscando satisfacer un objetivo.

Se ha tomado esta definicion porque establece que el concepto de agente es simplemente una forma de ver un sistema, como tambien indica Shoham [Shoham 93], que luego puede reflejarse a nivel de simbolos, con un programa tradicional. Esta concepcion es compatible con el ciclo de vida del software. De forma general, se pensaria en el agente como un ente que se comporta de acuerdo con el principio de racionalidad. Cuando hubiese que desarrollar fisicamente el agente, se atenderia a su arquitectura y componentes, lo cual entraria dentro del nivel simbolico.

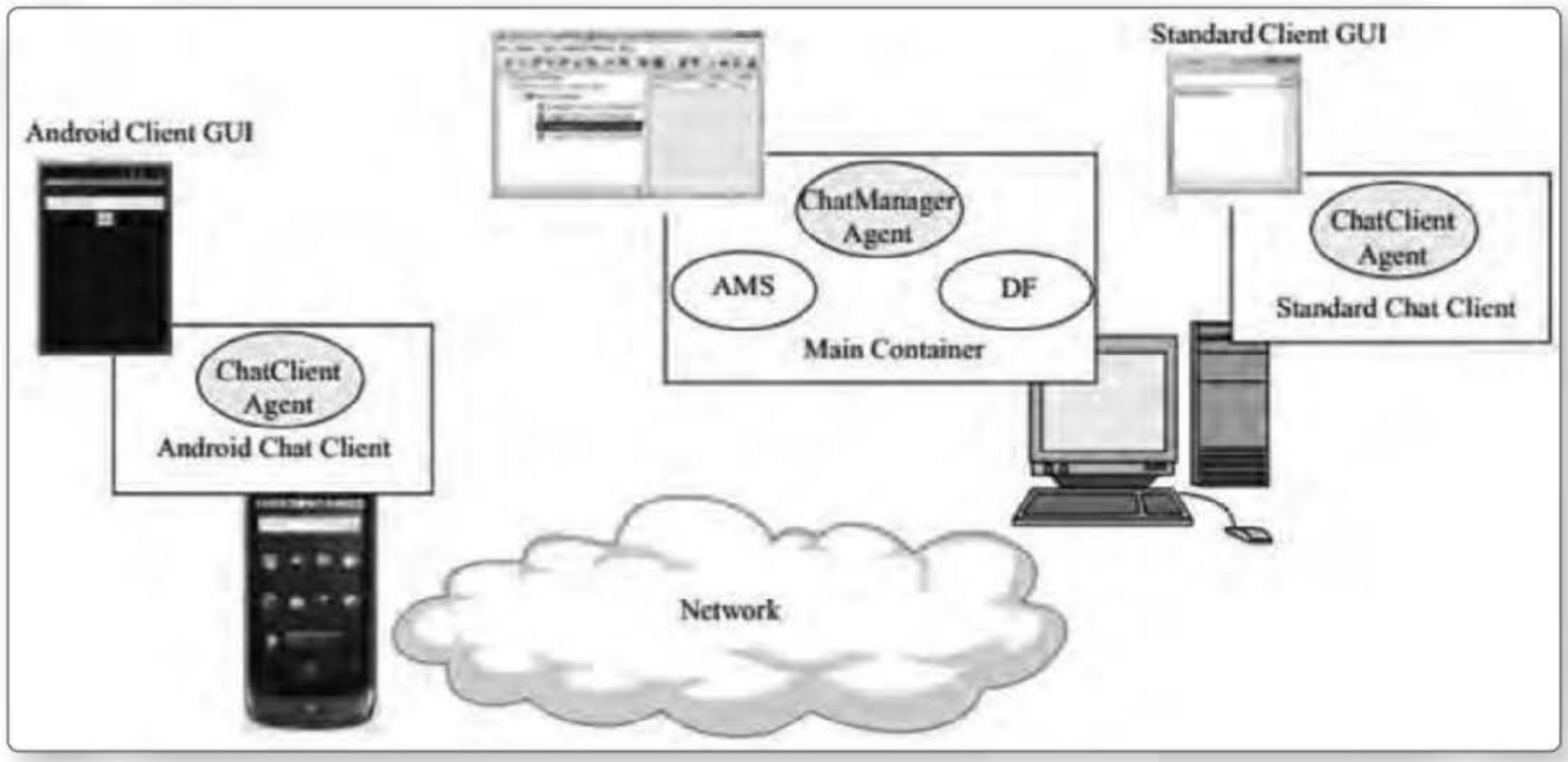
El diseno de agentes aislados ha sido estudiado en profundidad en la IA. Como resultado, se tienen las arquitecturas de subsuncion (Brooks [Brooks 91b]), arquitecturas de pizarra (Hearsay II [Carver y Lesser 92]), arquitecturas BDI (IRMA [Bratman, Israel y Pollack 88]) y arquitecturas para la resolucion generica de problemas.

Teniendo en cuenta el caracter distribuido de los entornos de aplicacion de agentes (especialmente tratamiento de informacion en internet) es normal concebir la colaboracion de varios agentes. En tales casos, la atencion se centra en como construir Sistema Multi-Agentes (SMA) que pueden entenderse como grupos de agentes que interaccionan entre si para conseguir objetivos comunes. La definicion de un SMA se puede encontrar en [Ferber 99], donde un SMA es un sistema que reune los elementos de la Ilustracion Un entorno Un conjunto de objetos. Estos objetos se encuentran integrados con el entorno, i.e. es posible en un momento dado asociar uno de estos objetos con un lugar en el entorno. Estos objetos son pasivos, pueden ser percibidos, creados, destruidos y modificados por agentes.
Un conjunto de agentes que se consideran como objetos especiales que representan las entidades activas del sistema.

Agentes como un nuevo paradigma de la Ingeniería de Software: para diseñar e implementar sistemas complejos distribuidos, como una herramienta para entender sociedades humanas: permiten una interesante forma de simular sociedades.

Los sistemas multiagentes (MAS) se describe así:

- Los agentes deben compartir conocimientos sobre el problema y las posibles soluciones.
- Los agentes en un MAS pueden compartir una meta o tener metas independientes.
- El ¨conocimiento global¨puede incluir control global, consistencia global, metas globales, etc
- La coordinación puede ser muy compleja



15.2. DESCRIPCIÓN DE LA APLICACIÓN

Como se representa en la figura anterior, esta aplicación se compone de tres módulos:

- *Android Chat Client (Android Chat Client):* Esta es una aplicación para Android que permite a los participantes a la charla de un dispositivo móvil. Incluye una interfaz gráfica de usuario de Android y un agente de la gestión de las interacciones con otros componentes.
- Standard Chat Client (estándar cliente chat): Se trata de una aplicación normal de Java que permite a los participantes de la charla desde un PC o cualquier otro dispositivo que soporte JVM estándar. Incluye los siguientes elementos:
  - Una interfaz gráfica SWING y
  - un agente de la gestión de las interacciones con otros componentes.
- *Chat Server (chatear server):* Esta es la principal plataforma de contenedores con un agente (llamado ChatManagerAgent) en la parte superior que realiza un seguimiento de todos los agentes que están conectados al chat.

15.3. JADE PROGRAMACIÓN PARA ANDROID

JADE (Java Framework de desarrollo de agente) es un software marco totalmente implementado en lenguaje Java. Simplifica la implementación de sistemas multiagentes a través de un middle-ware que cumple con las especificaciones de la FIPA (Foundation for Intelligent Physical Agents) y a través de un conjunto de herramientas gráficas que soporta las fases de implementación y depuración. La plataforma del agente puede distribuirse a través de máquinas (que tampoco deben compartir el mismo sistema operativo) y la configuración se controla a través de una GUI remota. La configuración se cambia en tiempo de ejecución incluso moviendo agentes de una máquina a otra, cuando sea necesario. JADE está completamente implementado en lenguaje Java y el requisito de sistema mínimo es la versión 1.4 de JAVA (el entorno de tiempo de ejecución o JDK).

La sinergia entre la plataforma JADE y las bibliotecas de salto permite obtener una plataforma de agente FIPA compatible con menos espacio físico y compatibilidad con entornos Java móviles hasta CLDC J2ME MIDP 1.0. Las bibliotecas de salto se han desarrollado con la colaboración del proyecto y pueden descargarse como un complemento de JADE de este mismo sitio web.

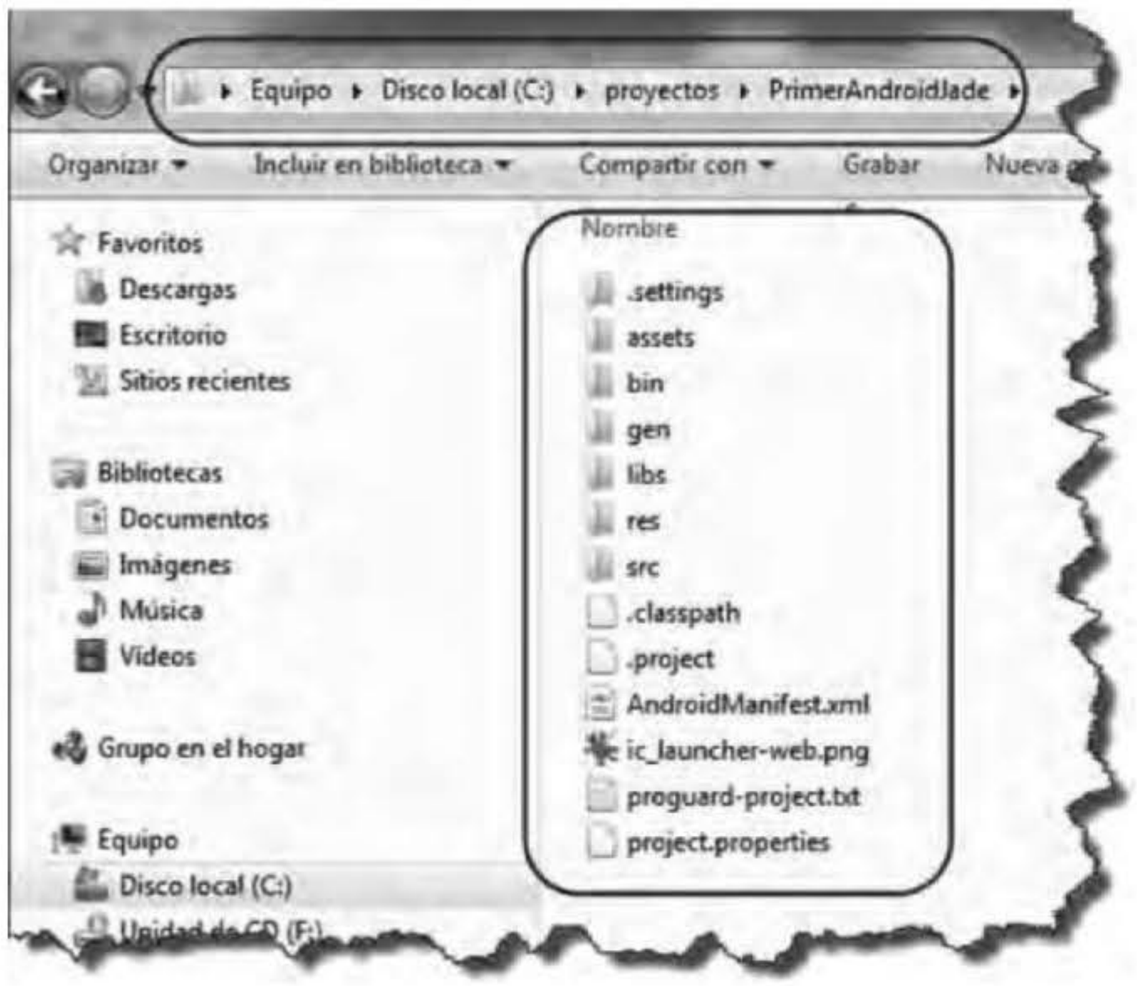
JADE es software libre y se distribuye por Telecom Italia, el titular del copyright, en software open source bajo los términos de la licencia LGPL (menor General público licencia versión 2). Desde mayo de 2003, una placa de JADE ha sido creada esa supervisión de la gestión del proyecto JADE. Actualmente, la junta JADE muestra cinco miembros: Telecom Italia, Motorola, Whitestein Technologies AG, Profactor GmbH y France Telecom R & D.

Paso 1: Descargue el archivo:

LeapAddOn-4.2.0.zip

Pasos 2: Descomprimirlo de:

LeapAddOn-4.2.0.zip

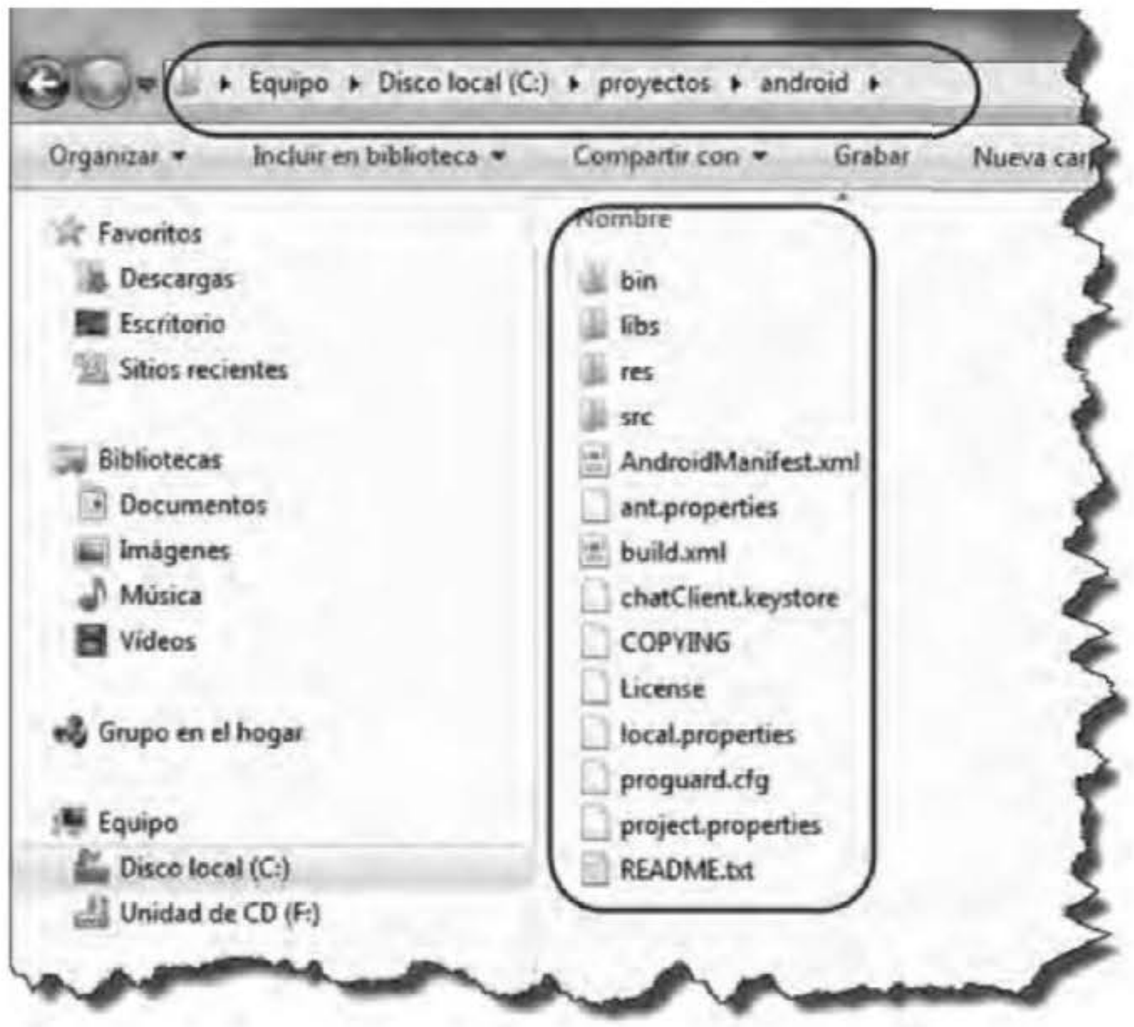


Pasos 3: Descargue el archivo:

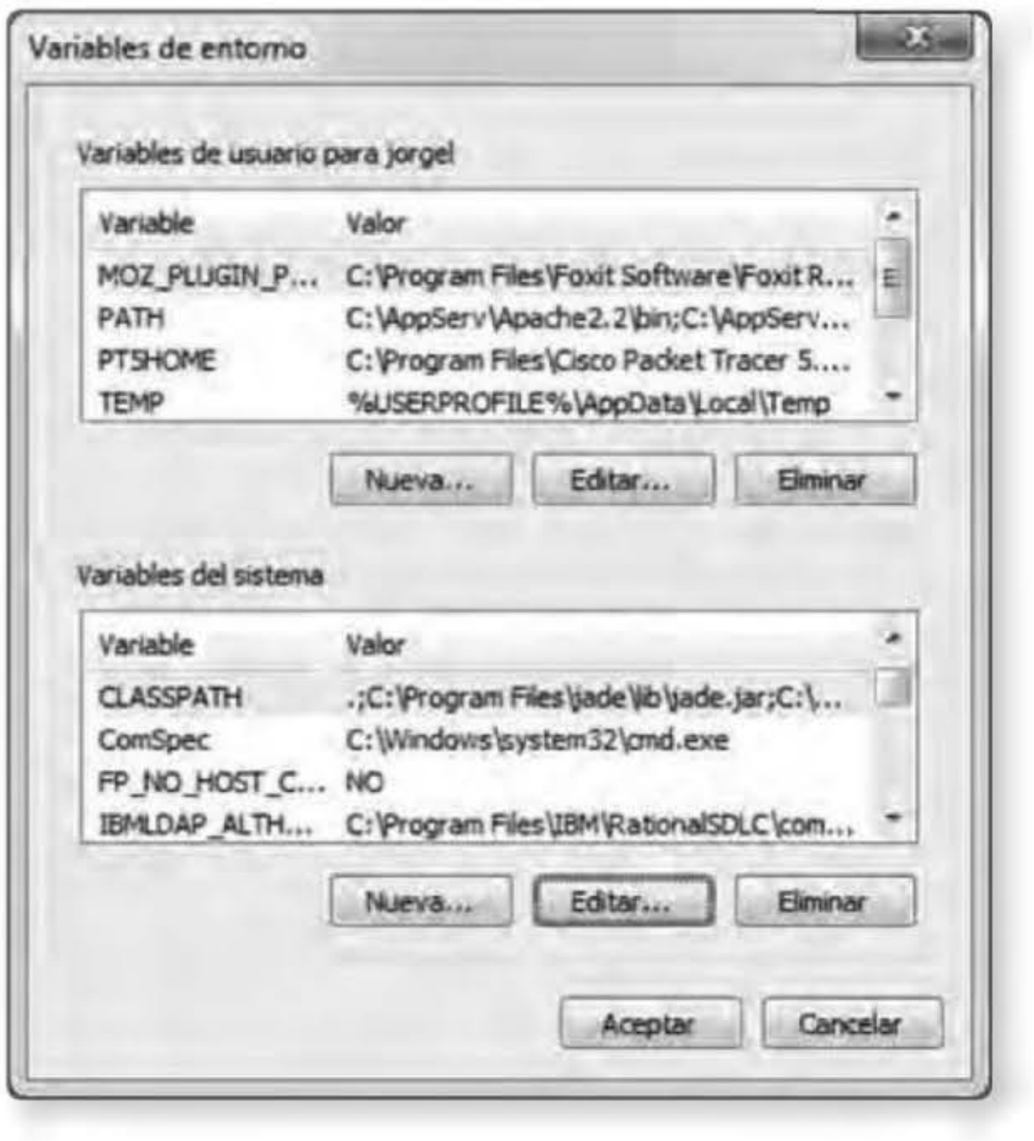
chatAndroid.zip

Pasos 4: Descomprimirlo de:

chatAndroid.zip



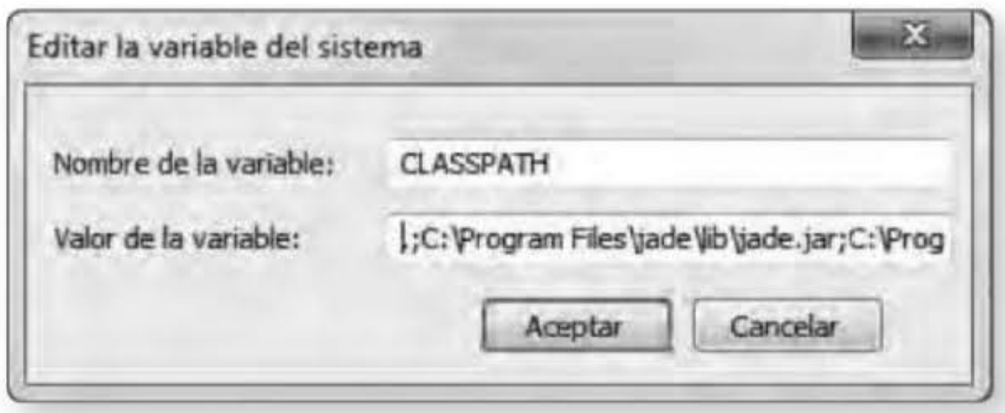
Pasos 5: Configuracion del entorno.



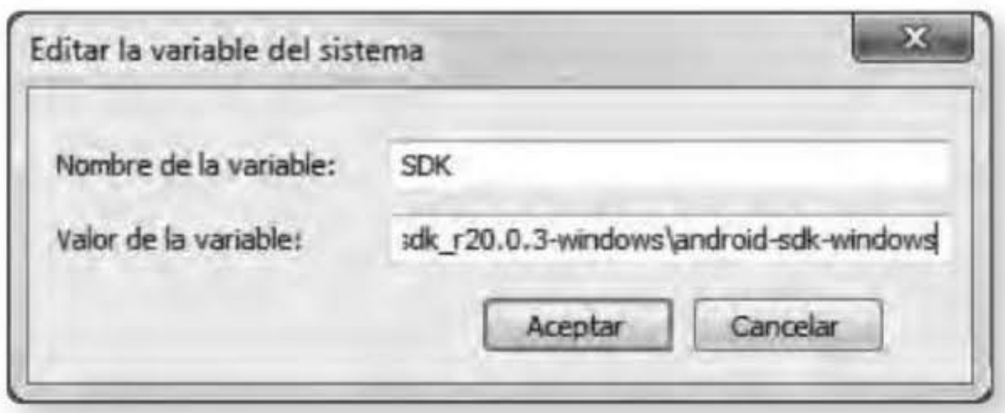
- Ant



- Jade

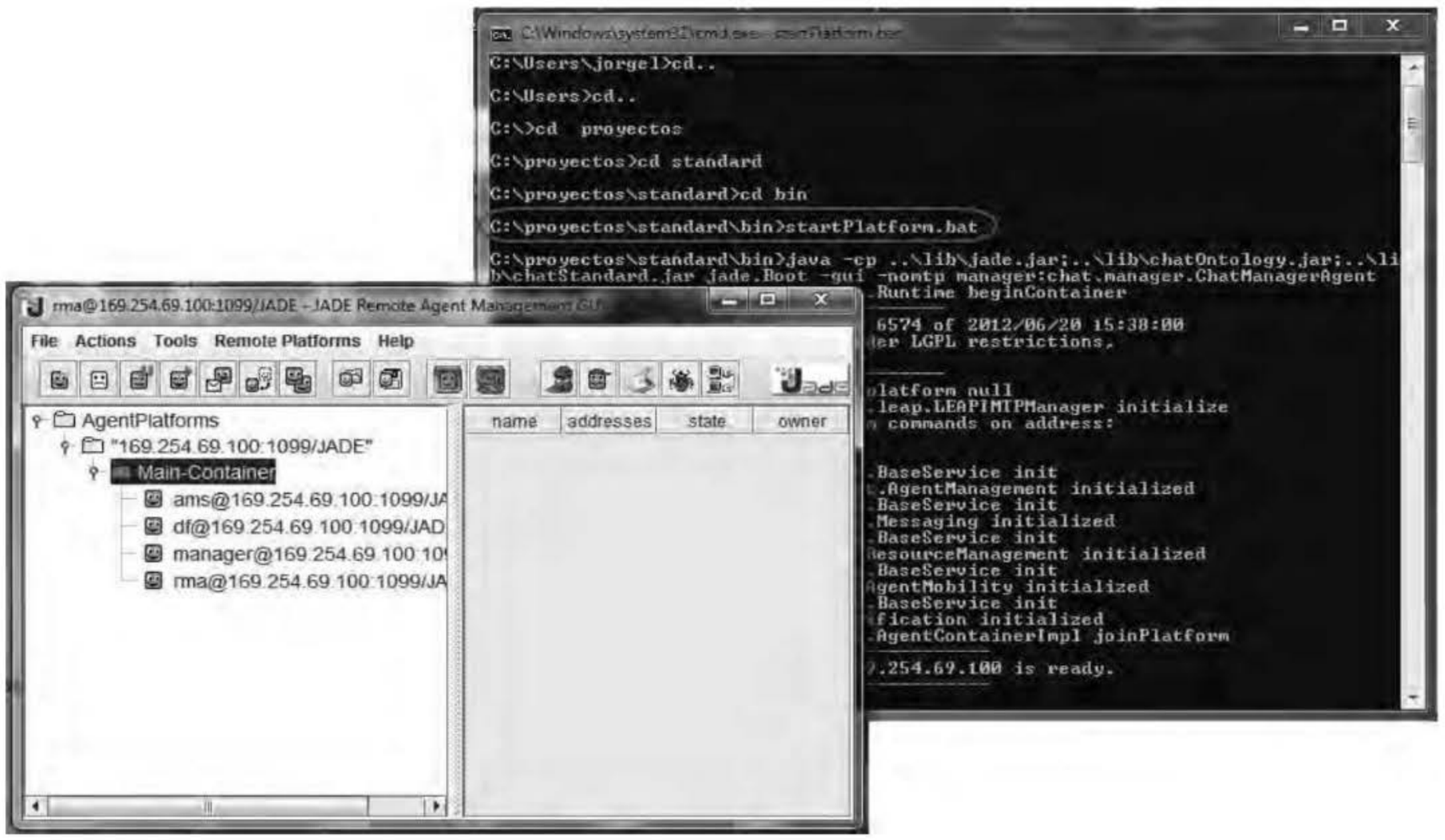


- SDK



Paso 6: Arranque la plataforma Jade.





Paso 7: Inicie el emulador en la versión 2.33.



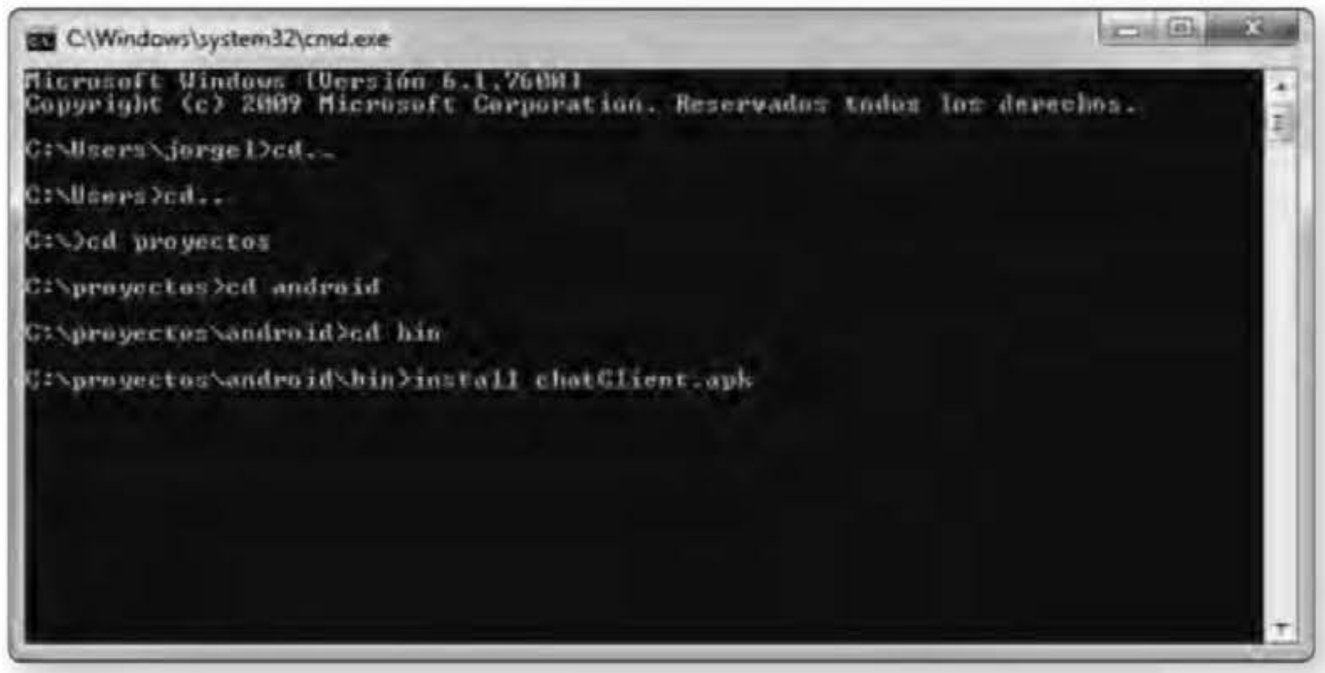
Se observa lo siguiente:



Paso 8: Instale el APK necesario para el chat–cliente.



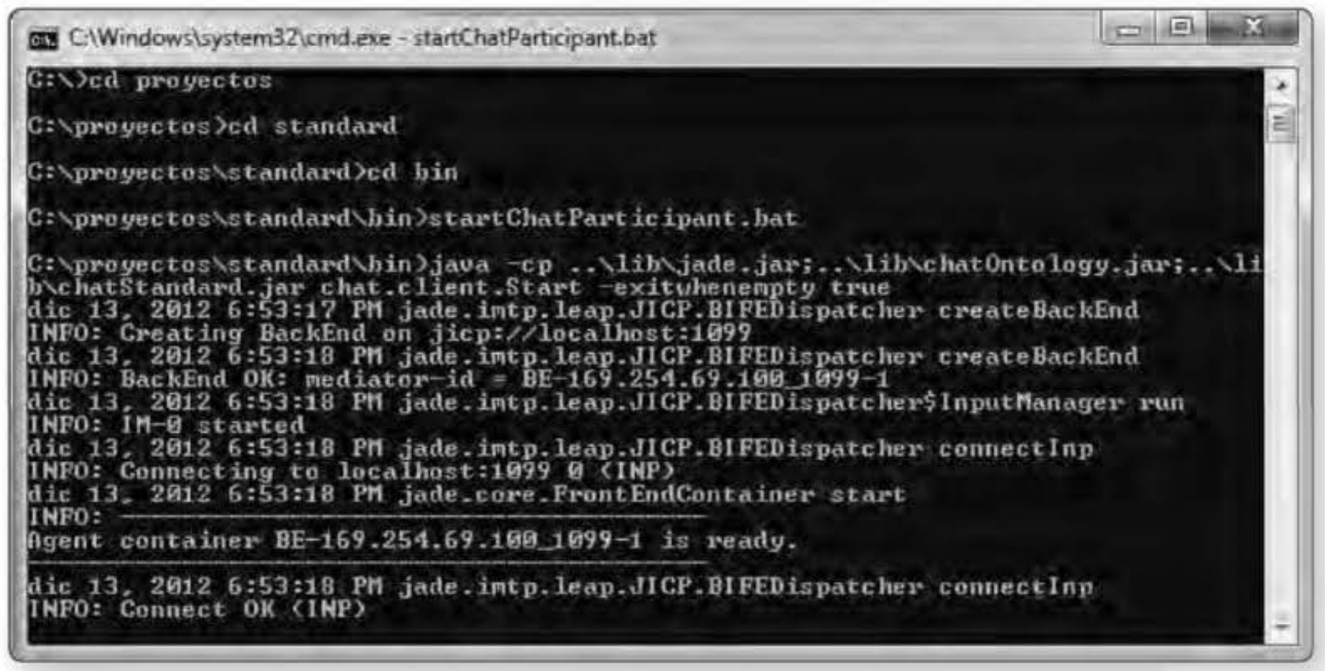
Instalación completa.



Paso 9: Ubique la aplicación *Chat*.



Paso 10: Inicie el chat Estándar.



Paso 11: Establezca comunicación.



15.4. CONCEPTOS BÁSICOS

15.4.1. Web semántica

Razonando sobre la web semántica en el contexto de la historia clínica

Actualmente los datos que se encuentran en la World Wide Web (WWW) carecen de significado y son un conjunto de datos aislados que se encuentran agrupados por medio de etiquetas (tags) de HTML (HiperText Markup Language) las que solo representan instrucciones de formato como: H1, TD, TR, etc. En ese sentido, la semántica, el significado de los datos que nos muestra la Web, es proveído por el ser humano cuando lee la página web. Sin embargo, el lenguaje HTML tiene la etiqueta <META> con la cual se intenta darle algo de semántica a los datos en la Web, ésta se utiliza para añadir información sobre la página y es la que los buscadores como Google consultan para encontrar coincidencias exactas con las palabras que el usuarios pretende encontrar, por lo tanto, su semántica es primitiva (Hebeler et al, 2009).

La web semántica propuesta por Tim Berners-Lee en el 2001 implementa la semántica a través de conexiones estandarizadas entre los datos para crear contexto [es mucho más que un conjunto de palabras claves; esto solo se logrará cuando se establezca relaciones entre ellas; es decir, relaciones entre los datos para crear contexto] y, por lo tanto, significado. En la siguiente construcción HTML:

```
<META name=”author” content=”Research Group”>
<META name=”keywords” content=”paciente”>,
```

¿A qué se refiere la palabra paciente? ¿Se refiere a construir una ontología que se enfoque en datos del paciente o síntomas del paciente o diagnósticos del paciente?, es decir, ¿qué se quiere decir con la palabra paciente? Como vemos, la semántica queda para que el lector humano la interprete. Sin embargo, si las palabras clave se relacionaran a otras palabras clave se formará una red de datos que revela significado. Así, paciente podría relacionarse a muchas otras palabras como médico, enfermera, medicina, etc. y se puede notar que estas relaciones expresan semántica, es decir, significado.

Fuente: Dr. Nestor Adolfo Mamani Macedo
(obtenido de http://www.usmp.edu.pe/ffia/gaceta/gaceta2-1.pdf)

Por ejemplo, lo anterior podría interpretarse como cuando "un médico atiende a un paciente e indica que la enfermera le brinde una medicina". Aunque esta semántica es aún imprecisa, la frase es sólo una de muchas otras interpretaciones que pudieran surgir, otra puede ser "una enfermera recibe un paciente en emergencia quien tomó una cierta medicina y solicita que el médico lo atienda para que realice el diagnostico".

Implementar relaciones dentro de la Web requiere que se formalice una estructura o arreglo de los términos en un estándar de reglas gramaticales específicas y en la creación de un nuevo estándar de términos o elementos de formación primitivos para la descripción de los datos y de los procesos, el vocabulario. Estos dos estándares juntos, la gramática y el vocabulario restricto forman el lenguaje y ayudan a incorporar significado, es decir semántica. Por otro lado, en la medida que la gramática y el vocabulario del lenguaje se expandan a través de nuevas relaciones la semántica se irá enriqueciendo. Los componentes de una aplicación de web semántica se dividen en dos categorías principales:

- componentes y
- herramientas de la web semántica asociadas.

Al respecto, Timothy (Tim) Berners-Lee, el creador de la Web, afirma:

> «La web semántica no es una Web separada sino una extensión de la actual, en la que la información está dada con significado bien definido lo que significa que permite que los computadores y las personas trabajen en cooperación» (Berners-Lee et al, 2001).

En otras palabras, la web semántica es una red de datos descritos y vinculados de manera que se establezca semántica y que se adhiere a una gramática establecida y a constructos o términos de un lenguaje (o vocabulario).

> Conceptualmente, es factible añadir semántica a la Web por medio un conjunto de instrucciones de programación; pero la programación en sí misma se complicaría innecesariamente en una inmensidad de comandos if, case, búsquedas en la base de datos y otras técnicas de programación. (Hebeler et al., 2009).

La web semántica propuesta por Berner-Lee et al. (2001) implementa la semántica a través de conexiones estandarizadas para relacionar información. Así, una persona o un programa puede decir fácilmente si este paciente es lo mismo que otra referencia paciente. Cada dato único se conecta a un contexto más amplio o Web. Esta Web ofrece precisión para las definiciones, relaciones a una jerarquía conceptual, relaciones a información asociada y relaciones a individuos específicos de paciente. La flexibilidad de la Web posibilita conexiones a toda la información necesaria.

Los términos que se relacionan y los caminos que se forman con todas las relaciones dan forma a un vocabulario de dominio, también conocido como *ontologiá*.



Fig. 1. Las palabras claves Juan y Perez en la WWW.

Las figuras 1 y 2 ilustran la diferencia en que se almacena la información en la WWW, como palabras claves independientes y una web semántica de datos. En el ejemplo, se utiliza las palabras claves Juan, Perez.

En la figura 1, se observa cómo en la WWW los datos están aislados y sin conexiones con otros datos limitando su semántica y dejando al lector humano que determine su significado. Un buscador retornará los datos de Juan y Perez cuando están juntos en un único elemento (por ejemplo, una página web) obteniendo resultados limitados o puede dar resultados de los dos datos por separado, en este caso la información no será coherente para el lector.

Leyenda: I →Instancia

En la Figura 2, se ve una ontología en la web semántica, los datos existen relacionados entre sí y, además, interrelacionados con otros datos y definiciones por medio de relaciones; todo este conglomerado de relaciones lógicas dan contexto y significado a las palabras Juan y Perez. De hecho, esto es una ontología de historia clínica dentro de la cual están las palabras claves. Aquí la web semántica permite que, al igual que lo hace un ser humano, una aplicación también pueda seguir estas relaciones y obtener información enriquecida. Un buscador en la web semántica retornará los datos Juan y Perez; pero además dirá su DNI, dirección y fecha de nacimiento; adicionalmente, podrá decir el número de historia clínica y los síntomas que tiene: mareos, fiebre e ictericia. Por otro lado, si el buscador fuera capaz de inferir, también retornará información complementaria, en este caso, es obvio que Juan Perez es un paciente de una institución sanitaria en particular.

Por supuesto, debe existir también un Juan Perez, ingeniero, contador, etc., en cuyo caso, la web semántica retornará a todos los Juan Perez con información acerca de cada uno. Es decir, la web semántica habrá navegado por varias ontologías diferentes y, todo esto, será transparente para el lector humano.

La web semántica alcanza la semántica a través de conexiones estandarizadas a información relacionada de un dominio (ontología), cada elemento de dato se conecta a un contexto mayor en la red, la cual ofrece potenciales caminos para su definición: relaciones a una jerarquía conceptual, relaciones a información asociada y relaciones a instancias específicas. Por lo tanto, las aplicaciones de web semántica siempre usan ontologías, cada una especializada en un área de dominio específica.

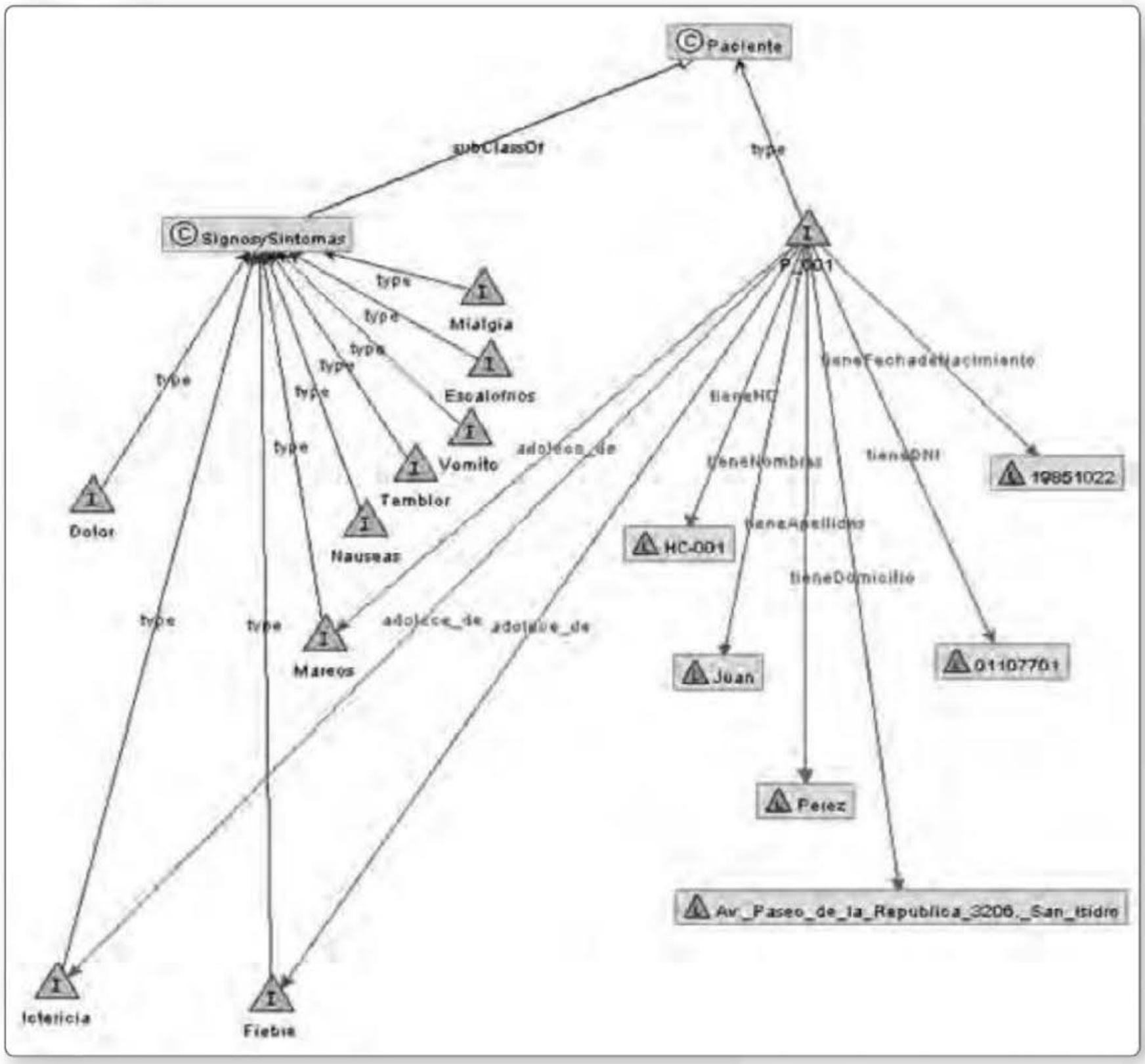


Fig. 2. Las palabras claves Juan y Perez en la web semántica.

15.4.2. Implementar la web semántica



La Web tal como la conocemos, consiste principalmente de contenido estático dirigido al ser humano, quien debe interpretarlo y darle significado; es decir, el usuario infiere la semántica. El contenido de una página web se vincula a otro contenido vía el URL (Universal Resource Locator). El contenido de la WWW no tiene construcciones lógicas formales, en cambio, la web semántica consiste principalmente de declaraciones para consumo de aplicaciones de software; estas declaraciones se vinculan a través de construcciones que pueden formar semántica; así, la semántica del vínculo proporciona un camino significativo definido en vez de uno interpretado por el usuario, tome como ejemplo la declaración Juan amigo de Carlos, es una declaración de asociación que nos dice que Juan es un amigo de Carlos, no un enemigo, ni un desconocido; es más, esta declaración permite inferir que hay una relación entre Juan y Carlos y viceversa.

El elemento fundamental de la web semántica es la declaración. Los diferentes tipos de declaraciones comparten el mismo estándar y describen conceptos, lógica, restricciones e individuos (instancias). El utilizar un mismo estándar posibilita su compartición e integración. En vez de ser un sustituto de la

WWW, la web semántica la extiende con significado mediante el uso de declaraciones estandarizadas para darle semántica y lograr así compartir información dispersa en la Web.

Las relaciones semánticas forman la web semántica. Estas relaciones incluyen definiciones, asociaciones, agregaciones y restricciones. La Figura 3 muestra un grafo de declaraciones. Las declaraciones y sus correspondientes relaciones establecen ambos conceptos (por ejemplo, una persona tiene Nombres y Apellidos) e instancias (por ejemplo, Código de la Persona P-001 con Nombres Juan y Apellidos Perez). Las declaraciones que definen los conceptos y las relaciones entre éstos forman una ontología. Las declaraciones que se refieren a individuos forman datos de instancia.

Las declaraciones pueden ser de dos tipos:

- asertivas
- inferidas

La primera requiere que la aplicación cree directamente la declaración de conceptos o de individuos, para hacerla valer en su momento (líneas solidas). La última requiere de un razonador para inferir declaraciones adicionales desde las declaraciones asertivas (línea verde punteada). Aquí se puede inferir que Juan puede tener un cáncer pancreático debido a que tiene los síntomas.

Note que en este ejemplo, la web semántica, está utilizando una combinación de dos ontologías, la ontología de historia clínica y la ontología de enfermedades. Debido a que Juan Perez tiene, entre sus síntomas, tres en particular: Mareos, Fiebre e Ictericia, que también son los síntomas del cáncer pancreático, es que se puede inferir la enfermedad que afecta a Juan Perez. En la web semántica, esta misma deducción podrá ser efectuada por un aplicativo.

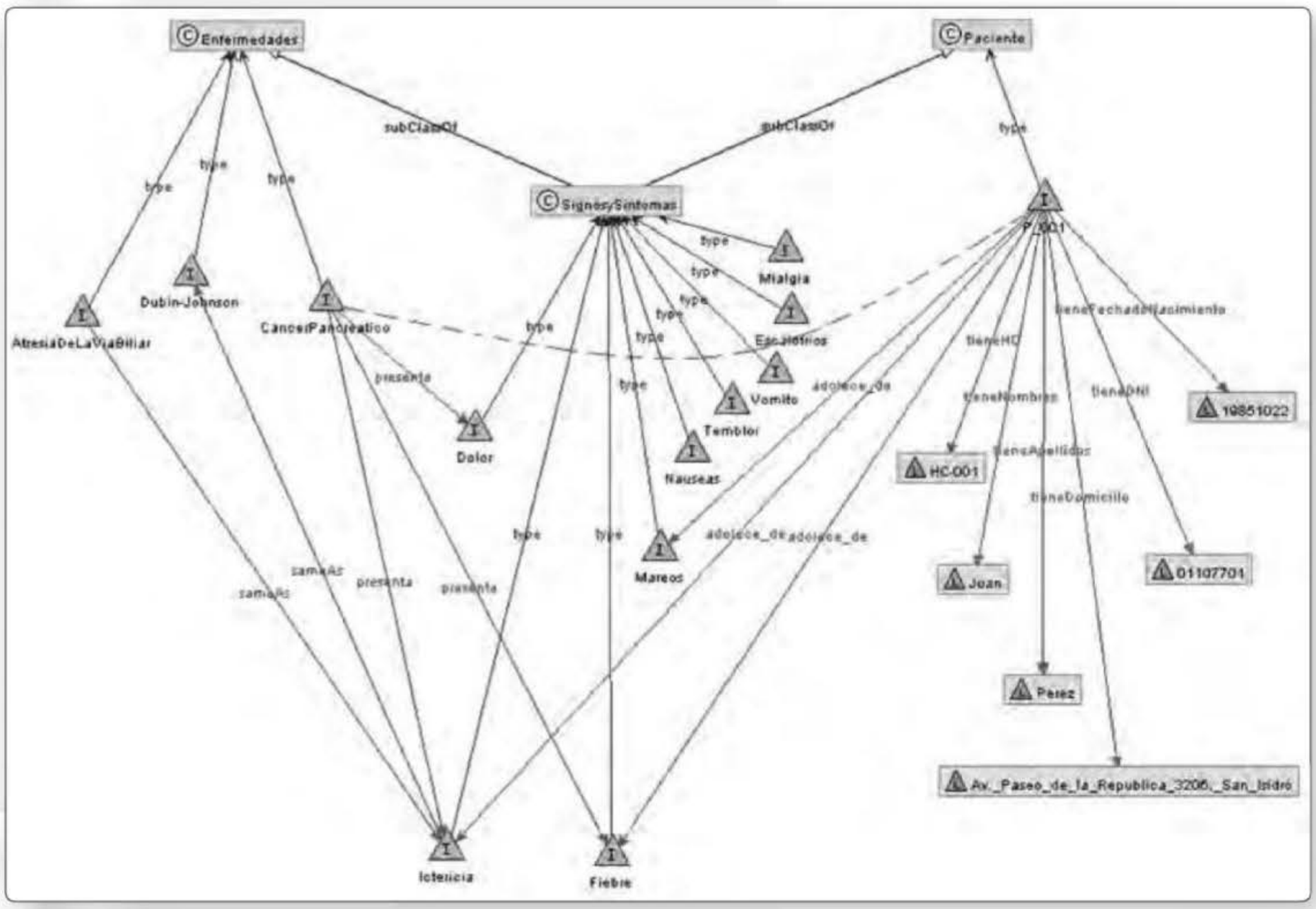


Figura 3 Inferencia en la web semántica

Este es un ejemplo sencillo que solo quiere explicar el proceso interno de la web semántica. En la práctica, este proceso se repetirá constantemente y con muchas ontologías a la vez.

Las declaraciones de web semántica emplean un lenguaje de web semántica (vocabulario + reglas de gramática) para identificar los diversos tipos de declaraciones y relaciones. La web semántica tiene varios lenguajes, que van de básicos hasta complejos. La web semántica existe principalmente en dos formas:

- bases de conocimiento y
- archivos.

Las **bases de conocimiento** ofrecen almacenamiento dinámico y extensible similar a las bases de datos relacionales. Los archivos típicamente contienen declaraciones estáticas. La diferencia es que las bases de conocimiento utilizan herramientas (motor de almacenamiento), los archivos no. La tabla 1 compara las bases de datos relacionales y bases de conocimiento.

Tabla 1: Comparación de bases de datos relacionales y bases de conocimiento

CARACTERÍSTICA	BD RELACIONAL	BASE DE CONOCIMIENTO
Estructura	Esquema	Declaraciones de Ontología
Datos	Filas	Declaraciones de Instancia
Lenguaje de Administración	DDL	Declaraciones de Ontología
Lenguaje de Consulta	SQL	SPARQL
Relaciones	Claves Foráneas	Múltiples Dimensiones
Lógica	Externa a la BD/Triggers	Declaraciones de Lógica Formal
Singularidad	Clave de la tabla	URI

Las **bases de datos relacionales** dependen de un esquema por cada estructura. Una base de conocimiento en el contexto de la web semántica depende de declaraciones de ontología para establecer la estructura. Las bases de datos relacionales están limitadas a un tipo de relación, la *clave foránea*. La web semántica ofrece relaciones y restricciones multidimensionales tales como herencia, parteDe, asociadoCon y muchos otros tipos, incluyendo relaciones lógicas y restricciones.

En las bases de conocimiento el lenguaje utilizado para formar la estructura y las instancias es el mismo lenguaje; pero son diferentes en las bases de datos relacionales. Las bases de datos relacionales ofrecen un lenguaje diferente,

- DDL *(Data Description Language)* para crear el esquema y
- DML *(Data Manipulation Language)* para el mantenimiento de los datos.

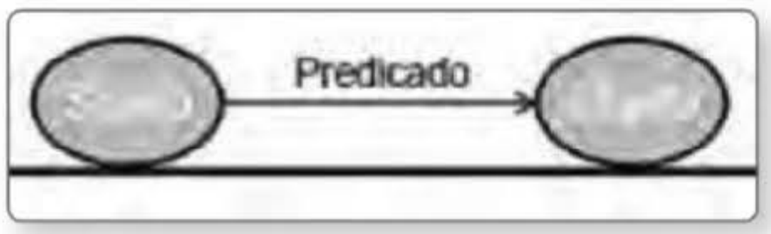
Es decir, en las bases de datos relacionales, añadir una tabla o columna es diferente a añadir una fila.

15.4.3. Componentes de programación

Según Hebeler et al. (2009), los componentes principales de la web semántica son los siguientes:

- *Declaración:* Es la base de la web semántica; se compone de varios elementos que suelen formar una triada o tripla (3-tupla). La triada consta de un
  - sujeto,
  - predicado y
  - objeto (por ejemplo, Juan esdeTipo persona).

Las declaraciones definen la estructura de información, instancias específicas y los límites de esa estructura y se relacionan entre sí para formar la red de datos que constituye la web semántica.



- URI (identificador uniforme de recursos): Proporciona un nombre único para los elementos que figuran en una declaración a través de toda la Internet. Así, cada componente de una declaración (sujeto, predicado y objeto) contiene un URI para afirmar su identidad en toda la WWW. Esto elimina conflictos de nomenclatura, asegura que dos elementos son los mismos o no y también puede proporcionar una ruta de acceso a información adicional. La estructura del URI es la siguiente:

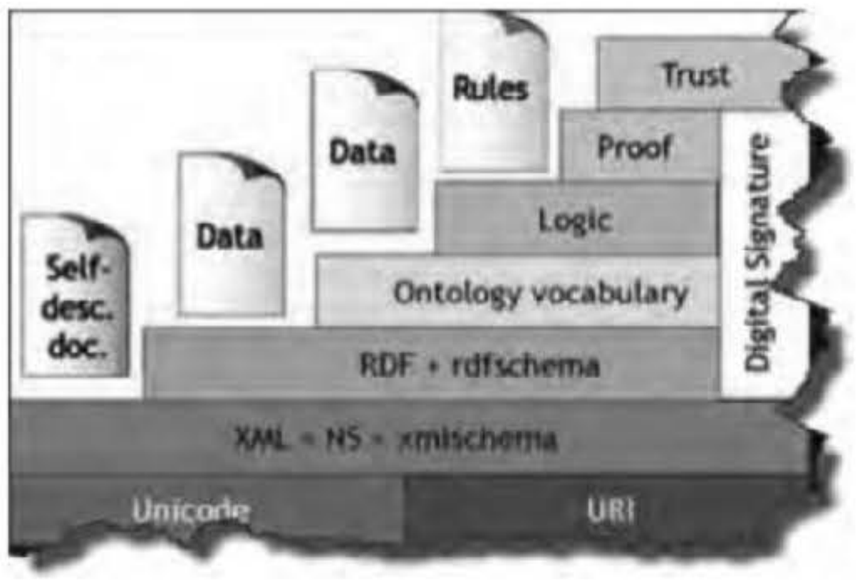
ESQUEMA: PARTE JERARQUICA? SOLICITUD # FRAGMENTO

A diferencia del URL, el URI permite incluir en la dirección una subdirección, determinada por el "fragmento".

- *Esquema:* Nombre que se refiere a una especificación para asignar los identificadores, e.g. urn:, tag:, cid:. En algunos casos también identifica el protocolo de acceso al recurso, por ejemplo "http:", "mailto:", "ftp:".
- *Autoridad:* Elemento jerárquico que identifica la autoridad de nombres (por ejemplo "//es.wikipedia.org").
- *Ruta:* Información usualmente organizada en forma jerárquica que identifica al recurso en el ámbito del esquema URI y la autoridad de nombres (ej. "/wiki/Uniform_Resource_Identifier").
- *Consulta:* Información con estructura no jerárquica (usualmente pares "clave=valor") que identifica al recurso en el ámbito del esquema URI y la autoridad de nombres. El comienzo de este componente se indica mediante el carácter "?".
- *Fragmento:* Permite identificar una parte del recurso principal o vista de una representación del mismo. El comienzo de este componente se indica mediante el carácter "#".

Aunque se acostumbra llamar URL a todas las direcciones web, URI es un identificador más completo y, por eso, es recomendado su uso en lugar de la expresión URL.

Un URI se diferencia de un URL en que permite incluir en la dirección una subdirección, determinada por el "fragmento".



- *Lenguaje:* Las declaraciones se expresan de acuerdo con un lenguaje de web semántica. El lenguaje se compone de un conjunto de palabras claves que proporcionan instrucción a las diversas herramientas de la web semántica. Hay varios lenguajes para elegir que ofrecen diversos grados de complejidad y expresividad semántica.
- *Ontología:* Una ontología consiste de declaraciones que definen conceptos, relaciones y restricciones; es análoga a un esquema de base de datos. La ontología forma un modelo de dominio de información especializada en áreas como finanzas y medicina o generalizada para la descripción de objetos comunes. Actualmente, existen muchas ontologías enriquecidas para incorporar en las aplicaciones que pueden ser usadas directamente o adaptarlas a necesidades específicas.
- *Datos de instancia:* Son las declaraciones que contienen información sobre casos específicos en lugar de un concepto genérico. "Juan" es una instancia, mientras que "persona" es un concepto o clase. Esto es análogo a los objetos/instancias en un programa orientado a objetos. Los datos de instancia constituyen la mayor parte de la web semántica. Una ontología que contiene el concepto persona puede ser utilizada por millones de instancias de persona.

15.4.4. Herramientas

Las herramientas que construyen, manipulan, interrogan y enriquecen la Web Semántica son las siguientes: (Hebeler et al, 2009):

- *Herramientas de construcción:* Permiten construir e integrar una web semántica a través de la creación o importación de las declaraciones de la ontología y las instancias. Hay herramientas basadas en GUI que permiten ver y explorar los datos de la web semántica; por ejemplo, Protégé. Existen, también, varias interfaces de programación de aplicaciones (API) para integrar con los programas como por ejemplo, Jena.
- *Herramientas de interrogación:* Estas herramientas navegan a través de la web semántica para devolver una respuesta solicitada. Existen varios métodos de interrogatorio que van desde la simple navegación gráfica para buscar, hasta un lenguaje de consulta completo; por ejemplo, SPARQL (SPARQL *Protocol and RDF Query Language*).
- *Razonadores:* Los razonadores añaden inferencia a su web semántica. La inferencia crea adiciones lógicas que permiten que los conceptos y las relaciones se relacionen adecuadamente con otros, tal como una "persona" es un *ser vivo*, el "papá" es un *padre*, "casado" es un *tipo de relación* o "casado" es una *relación simétrica*. Hay varios tipos de razonadores que ofrecen distintos niveles de razonamiento; por ejemplo, "Pellet" presenta un alto nivel de razonamiento. Los razonadores aprovechan las declaraciones afirmativas para crear declaraciones auxiliares lógicamente válidas.
- *Motores de reglas:* Los motores de reglas soportan inferencia normalmente más allá de lo que puede deducirse con la lógica descriptiva, como ejemplo está Jess. Las reglas permiten fusionar ontologías y otras tareas de lógica más grandes, incluyendo métodos de programación, tales como

búsquedas de conteo y de cadena. Los motores de reglas son manejados por reglas que se pueden considerar parte de toda la representación del conocimiento. Cada motor de reglas se adhiere a un lenguaje de reglas determinado.

- *Entornos semánticos:* Estos empaquetan las herramientas mencionadas anteriormente para trabajar como una unidad integrada. Existen alternativas de código abierto tanto para un entorno gráfico de desarrollo integrado (IDE) como para una interfaz de programación (API); un ejemplo es el Jena, aunque también hay varios entornos semánticos comerciales excelentes. Esto le permite empezar a programar de inmediato.

15.4.5. Impactos de la web semántica en la programación

Para que las aplicaciones, se toma ventaja de la web semántica y de sus herramientas. Estas deben adaptar sus expectativas e impactos en algunas de las siguientes categorías (Hebeler et al, 2009):

- Centrada en la Web de datos: La aplicación de web semántica debería estar centrada en los datos. Los datos son la clave, no las instrucciones de programación. en la web semántica los datos están enriquecidos a tal punto que se requiere de menos programación.
- Datos semánticos: Una aplicación de web semántica debería poner significado directamente dentro de los datos en vez de ponerlo dentro de las instrucciones de programación o dejarla a la interpretación del usuario. La web semántica ofrece métodos para definir la información, sus relaciones con otras informaciones, su herencia conceptual y su formación lógica.

Ejemplo:

Una historia clínica dentro de la web semántica podría tener las relaciones pacienteDe para asociar a un paciente con el(los) médico(s) con quien(es) se está atendiendo. Ya que las relaciones tienen independencia de los conceptos, una aplicación podría consultar por todos los datos paciente De Carlos Zapata. El resultado sería todos los pacientes del Dr. Carlos Zapata.

- Integración/Compartición de datos: Una aplicación de web semántica debería tratar de accesar y compartir recursos ricos en información a través de la WWW cuando sea apropiado, tomando ventaja de las diversas fuentes de datos ya existentes. La capacidad de intercambiar información aumenta el valor de la información ya que permite a las aplicaciones utilizar la más actualizada en vez de formarla desde cero. Los artefactos de web semántica (ontologías identificadas con URI) son únicos en toda la Internet.
- Datos dinámicos: Una aplicación de web semántica debería permitir cambios dinámicos en tiempo de ejecución a la estructura y contenidos de su información. En la WWW, el cambio es dinámico y continuo. Las aplicaciones de web semántica pueden insertar nuevos datos en cualquier momento. Estos nuevos datos se obtienen a través de la inferencia.

Estos cuatro impactos potencialmente cambian la forma en que se diseña y programa una aplicación y guían su solución a un uso óptimo de la web semántica.

15.4.6. Razonador

Un razonador se guía por una regla de tres (porque, se sigue que, en consecuencia); este vive solo en el presente; el futuro es consecuencia de lo que se realice en el presente. Se encuentra siempre a la espera de las consecuencias que deban seguirse; esto implica que siempre estará ocupado controlando las premisas para que ocurra lo que deba ocurrir. Es un tecnólogo de lo que realiza.

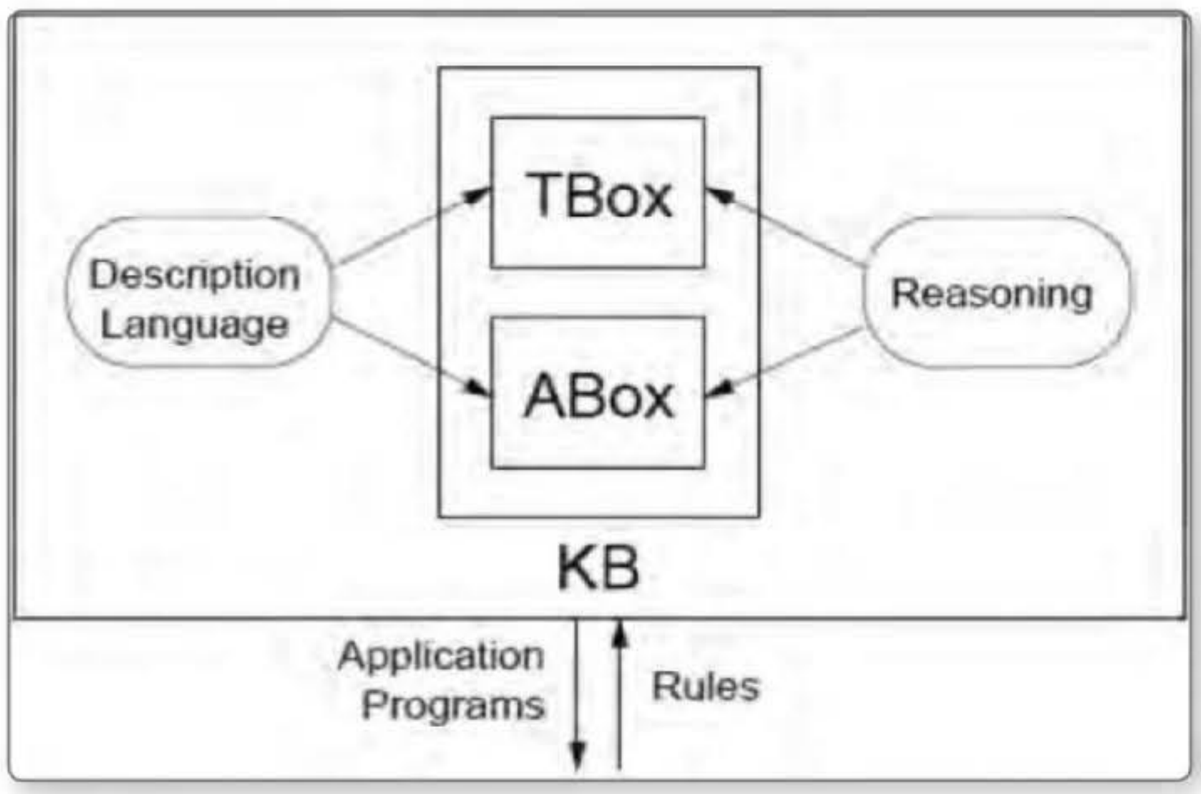
Un razonador orientado a la historia clínica sería aquel que basado en el conocimiento proponga un posible diagnostico; para lograr este propósito se necesita una base de conocimiento (compuesto normalmente por hechos y reglas) y reglas que sean capaces de inferir un nuevo conocimiento.

Existen dos formas de conseguir información de una base de conocimiento.

- Mediante una consulta al modelo de la cual el razonador devolverá la información de las instancias existentes o bien
- Mediante la adición de reglas, donde el razonador; además, de devolver las instancias que resulten de la consulta al modelo, será capaz de generar nuevas instancias que se infieran de las existentes. Se le denomina a este tipo inferencia o razonamiento.

La base del conocimiento (KB) está basada en dos componentes:

- TBox y
- ABox



Figuran 4 Componentes de una base del conocimiento.

El TBox define los dominios y las propiedades que pertenecen a esos dominios y la dependencia jerárquica entre dichos dominios. Los individuos se especifican sobre el componente ABox.
Así, por ejemplo, se definen los siguientes axiomas:

Médico es Persona (Médico $\subseteq$ Persona)

Carlos es Médico (Carlos $\in$ Médico)

Donde el primer axioma es de tipo TBox y el segundo es ABox. De esta información se infiere que Carlos $\in$ Persona.

Una base de conocimiento (TBox y ABox) es equivalente a un conjunto de axiomas de la LPO (lógica de primer orden) y, por tanto, se puede definir un cálculo o sistema de inferencia que permite derivar conocimiento implícito a partir del explícito de la base de conocimiento.

15.4.7. Lógica descriptiva

También llamada lógica de descripción (DL por *Description Logics*) son una familia de lenguajes de representación de conocimiento que describen al dominio en función de conceptos (clases), roles (relaciones entre clases) e individuos (instancias de las clases, pueden relacionarse con un rol).

El nombre de lógica descriptiva nació en la década del 1980. Anteriormente, se le llamaba de la siguiente manera, según orden cronológo:

- Sistemas terminológicos
- Lenguajes de conceptos

Hoy en día, la lógica descriptiva es la piedra angular de la web semántica para su uso en la base de conocimiento o en el diseño de las ontologías.

El primer sistema experto basado en DL fue KL-ONE (por Brachman y Schmolze, 1985). Luego aparecieron otros sistemas como: LOOM, BACK, KRIS, CLASSIC, FACT, RACER, CEL, KAON2, PELLET y otros.

A. Notaciones

- *Lógica ALC (attributive Language):* Conocida como sintaxis alemana debido a sus creadores. Soporta la lógica descriptiva de conceptos, roles e individuos.
- *Sistema SH:* Actualmente, se está implementando para el uso de las ontologías.
- " $\mathcal{SHIQ}$ es $\mathcal{ALCQI}$ + roles transitivos + inclusión roles.
- $\mathcal{SHOIQ}$ es $\mathcal{SHIQ}$ + nominales.
- Se demuestra también que $\mathcal{SHOIQ}$ es $\mathcal{SHOIN}$ extendida con restricciones cualificadas.
- $\mathcal{SHOIN}(\mathcal{D})$ es $\mathcal{ALCIN}$ + nominales + dominios concretos ($\mathcal{D}$)."

  «Aunque extender una lógica con dominios concretos la dota de una expresividad valorada para representar ontologías, fácilmente puede llevar a la indecidibilidad. Verá, sin embargo, que $\mathcal{SHOIN}(\mathcal{D})$ es decidible y es base para el lenguaje de ontología actualmente más aceptado».

DL	PROPIEDAD EXPRESABLE EN LA LÓGICA	COMPLEJIDAD
$\mathcal{ALC}$	lógica de descripción básica	PSPACE
$\mathcal{ALCN}$	+ restricciones numéricas no calificadas $(\mathcal{N})$	PSPACE
$\mathcal{ALC}$ **reg**	+ expresiones regulares sobre roles (reg)	EXPTIME
$\mathcal{ALCI}$ **reg**	+ inverso de roles $(\mathcal{I})$	EXPTIME
$\mathcal{ALCFI}$ **reg**	+ restricciones funcionales sobre roles atómicos $(\mathcal{F})$	EXPTIME
$\mathcal{ALCQI}$ **reg**	+ restricciones numéricas calificadas $(\mathcal{Q})$	EXPTIME
$\mathcal{ALCQO}$ **reg**	+ un alfabeto para los objetos del dominio $(\mathcal{O})$	EXPTIME
$\mathcal{SHIQ}$	$\mathcal{ALCQI}$ + roles transitivos + inclusión roles	EXPTIME
$\mathcal{SHOIN}$	+ restricciones numéricas no calificadas $(\mathcal{N})$	EXPTIME
$\mathcal{SHOIQ}$	$\mathcal{SHIQ}$ + nominales	NEXPTIME
$\mathcal{SHOIN}(\mathcal{D})$	$\mathcal{SHOIN}$ + dominios concretos	EXPTIME

B. *Algunos razonadores para lógica descriptiva*

FaCT++, KAON2, Pellet, RacerPro, etc. determinando que Pellet, de Clark & Parsia es el que más se adecúa al tipo de proyecto. Debido a su condición de open source (facilita la integración y libre modificación del código fuente) así como por la calidad de su documentación, el soporte y las constantes actualizaciones que mantienen activo el proyecto y en continuo crecimiento.

15.4.8. Lenguajes de la web semántica

Desde la década de los 90, se comenzó la investigación de cómo llevar una representación de conocimiento a la *World Wide Web*. En 1997, paralelamente al desarrollo de XML *(Extensible Markup Language)*, apareció RDF *(Resource Description Framework)* como una capa por encima de XML, proporcionando una base común para expresar la semántica, inspirado en PICS, Dublin Core, XML y MCF. Un año después, apareció RDFS *(Resource Description Framework Schema)* se inspiró en los sistemas orientados a objetos y lenguajes como Java. En 1999, la primera especificación RDFS apareció OIL *(Ontology Inference Layer)*.

La *Web-Ontology* (WebOnt) del W3C tomó como punto de partida OIL que es básicamente una ampliación de la definición de RDFS y su evolución DAM+OIL, para la construcción de un nuevo lenguaje con el nombre "OWL", el cual se convirtió en el estándar por la *W3C Recommendation* en febrero de 2004. El cual tomó 19 años de investigación en el área de *Description Logics*.

A. *Componentes de la web semántica*

- *XML* aporta la sintaxis superficial para los documentos estructurados, pero sin dotarles de ninguna restricción sobre el significado.
- *XML* Schema es un lenguaje para definir la estructura de los documentos XML.
- *RDF* es un modelo de datos para los recursos y las relaciones que se puedan establecer entre ellos. Aporta una semántica básica para este modelo de datos que puede representarse mediante XML. Estas expresiones se conocen como tripletas:

- Sujeto identifica un recurso.
- Predicado o propiedades define las relaciones y tipos de valores a un sujeto.
- Objeto también llamado literal o valor.

- *RDF Schema* describe las propiedades y relaciones de un RDF; además, proporciona la semántica para establecer jerarquías entre las relaciones.
- *OWL* aporta más al vocabulario para describir clases y propiedades: ofrece cardinalidad, tipos de relaciones (ej. disyunción), igualdad, tipología de propiedades más complejas, clases enumeradas y otras. OWL se divide en tres tipos:
  - Lite, el más sencillo.
  - DL, evolución del anterior, con mayor expresividad soporta razonadores.
  - Full.

> ***Sobre el acrónimo OWL***
>
> *Alguien podría pensar que el acrónimo correcto para Web Ontology Language debería ser WOL en lugar de OWL. Otros creen que el orden ha sido elegido en honor del personaje Owl de Winnie the Pooh, quien escribe su nombre WOL en lugar de OWL. Realmente, OWL fue propuestocomo un acrónimo que fuese fácilmente pronunciable en inglés, facilitase buenos logos y se relacione con el prestigioso proyecto de representación del conocimiento de los años setenta de Bill Martin One World Language.*
>
> ***Fuente: es.wikipedia.org***

B. Implementar un razonador en la web semántica orientada a historias clínicas

Como se ha visto, la **OWL** es una base de conocimiento la cual deriva de un RDF; al cual se le puede añadir razonadores para obtener una inferencia a partir de una consulta, reglas o ambas.

Para el proyecto de historias clínicas se optó por usar el razonador Pellet. Para lograr esta integración de un razonador con una base de conocimiento se necesita de algún framework que lo permita entre los cuales se tiene los siguientes:

FRAMEWORK	LENGUAJE INTERFAZ	NIVEL SEMÁNTICA	DESCRIPCIÓN
Jena	Java	RDF to OWL 2	Ampliamente utilizado marco de la Web Semántica para Java. Proporciona una interfaz de SPARQL, RDF y API OWL, y el apoyo de inferencia. Permite almacenar múltiples y mecanismos de razonamiento y permite la integración de los mecanismos de costumbre. http://jena.sourceforge.net.
Sesame	Java &RESTful web service	RDF	Proporciona una SPARQL interfaz y una interfaz de servidor HTTP. Se suministra con almacenamiento y múltiples mecanismos de razonamiento y también permite la integración de los mecanismos de medida. http://www.openrdf.org.

FRAMEWORK	LENGUAJE INTERFAZ	NIVEL SEMÁNTICA	DESCRIPCIÓN
OWL API	Java	OWL 2	La OWL API se basa en la sintaxis funcional de OWL 2 y contiene un interfaz común para muchos razonadores. http://owlapi.sourceforge.net.
RAP-RDF API	PHP	RDF	RAP es una fuente abierta RDF API y una suite de software para almacenar, consulta y manipulación de RDF en PHP. http://sourceforge.net/projects/dfapi-php.
Redland	C, Python, Ruby, Perl & PHP	RDF	Colección de las bibliotecas RDF para C, con enlaces para los demás idiomas. Proporciona RDF API, analizadores, y las interfaces de consulta. http://librdf.org.
linqToRDF	.NET	RDF	El marco de Semántica Web para él .NET creado en la Microsoft Consulta de Lenguaje-Integrado (LINQ) Marco (Consulta independiente del lenguaje y el sistema de procesamiento de datos). http://code.google.com/p/linqtordf.

Debido a que Jena puede manejar OWL y OWL2 y permite añadirle un razonador para nuestro caso Pellet y obtener las inferencias; además se podrá introducir a un agente inteligente la base de conocimiento OWL y añadirle las inferencias obtenidas del modelo a este.

15.4.9. Pellet

Razonador basado en tableros ha sido programado en Java, es gratuito y open source. Originalmente desarrollado por el laboratorio Mindswap de Maryland. Se puede utilizar fácilmente desde las librerías de gestión de ontologías (como la API de Jena y OWL) y ofrece una interfaz DIG.

Soporta razonamiento con la expresividad total de OWL-DL (sintaxis SHOIN(D) y SHIN(D)) incluyendo razonamiento acerca de los nominales (clases enumeradas), soporta especificaciones OWL2(SROIQ(D)); también brinda razonamiento con las proporciones inversas de OWL-Full.

Pellet comprueba la consistencia incremental y la clasifica. Están basados en los algoritmos Tableaux desarrollados para lógicas expresivas potentes, la cual valida ontologías.

A. Arquitectura de Pellet

Comprueba la consistencia de una KB (base de conocimiento); es decir, un par de ABox y un TBox. El razonador se acopla con un datatype de oracle para poder chequear la coherencia de las conjunciones de un *XML Schema simple datatype*.

Las ontologías OWL se cargan en el razonador la cual valida y repara la ontología. Este paso asegura que todos tengan un tipo apropiado de tripletas (un requerimiento de la OWL DL pero no de la OWL FULL) y si faltan algunas declaraciones de tipo se añaden usando algunas heurísticas. Durante la fase de carga axiomas sobre las clases (subclase, la clase equivalente o axiomas disyunción) se ponen en el componente TBox y afirmaciones acerca de los individuos (tipo y afirmación propiedad) se almacenan en el componente ABox.Axiomas TBox para pasar por el proceso previo del estándar de razonadores DL antes de que se alimenten al razonador de tablas.

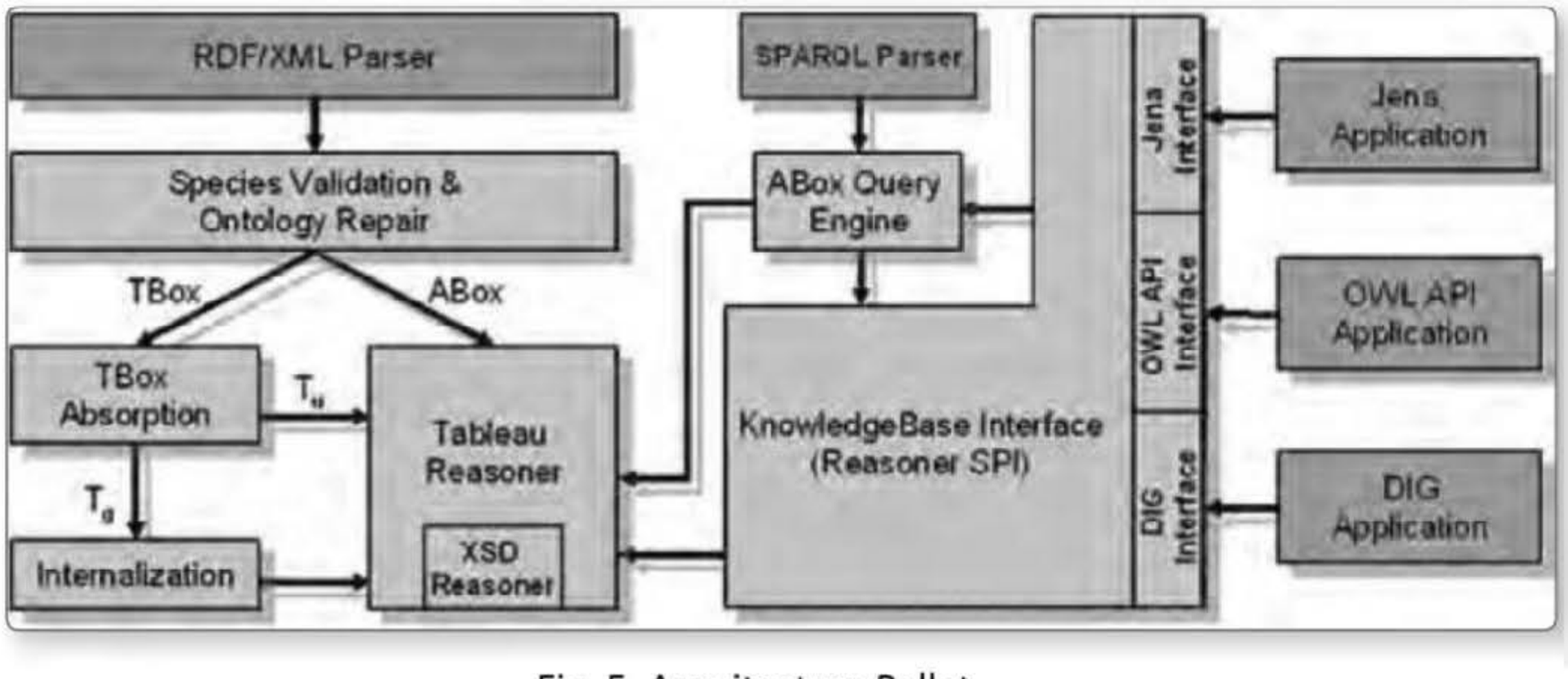


Fig. 5. Arquitectura Pellet

La función de un razonador de tablas es en esencia el control de la satisfacibilidad de una ABox xon respecto a un TBox. Las demás tareas de razonamiento se reducen a la consistencia KB la cual es testeada con una adecuada transformación. El sistema de programación de interface (SPI) de Pellet ofrece funciones genéricas, las cuales administran tales transformaciones. La interfaz de KB se basa como el resto de los componentes internos en la librería Aterm (termino de anotaciones a corto plazo). Este consiste en un tipo de datos abstractos creado para el intercambio de datos de estructuras en forma de árbol entre las aplicaciones distribuidas. Esta librería permite maximizar el subtermino sharing (compartir) y la recolección automática de basura por lo que es muy adecuado para la representación de las expresiones de la clase del OWL. Los términos de la función sharing reduce el consumo de memoria total dedicado para el almacenamiento de concepto de expresiones y facilita transformar los datos de Pellet SPI para una API externa.

15.5. JENA

Jena es un framework de Java para la construcción de aplicaciones en la web semántica. Es de código abierto y ha sido desarrollado a partir de trabajar con *HP Labs Semantic Web Programme*.

15.5.1. Características

Permite manipular ontologías (añadir hechos, borrarlos y editarlos), realizar consultas contra ellas y almacenar. Los recursos no están ligados estáticamente a una clase Java particular.
Los principales problemas que presentan son los siguientes:

- El recurso no cambió, pero la clase Java para modelarlo es otra.
- No se puede modificar la clase dinámicamente.

Jena considera que la abstracción Java del recurso es sólo una vista del mismo. Se compone de:

- API de procesamiento RDF.
- API de procesamiento OWL, DAML y RDF Schema.
- Un motor de razonamiento basado en reglas.
- Un motor de consultas SPARQL, RDQL.
- El soporte es proporcionado por jena-users@incubator.apache.org.

15.5.2. Procesamiento de la API RDF y OWL

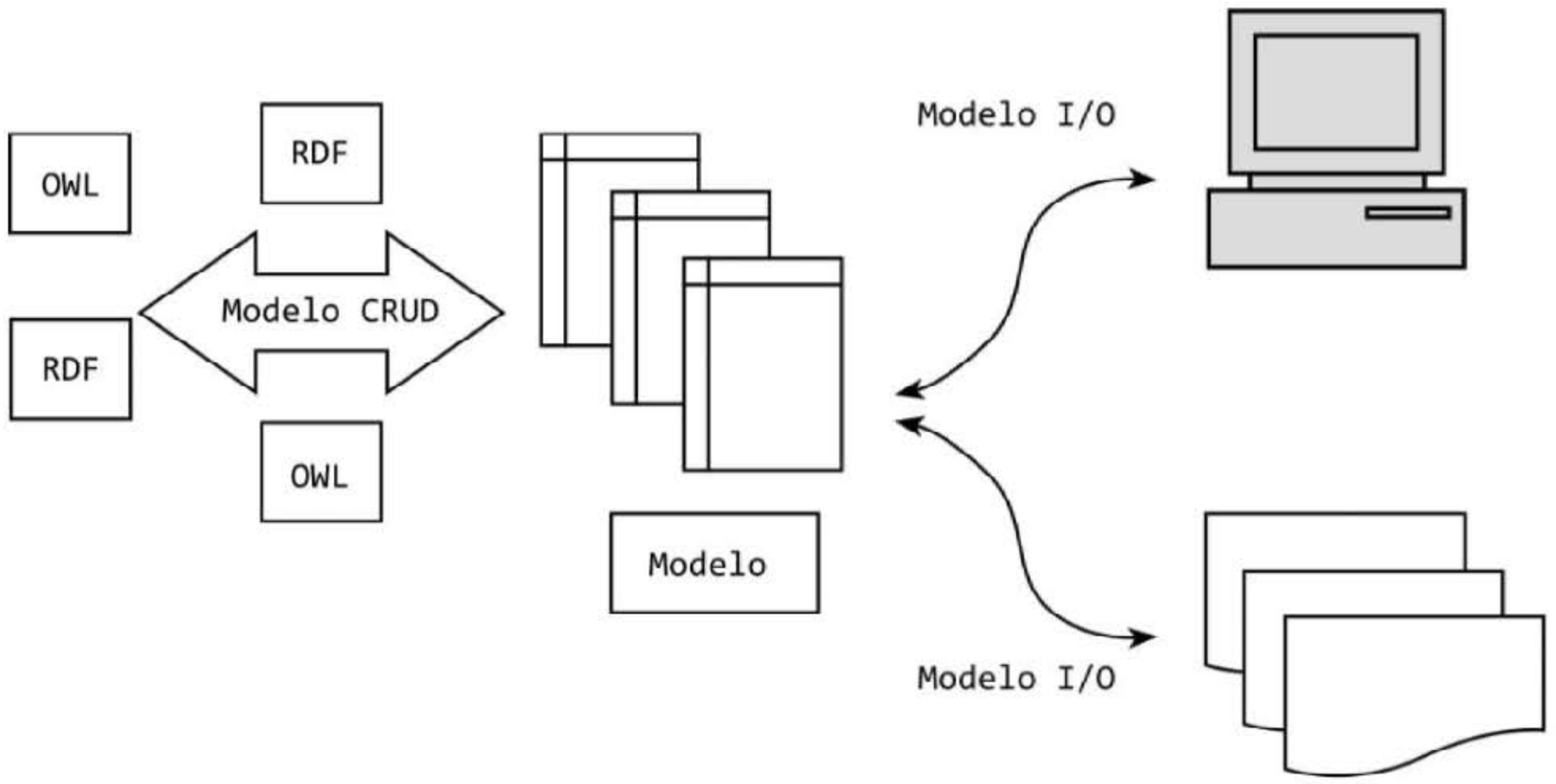


Fig. 6. Jena API RDF y OWL

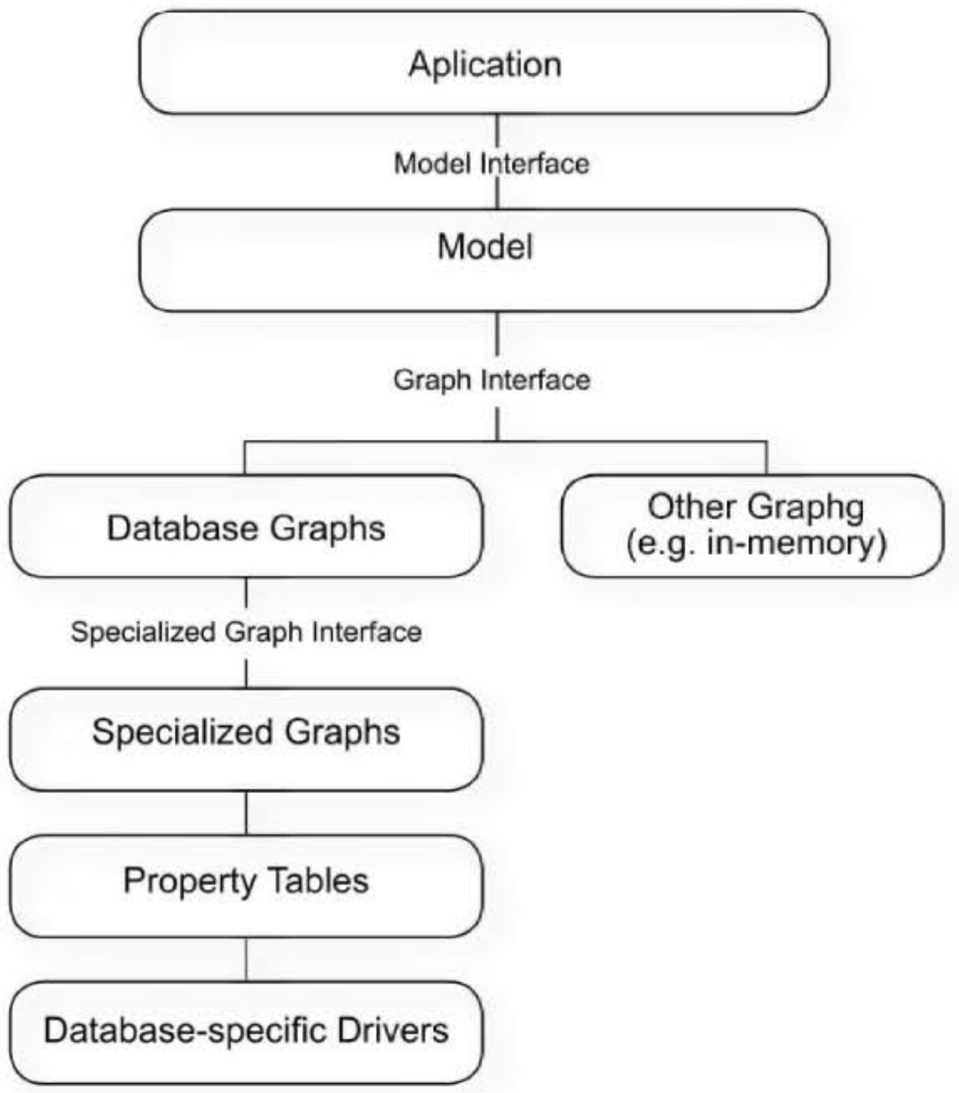


Fig. 7. Descripción de la arquitectura Jena.

El grafico detalla cómo es que Jena procesa una base de conocimiento:

Paso 1: Carga o crea una ontología.

Paso 2: Almacena la información en un modelo que se almacena en memoria y es a este al cual uno puede manipular.

Paso 3: Una vez realizada todas las acciones, alternativamente guarda un archivo físico.

15.5.3. Mecanismo de inferencia en Jena

La inferencia se da mediante un razonador que puede provenir del mismo Jena u otro como Pellet. Estos se deben ejecutar al momento de realizar algún cambio, inserción o consulta a la base de conocimiento.

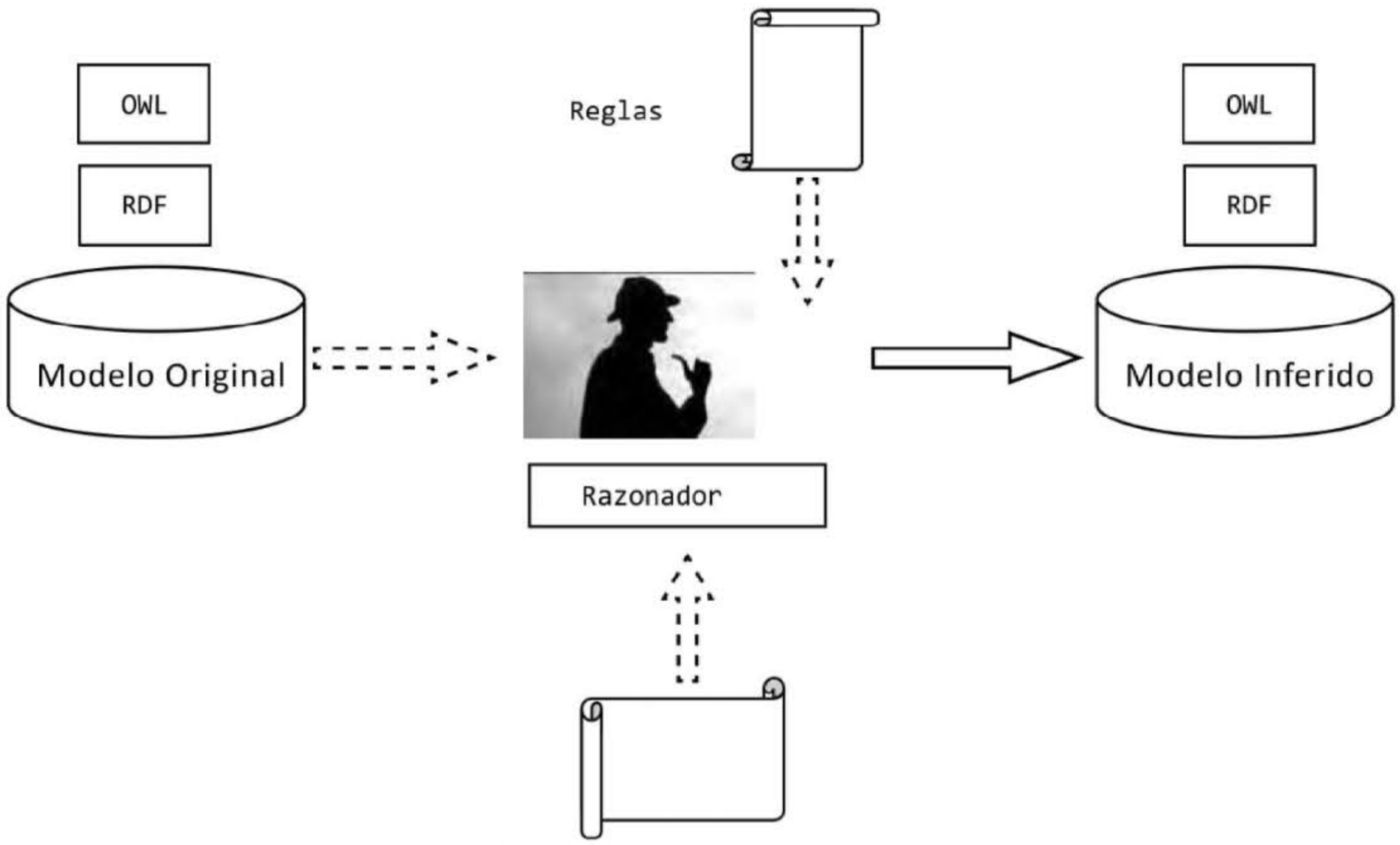


Fig. 8. Jena y razonador

15.5.4. Consultas SPARQL (estándar de la W3C) en Jena

SPARQL se basa en anteriores lenguajes de consulta RDF como rdfDB, RDQL y SeRQL. Se trata de una herramienta que permite hacer consultas a una base de conocimiento como la OWL, el cual ha sido elaborado por la comunidad de la W3C. Técnicamente la consulta SPARQL se basa en un conjunto de tripletas (sujeto, predicado y objeto) o (ID, propiedad y valor). Al respecto, Tim Berners Lee señala lo siguiente:

Pretender usar la web semántica sin SPARQL es como pretender usar una base de datos relacional sin SQL", "SPARQL hace posible consultar información desde bases de datos y otros orígenes de datos en sus estados primitivos a través de la Web.

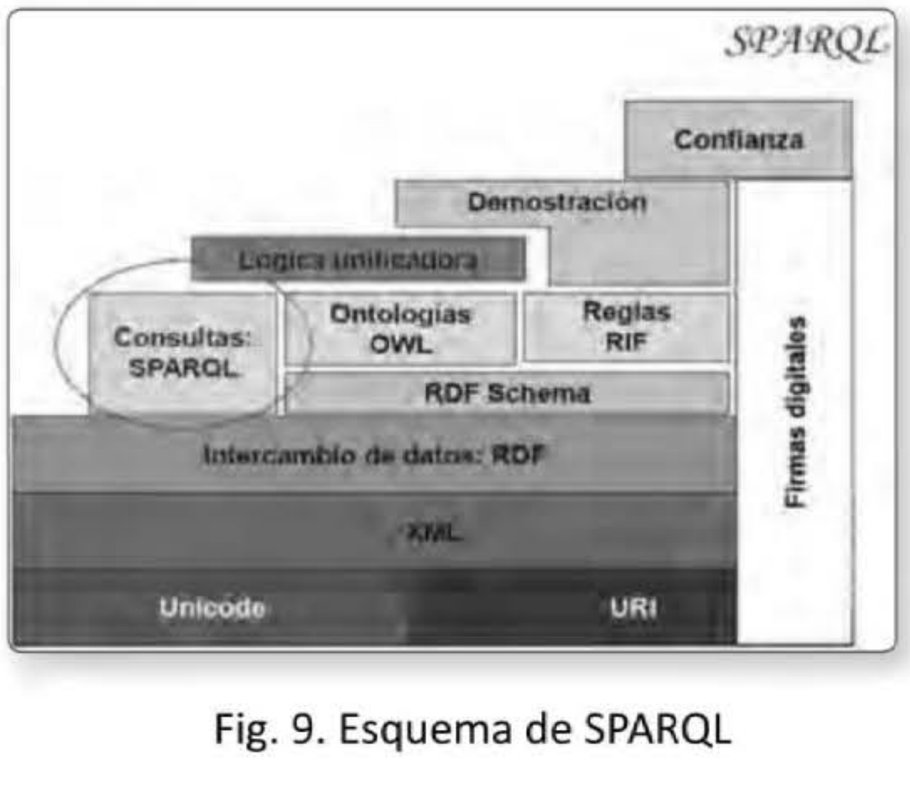


Fig. 9. Esquema de SPARQL

A. *Sintaxis*

- URIs entre < >

  <http://www.misitio.com>

- Prefijos para espacios de nombres (o Name Spaces)

  PREFIX x: <http://www.hca.pe/>
  x:doctor

- Nodos anónimos

  _:nombre ó [ ]

- Literales entre " "

  "Jesus fernandez"
  "234"^^xsd:integer

- Variables empiezan con "?" o "$"

  ?nombre
  $apellido

- Comentarios empiezan con #

  # esto es un comentario

B. *Como lenguaje de consulta posee las siguientes cláusulas*

- PREFIX: Inserta una etiqueta a un URI de la ontología.
- WHERE: Es un proceso secuencial, del cual se obtienen los valores que contendrán esas variables.
- SELECT: Aquí se declaran las variables a mostrar las cuales siguen el patrón "sujeto, predicado y valor". En el siguiente ejemplo, se muestra cómo se listan todos los sujetos con sus propiedades y valores correspondientes.

```
select * where {
?s ?p ?o. }
```

- FILTER: Añade restricciones a los valores de las variables encontradas, utiliza funciones y operadores de Xpath 2.0.
- REGEX: Se utiliza para restringir valores de tipo texto o string.
  - regex(?Expresión, ?Patrón [, ?Flags])
  - ?Expresión: variable a tratar.
  - ?Patrón: patrón de la restricción.
  - ?Flags: opciones para el encaje.

```
PREFIX e: < http://www.misitio.com#>
SELECT ?n ?e WHERE {
?x e:nombre ?n .
?x e:edad ?e
FILTER regex(?n,"A","i")
}
```

- CONSTRUCT: Permite construir un grafo RDF a partir de una consulta.
- DESCRIBE: Devuelve una descripción del grafo en RDF.
- ASK: Devuelve falso o verdad a una determinada búsqueda.

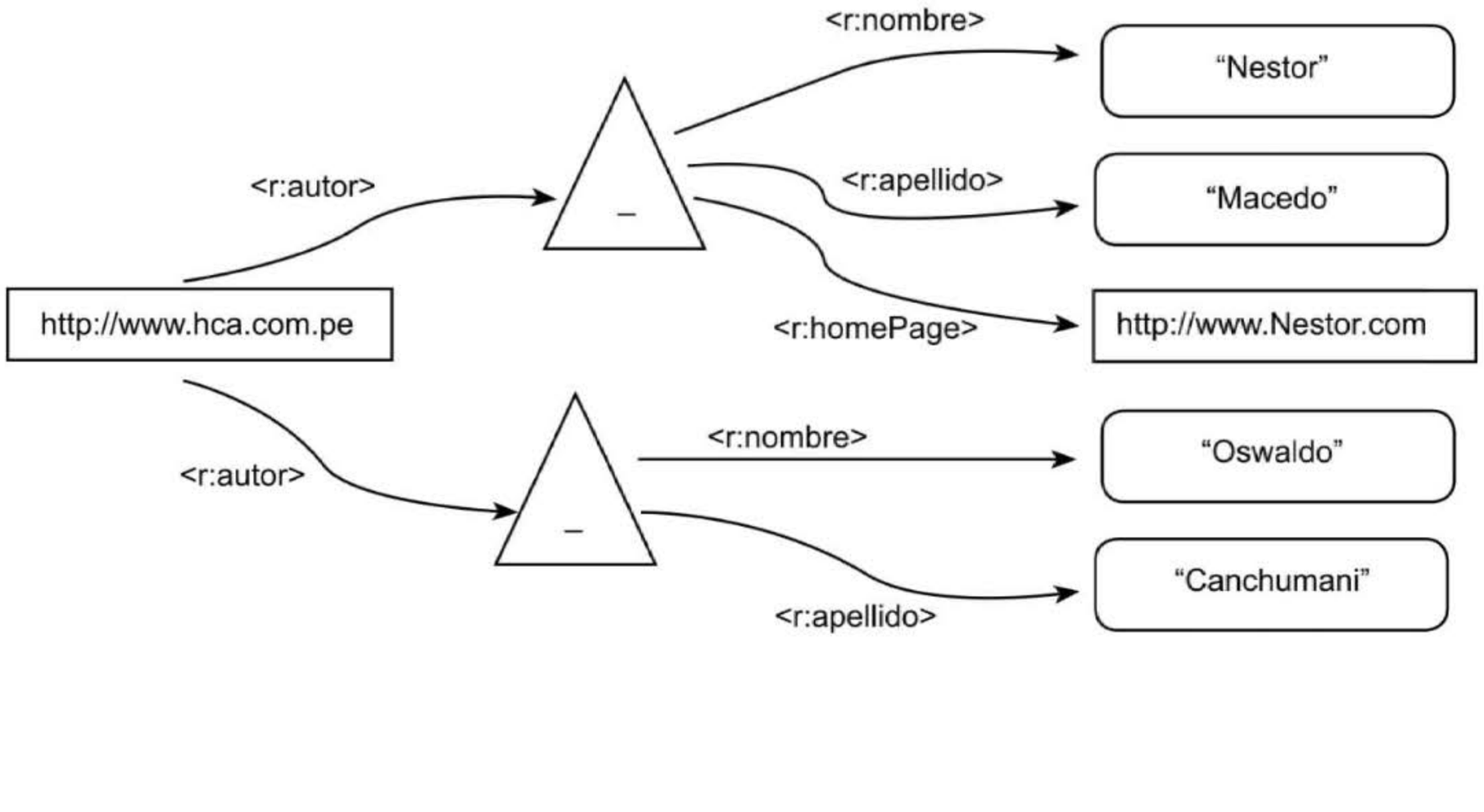
```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX r: < http://www.misitio.com#>
CONSTRUCT { ?x r:nombre ?n }
WHERE { ?x foaf:name ?n }
```

- OPTIONAL: Muestra los valores de las tripletas sin fallar aun cuando estas no existan.
- UNION: Junta dos o más grafos.

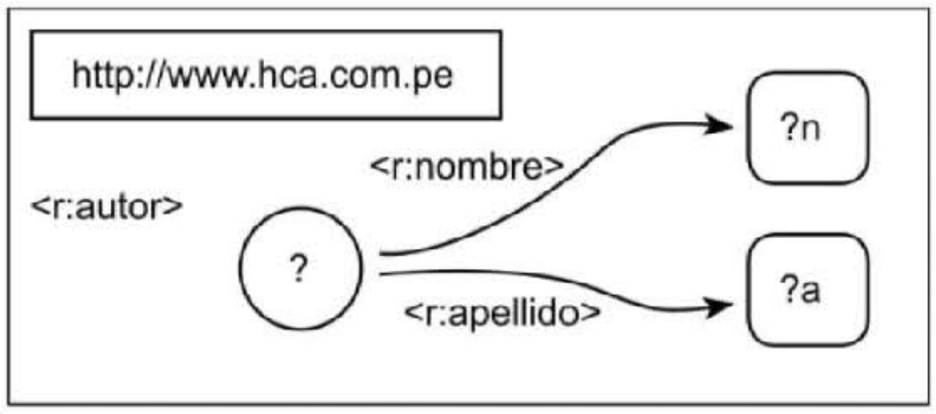
```
prefix e: < http://www.misitio.com #>.
prefix foaf: <http://xmlns.com/foaf/0.1/>.
SELECT ?n
WHERE {
{ ?x foaf:name ?n }
UNION{ ?y p:nombre ?n }
}
```

- DISTINCT: Muestra solo los valores diferentes.
- ODER BY: Ordena la variable de forma descendente o ascendente.
- LIMIT: Limita la cantidad de valores a mostrar.

C. Se basa en encaje de patrones en el grafo



```
PREFIX r: <http://www.ministeriosalud.gob.pe>
SELECT ?n ?a
WHERE {http://www.hca.com.pe r:autor ?x.
?x r:nombre ?n.
?x r:apellidos ?a.}
```



D. Los resultados se obtienen de la cláusula WHERE de la siguiente manera

["?d" variable que toma los valores del dominio los cuales pueden ser cualquiera de la tripleta, está determinada por la siguiente instrucción. No siempre es una variable] ["prefijo"]:["propiedad/clase/instancia"] ["?r" variable que toma los valores del rango y al igual que el dominio puede ser cualquiera de la tripleta pero está determinada por la instrucción anterior. No siempre es una variable] ["." El punto nos indica que sigue una instrucción en la línea siguiente y también sirve para terminar la clausula WHERE]

- **RESULTADOS:**

?n	?a
"Nestor"	"Macedo"
"Oswaldo"	"Canchumani"

- **ALGUNOS EJEMPLOS**

- Lista todos los atributos DataType Property

```
prefix owl: <http://www.w3.org/2002/07/owl#>
select $dp
where {
 $dp a owl:DatatypeProperty
}
```

- Listar todas las clases RDF

```
select ?class
where {
 [] a ?class
}
```

- Listar todas las propiedades RDF

```
select ?prop
where {
 [] ?prop []
}
```

- Listar todas las clases OWL

```
prefix owl: <http://www.w3.org/2002/07/owl#>
select ?class
where {
 ?class a owl:Class
}
```

- Listar todas las clases con sus individuos (sujetos), propiedades y valores

```
SELECT DISTINCT ?clases ?id ?propiedad ?valor
where {?clases a owl:Class.
?id rdf:type ?clases.
?id ?propiedad ?valor.
?d a owl:ObjectProperty.
?s a owl:DatatypeProperty;
FILTER(?propiedad = ?d || ?propiedad = ?s)
}
```

EJECUTANDO LA APLICACIÓN

Siguiente ejemplo es una ejecución de una consulta en el framework JENA:

```
import com.hp.hpl.jena.ontology.Individual;
import com.hp.hpl.jena.query.* ;
import com.hp.hpl.jena.ontology.OntModel;
import com.hp.hpl.jena.rdf.model.Model;

public class Main {
  private static OntModel model; //consultar capitulo 7.10
  private static String NS;
  public static void main(String[] args) {
```

```
String queryString =""
+ "PREFIX owl: <http://www.w3.org/2002/07/owl#>"
+ "PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>"
+ "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>"

+ "PREFIX base:<http://www.owl-ontologies.com/Ontology1282851039.owl#>"

+ "SELECT ?s"
+ " WHERE{?s rdf:type base:Profesor}";
```

- Listar todas las clases OWL

```
prefix owl: <http://www.w3.org/2002/07/owl#>
select ?class
where {
 ?class a owl:Class
}
```

- Listar todas las clases con sus individuos (sujetos), propiedades y valores

```
SELECT DISTINCT ?clases ?id ?propiedad ?valor
where {?clases a owl:Class.
?id rdf:type ?clases.
?id ?propiedad ?valor.
?d a owl:ObjectProperty.
?s a owl:DatatypeProperty;
FILTER(?propiedad = ?d || ?propiedad = ?s)
}
```

EJECUTANDO LA APLICACIÓN

Siguiente ejemplo es una ejecución de una consulta en el framework JENA:

```
import com.hp.hpl.jena.ontology.Individual;
import com.hp.hpl.jena.query.* ;
import com.hp.hpl.jena.ontology.OntModel;
import com.hp.hpl.jena.rdf.model.Model;

public class Main {
  private static OntModel model; //consultar capitulo 7.10
  private static String NS;
  public static void main(String[] args) {
```

```
String queryString =""
+ "PREFIX owl: <http://www.w3.org/2002/07/owl#>"
+ "PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>"
+ "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>"

+ "PREFIX base:<http://www.owl-ontologies.com/Ontology1282851039.owl#>"

+ "SELECT ?s"
+ " WHERE{?s rdf:type base:Profesor}";
```

```
model = ModelFactory.createOntologyModel( OntModelSpec.OWL_DL_MEM );

//Establecemos el NameSpace por defecto para nuestra ontología
 NS = "http://www.owl-ontologies.com/Ontology1282851039.owl#";

//Si se desea se crea un prefijo en el modelo ontológico de la siguiente manera.
//model.setNsPrefix(NS, "http://www.owl-ontologies.com/Ontology1282851039.owl#");

    //abrir el archivo con la ontología
    java.io.InputStream in = FileManager.get().open( "datos/profesor.owl" );
    if (in == null) {
      throw new IllegalArgumentException("Archivo no encontrado");
    }
    // Lee el archivo OWL y lo carga al modelo
    model.read(in, "");

  Query query = QueryFactory.create(queryString);
// Ejecutar la consulta y obtener los resultados
  QueryExecution qe = QueryExecutionFactory.create(query, model);

    try {
      ResultSet results = qe.execSelect();
      ResultSetFormatter.out(System.out, results, query) ;
    } finally { qe.close() ; }
}
}
```

El ejemplo anterior muestra todos los individuos de la clase profesor.

```
| s       |
====================
| base:Profesor_3 |
| base:Profesor_1 |
| base:Profesor_2 |
```

Nota:

Es conveniente crear los prefijos básicos para realizar cualquier consulta tales como:

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: http://www.w3.org/2000/01/rdf-schema#
```

Luego cree el prefijo "base", por ejemplo, que constituye el URL de su página u ontologiá.

E. Grafo de RDF

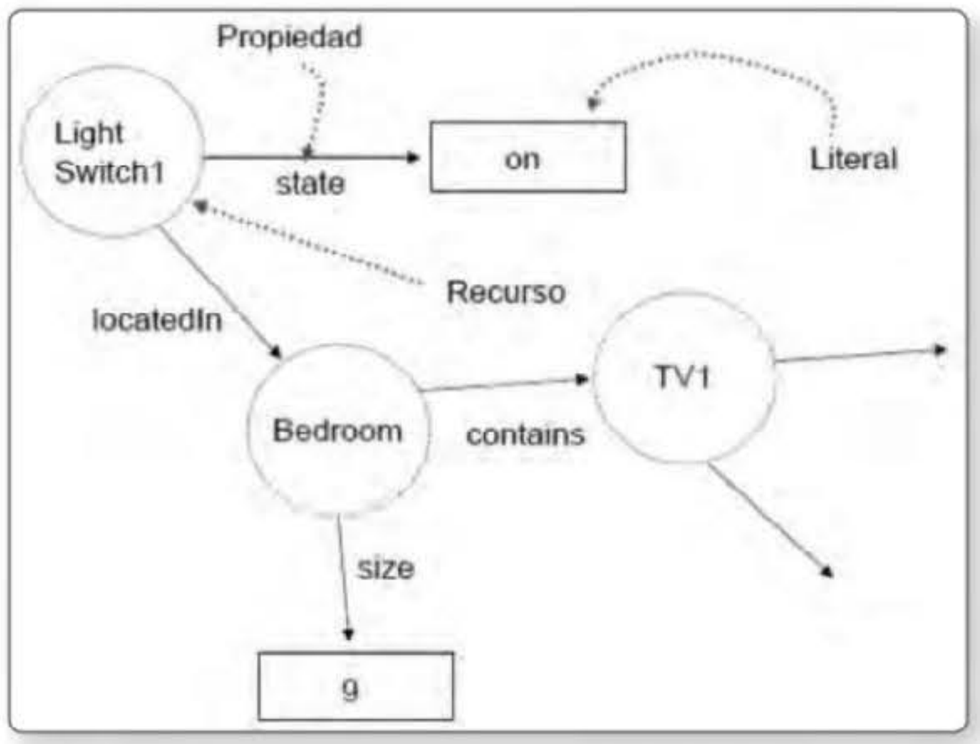


Fig. 10. Jena API RDF.

Se basa en la tripleta ya mencionada (recurso, propiedad, valor/literal).

F. Abstracciones claves

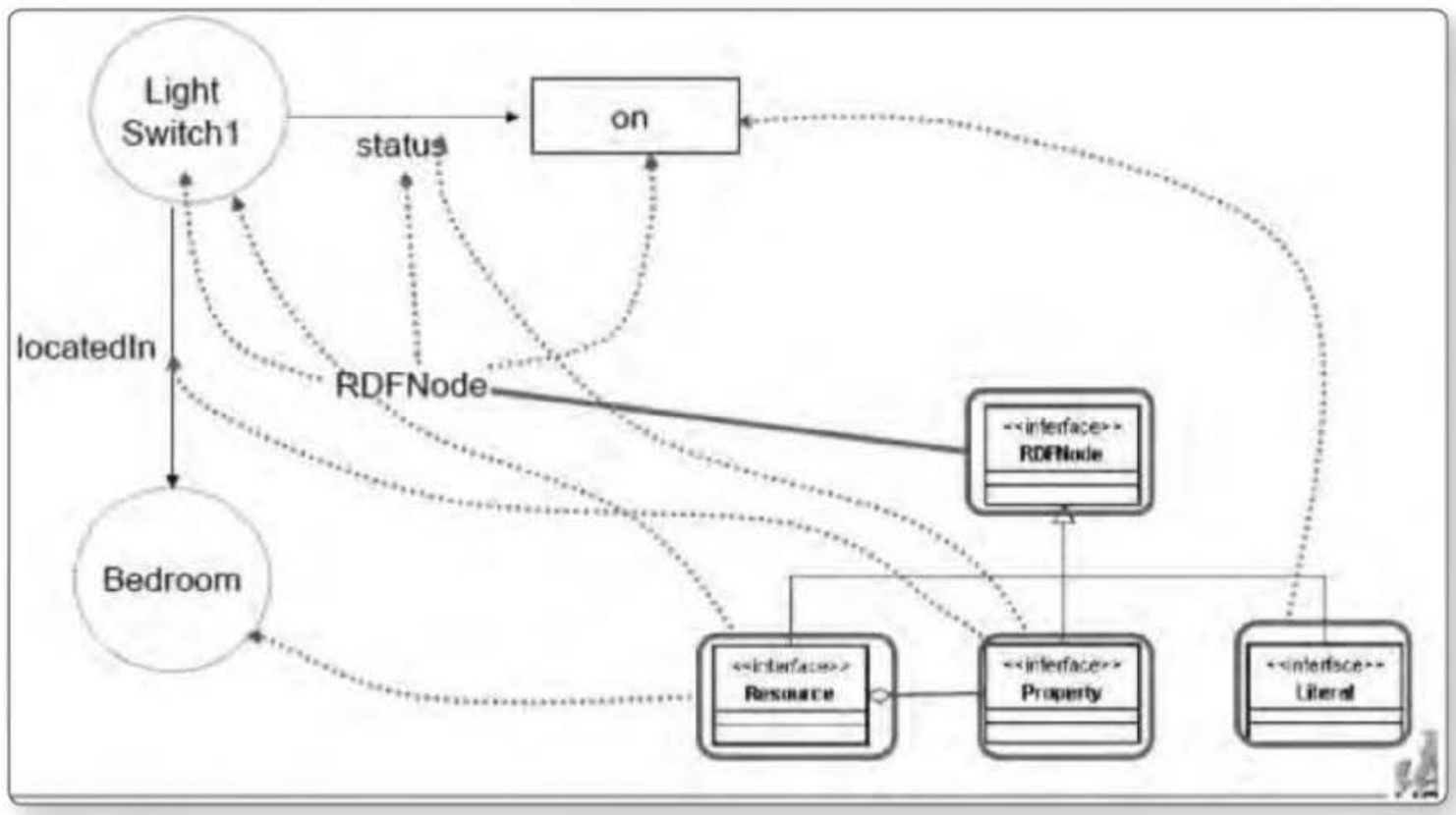


Fig. 11. Jena API RDF Abstracciones claves

G. Escenario típico de uso

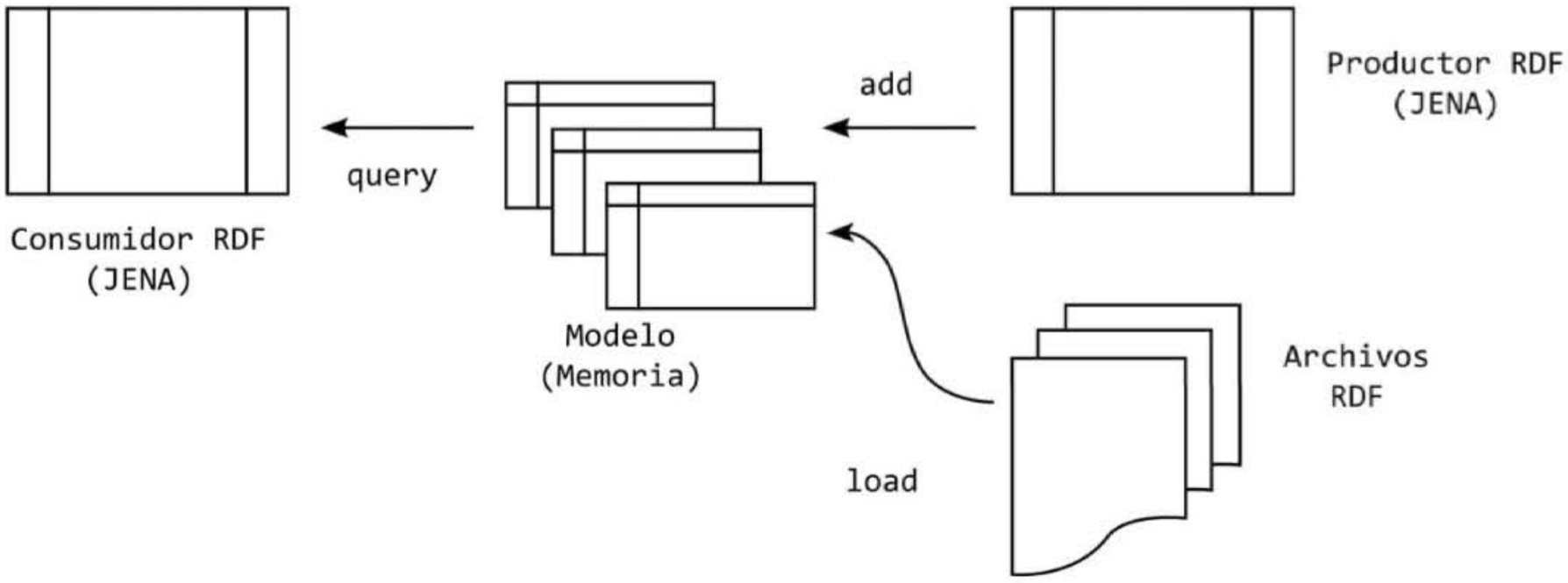


Fig. 12. Jena Escenario típico de uso.

Se trata de un modelo en memoria; donde el usuario realiza consultas y modificaciones, en un entorno de escritorio sin uso de la Web.

H. Escenario avanzado de uso

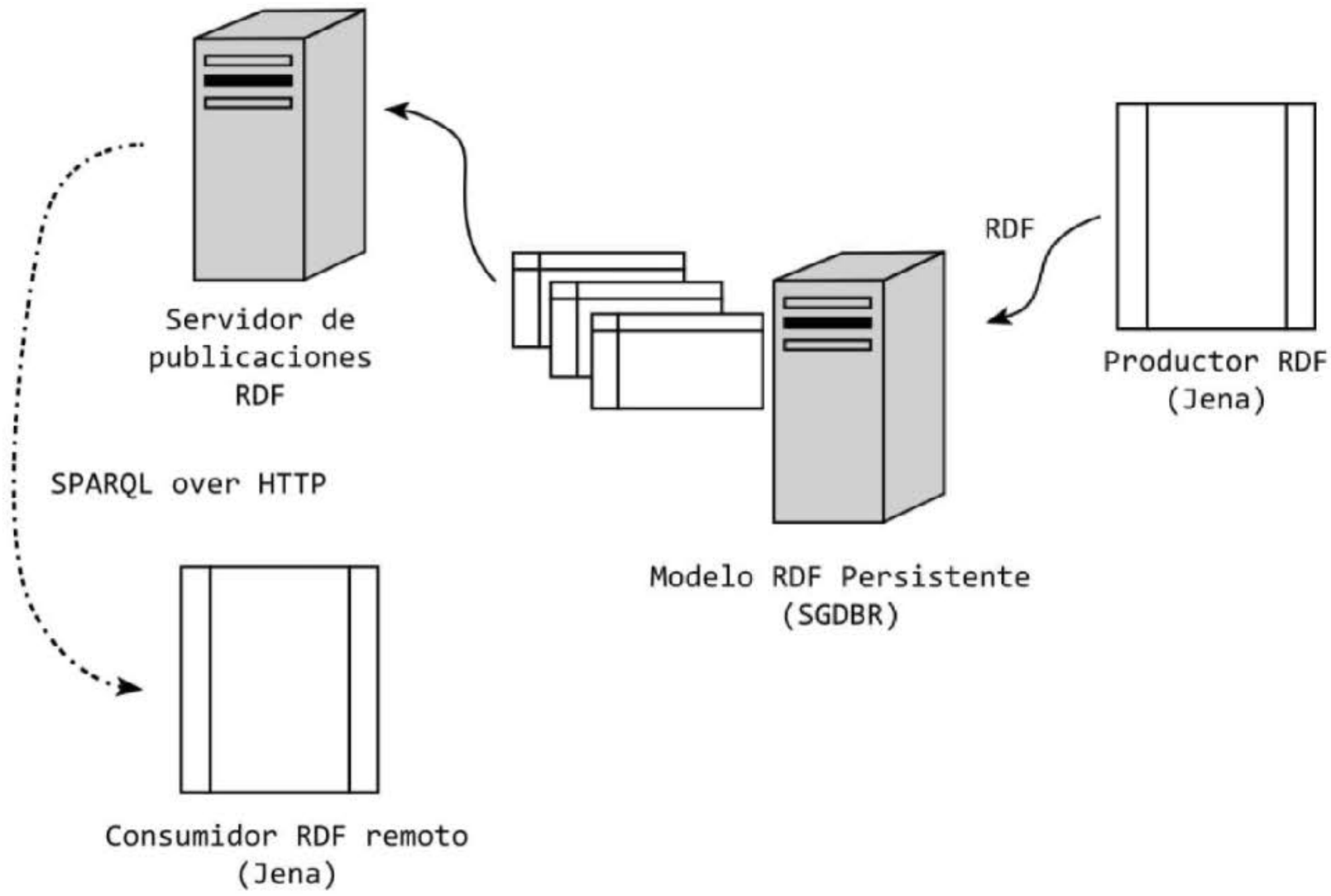


Fig. 13. Jena escenario avanzado de uso.

En este modelo, se genera una persistencia del modelo ontológico; el cual es publicado en un servidor que se accede de forma remota.

I. Creación de RDF

Existen 2 formas:

- Crearlo a través del API de Jena.
- Al cargar un modelo existente.
  - Desde un archivo.
  - Desde un modelo persistente.

API Jena RDF creará un Statement de RDF.

```
Model model = ModelFactory.createDefaultModel();
Resource id = model.createResource("http://www.pacis.org#Nestor");
Property edad = model.createProperty(Constantes.NATURALLY_OCCURRING_NS + "edad");
Literal id_edad= model.createLiteral("42");
yangtze.addProperty(edad, id_edad);
model.write(System.out,"N-TRIPLE");
```

Tendría la forma (Nestor, edad, 42) donde:

- Néstor sería el individuo, instancia, ID.
- Edad seria una propiedad, predicado.
- 42 seria valor, objeto.

CARGAR UN ARCHIVO RDF EXTERNO

```
model.read("file:ubicación/archivo.rdf","RDF/XML");
model.write(new FileWriter("ubicación/archivo.rdf"),"RDF/XML");
```

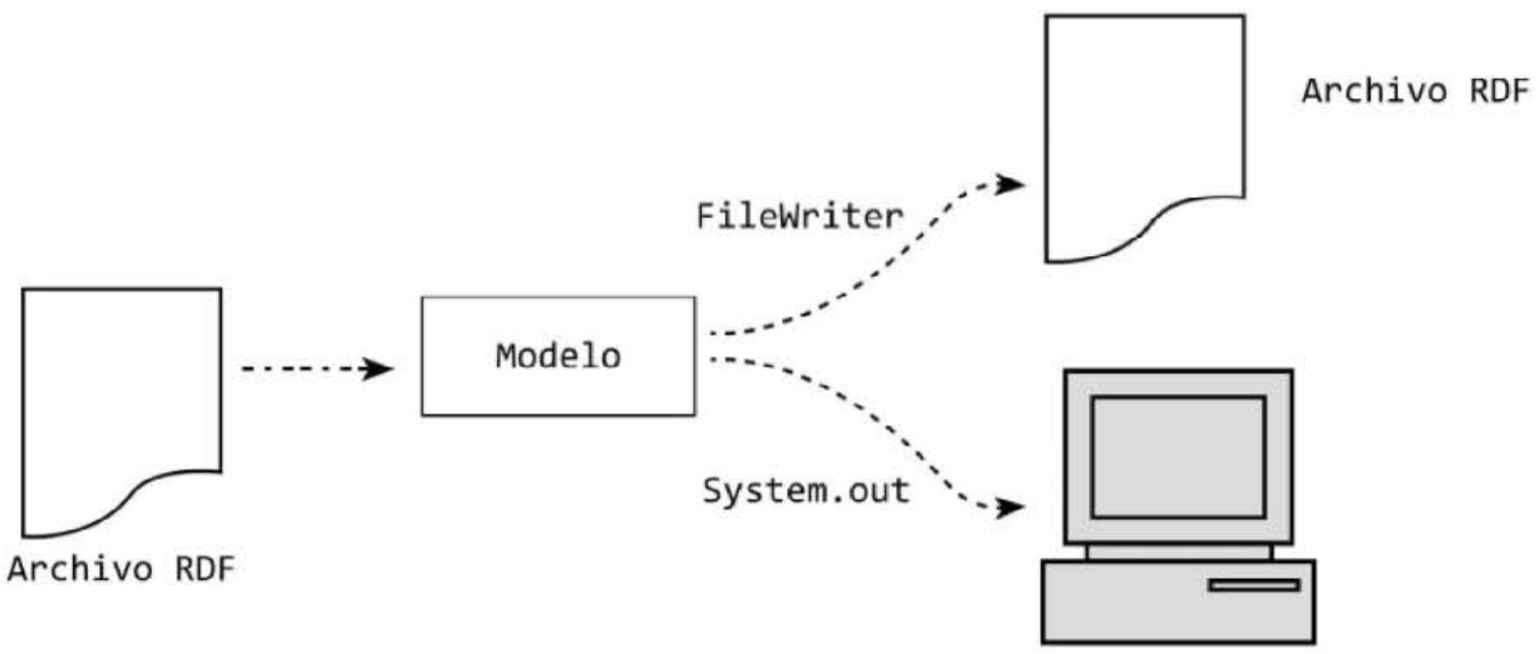


Fig. 14. Jena cargar y escribir modelo ontológico.

- Tipos de modelo
  - N-TRIPLE
  - N3
  - RDF/XML
  - RDF/XML-ABBREV

- Modelo CRUD
  - Create, crear.
  - Retrieve, recuperar.
  - Update, actualizar.
  - Delete, eliminar o borrar.

```
for(Iterator it = model.listStatements();it.hasNext();){
Statement stmt = (Statement) it.next();
System.out.println(stmt.getSubject().getLocalName());
}
```

CONSULTA DEL MODELO: SELECTORES

```
" Select * from MODEL where SUBJECT='Nestor' "

Resource id = model.getResource("http://www.pacis.org#Nestor ");
Selector selector =new SimpleSelector(id, (Property)null, (Resource)null);
for(Iterator it = model.listStatements(selector);it.hasNext();){ …}
// o, equivalentemente…
for(Iterator it = model.listStatements(id, (Property)null,
(Resource)null) ;it.hasNext();){ …}
```

ACTUALIZANDO EL MODELO.

El método changeobject de la clase statement es usado para este propósito.

```
Resource id = model.getResource("http://www.pacis.org#Nestor ");
Resource chinaSea =model.getResource("http://www.china.gov#ChinaSea");
Property   emptiesInto   =model.getProperty(Constantes.NATURALLY_OCCURRING_NS   +
"emptiesInto");
id.getProperty(emptiesInto).changeObject(chinaSea);
```

BORRAR UN STATEMENT

El método remove() de la clase Statement es usado para este propósito.

```
Resource yangtze =model.getResource("http://www.china.org/rivers#Yangtze");
Property emptiesInto =model.getProperty(Constantes.NATURALLY_OCCURRING_NS +
"emptiesInto");
yangtze.getProperty(empriesInto).remove();
```

K. Inferencia básica a un RDF

```
Property teachesAtProp = model.createProperty(ns, "teachesAt");
Property worksAtProp = model.createProperty(ns, "worksAt");
model.add(teachesAtProp, RDFS.subPropertyOf, worksAtProp);

model.createResource(ns + "Carmen").addProperty(teachesAtProp, "ETSIT");

(Carmen, teachesAt, ETSIT)

Reasoner reasoner = ReasonerRegistry.getRDFSReasoner();
InfModel infModel = ModelFactory.createInfModel(reasoner, data);

(Carmen, worksAt, ETSIT)
```

El *namespace* es la dirección fuente o base del RDF (http://www.pacis.org#). La variable "ns" es un string que contiene al namespace.

En el ejemplo anterior se agrega al individuo "Carmen", el cual posee como propiedad teachesAt y esta tiene como subpropiedad "worksAt", por lo que al ejecutar el razonador se muestra que el individuo Carmen posee como propiedad worksAt.

L. Jena API Ontología

Cree un modelo ontológico con el API de Jena. Los modelos son conjuntos statements.

```
OntModel ontology =ModelFactory.createOntologyModel(OntModelSpec.OWL_DL_MEM);
ontology.createClass(ns + " Stream ");
ontology.createClass(ns + "Brook");
ontology.createClass(ns + "River");
ontology.createClass(ns + "Tributary");

FunctionalProperty emptiesInto = m.createObjectProperty(ns + "emptiesInto",true).
convertToFunctionalProperty();

ObjectProperty feedsFrom = m.createObjectProperty(ns + "feedsFrom");
contains.setInverseOf(locatedIn);

ontology.add(feedsFrom, RDF.type, OWL.InverseFunctionalProperty);

//Almacenamos la ontología en un archivo OWL (opcional)
File file = new File("/Miproyecto/Mi_archivo.owl");
//Hay que capturar las Excepciones
if (!file.exists()){
   file.createNewFile();
}
model.write(new PrintWriter(file));
```

- **MODELFACTORY**

  Permite crear el modelo a través de su método createOntologyModel(); el cual crea un modelo por defecto que es compatible con versiones anteriores a Jena. Estos valores predeterminados son los siguientes: Permite crear el modelo a través de su método createOntologyModel(); el cual crea un modelo por defecto que es compatible con versiones anteriores a Jena. Estos valores predeterminados son los siguientes:

- OWl-Full lenguaje.
- In-memory de almacenamiento.
- RDFS para inferencia, que produce principalmente vínculos para las subclases y subproperty jerárquicas.

```
OntModel model = ModelFactory.createOntologyModel();
```

LENGUAJE ONTOLÓGICO	URI
RDFS	http://www.w3.org/2000/01/rdf-schema#
DAML+OIL	http://www.daml.org/2001/03/daml+oil#
OWL Full	http://www.w3.org/2002/07/owl#
OWL DL	http://www.w3.org/TR/owl-features/#term_OWLDL
OWL Lite	http://www.w3.org/TR/owl-features/#term_OWLLite

- **ONTMODELSPEC**

Define el tipo de modelo ontológico.

```
OntModel model = ModelFactory.createOntologyModel(OntModelSpec.OWL_ MEM);
```

ONTMODELSPEC	LANGUAGE PROFILE	ALMACENAMIENTO MODEL	RAZONADOR
OWL_MEM	OWL full	in-memory	Ninguno
OWL_MEM_TRANS_INF	OWL full	in-memory	transitive class-hierarchy inference
OWL_MEM_RULE_INF	OWL full	in-memory	rule-based reasoner with OWL rules
OWL_MEM_MICRO_RULE_INF	OWL full	in-memory	optimised rule-based reasoner with OWL rules
OWL_MEM_MINI_RULE_INF	OWL full	in-memory	rule-based reasoner with subset of OWL rules
OWL_DL_MEM	OWL DL	in-memory	Ninguno
OWL_DL_MEM_RDFS_INF	OWL DL	in-memory	rule reasoner with RDFS-level entailment-rules
OWL_DL_MEM_TRANS_INF	OWL DL	in-memory	transitive class-hierarchy inference
OWL_DL_MEM_RULE_INF	OWL DL	in-memory	rule-based reasoner with OWL rules
OWL_LITE_MEM	OWL Lite	in-memory	Ninguno
OWL_LITE_MEM_TRANS_INF	OWL Lite	in-memory	transitive class-hierarchy inference
OWL_LITE_MEM_RDFS_INF	OWL Lite	in-memory	rule reasoner with RDFS-level entailment-rules
OWL_LITE_MEM_RULES_INF	OWL Lite	in-memory	rule-based reasoner with OWL rules
DAML_MEM	DAML+OIL	in-memory	Ninguno
DAML_MEM_TRANS_INF	DAML+OIL	in-memory	transitive class-hierarchy inference
DAML_MEM_RDFS_INF	DAML+OIL	in-memory	rule reasoner with RDFS-level entailment-rules
DAML_MEM_RULE_INF	DAML+OIL	in-memory	rule-based reasoner with DAML rules
RDFS_MEM	RDFS	in-memory	Ninguno
RDFS_MEM_TRANS_INF	RDFS	in-memory	transitive class-hierarchy inference
RDFS_MEM_RDFS_INF	RDFS	in-memory	rule reasoner with RDFS-level entailment-rules

LENGUAJE ONTOLÓGICO	URI
RDFS	http://www.w3.org/2000/01/rdf-schema#
DAML+OIL	http://www.daml.org/2001/03/daml+oil#
OWL Full	http://www.w3.org/2002/07/owl#
OWL DL	http://www.w3.org/TR/owl-features/#term_OWLDL
OWL Lite	http://www.w3.org/TR/owl-features/#term_OWLLite

- **ONTMODELSPEC**

Define el tipo de modelo ontológico.

```
OntModel model = ModelFactory.createOntologyModel(OntModelSpec.OWL_ MEM);
```

ONTMODELSPEC	LANGUAGE PROFILE	ALMACENAMIENTO MODEL	RAZONADOR
OWL_MEM	OWL full	in-memory	Ninguno
OWL_MEM_TRANS_INF	OWL full	in-memory	transitive class-hierarchy inference
OWL_MEM_RULE_INF	OWL full	in-memory	rule-based reasoner with OWL rules
OWL_MEM_MICRO_RULE_INF	OWL full	in-memory	optimised rule-based reasoner with OWL rules
OWL_MEM_MINI_RULE_INF	OWL full	in-memory	rule-based reasoner with subset of OWL rules
OWL_DL_MEM	OWL DL	in-memory	Ninguno
OWL_DL_MEM_RDFS_INF	OWL DL	in-memory	rule reasoner with RDFS-level entailment-rules
OWL_DL_MEM_TRANS_INF	OWL DL	in-memory	transitive class-hierarchy inference
OWL_DL_MEM_RULE_INF	OWL DL	in-memory	rule-based reasoner with OWL rules
OWL_LITE_MEM	OWL Lite	in-memory	Ninguno
OWL_LITE_MEM_TRANS_INF	OWL Lite	in-memory	transitive class-hierarchy inference
OWL_LITE_MEM_RDFS_INF	OWL Lite	in-memory	rule reasoner with RDFS-level entailment-rules
OWL_LITE_MEM_RULES_INF	OWL Lite	in-memory	rule-based reasoner with OWL rules
DAML_MEM	DAML+OIL	in-memory	Ninguno
DAML_MEM_TRANS_INF	DAML+OIL	in-memory	transitive class-hierarchy inference
DAML_MEM_RDFS_INF	DAML+OIL	in-memory	rule reasoner with RDFS-level entailment-rules
DAML_MEM_RULE_INF	DAML+OIL	in-memory	rule-based reasoner with DAML rules
RDFS_MEM	RDFS	in-memory	Ninguno
RDFS_MEM_TRANS_INF	RDFS	in-memory	transitive class-hierarchy inference
RDFS_MEM_RDFS_INF	RDFS	in-memory	rule reasoner with RDFS-level entailment-rules

M. Personalizar un modelo específico

Se crea un *OntModelSpec* con un tipo definido; el cual se modificará con el documento ontológico y luego se creará el modelo con el tipo de ontología, tal como se muestra:

```
OntModelSpec s = new OntModelSpec( OntModelSpec.OWL_DL_MEM_RULE_INF);
    s.setDocumentManager( myDocMgr );
    OntModel m = ModelFactory.createOntologyModel( s );
```

15.5.5. Procesamiento de importaciones ontológicas

Una ontología puede importar a otra para la integración de los modelos. La figura muestra cómo un modelo de ontología construye una colección de modelos de importación.

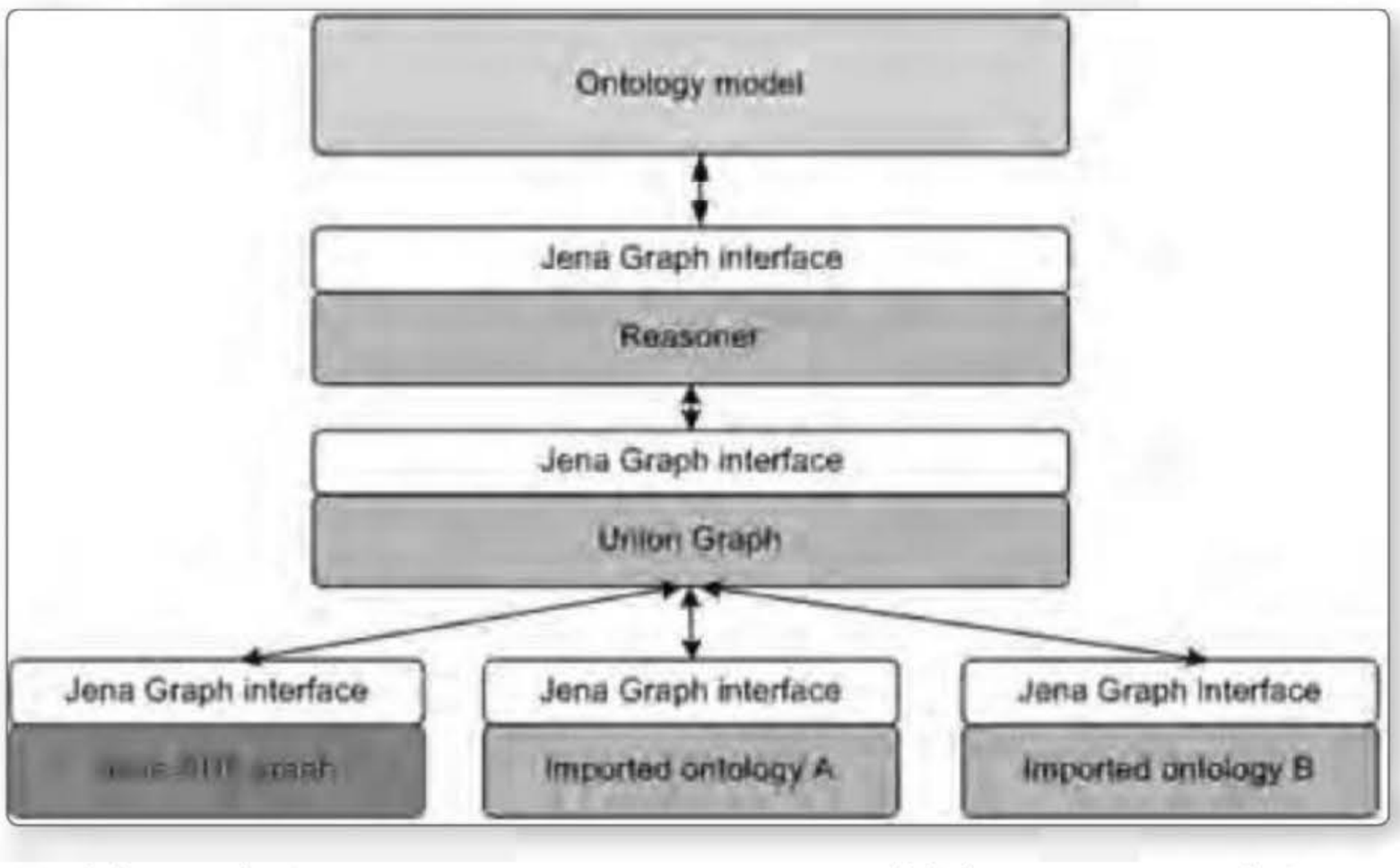


Fig. 15. Modelo ontologico compuesto por una estructura del documento para la importación.

Se va a cargar un archivo o documento de ontología de la misma manera que un modelo de Jena, con el método read de la clase OntModel. Existen varias variantes de lectura.

```
OntModel m = ModelFactory.createOntologyModel(OntModelSpec. OWL_DL_MEM_RULE_INF);
//tipos de lectura
m.read( String url )
m.read( Reader reader, String base )
m.read( InputStream reader, String base )
m.read( String url, String lang )
m.read( Reader reader, String base, String Lang )
m.read( InputStream reader, String base, String Lang )
```

Al usar el método read() se debe tener cuidado ya que pueden aceptar variantes del java.io.Reader de argumento al cargar documentos XML que contienen conjuntos de caracteres internacionalizados, ya que el manejo de codificación de caracteres por el lector y por los analizadores XML no es compatible.

Al cargarse la ontología, esta también carga a los individuos o instancias que se encuentran en ella. Cada documento ontológico importado se mantiene en una estructura de grafo separada al de la base. Esto es importante; ya que solo es el modelo base el cual se actualizará sin afectar a las importaciones.

Las **importaciones** son un proceso recursivo; por lo que si un documento base importa a la ontología A y este, a su vez, importa a B, quedaría la forma que se muestra en la figura anterior. Tenga presente que las importaciones se han aplanado. Un **ciclo de verificación** se utiliza para prevenir que el alimentador de documentación se atasque; así por ejemplo, si A importa a B, ¿qué importa A?

A. Administrando documentos ontológicos

Cada modelo de la ontología tiene asociado un *gestor de documentos* que ayuda a la transformación y se ocupa de la manipulación de los documentos relacionados. Para mayor comodidad, hay un gestor de documentos global que se utiliza por defecto por los modelos de la ontología. Se puede obtener una referencia a esta instancia para compartir, a través de *OntDocumentManager.getInstance ()*. En muchos casos, será suficiente con solo cambiar la configuración en el gestor del documento global para satisfacer las necesidades de su aplicación. Sin embargo, para tener un control más fino, puede crear separadamente un documento administrador y pasarlo al modelo ontológico cuando se crea a través del modelo *factory*. Para ello, cree un objeto especificación de la ontología y establezca el gestor de documentos. Por ejemplo:

```
OntDocumentManager mgr = new OntDocumentManager();
 … Codificacion …
// ahora utilizamos lo siguiente
OntModelSpec s = new OntModelSpec( OntModelSpec.RDFS_MEM );
s.setDocumentManager( mgr );
OntModel m = ModelFactory.createOntologyModel( s );
```

Tenga en cuenta que el modelo conserva una referencia al gestor de documentos que se creó. Así, si usted cambia las propiedades del administrador de documentos, serán afectados los documentos que previamente han sido construidos por el administrador de documentos.

B. Política sobre la administración de documentos

El administrador de documentos tiene un gran número de opciones de configuración y hay dos formas en que se puede cambiar los requisitos de la aplicación:

- Configurar los distintos parámetros del administrador de documentos por el código de Java.
- Cuando un administrador de documento dado se crea que puede cargar los valores de los diferentes parámetros a partir de una política del archivo, expresada en RDF.

El administrador de documentos tiene una lista de URL que se va a buscar desde una política de documento. Se detendrá en la primera entrada en la lista que se resuelve en el documento recuperable. La ruta de búsqueda por defecto de la política es así:

```
file::. /etc/ ont-policy.rdf; file:ont-policy.rdf . Usted puede encontrar la política por
defecto, que puede servir como plantilla para definir sus propias políticas, en el etc/
dentro del directorio de descarga de Jena.
Puede establecer las propiedades generales del administrador de documentos en la
política de la siguiente manera:
```

```
<DocumentManagerPolicy>
 <!-- policy for controlling the document manager's behaviour -->
 <processImports rdf:datatype="&xsd;boolean">true</processImports>
 <cacheModels rdf:datatype="&xsd;boolean">true</cacheModels>
</DocumentManagerPolicy>
```

Se puede encontrar el esquema simple que declara las distintas propiedades que se utilizan en este tipo de documento de política ontología (en el directorio de los vocabularios de la descarga de Jena) llamada ont-manager.rdf. Para cambiar la ruta de búsqueda que el administrador de documentos utiliza para inicializar. Si puede pasar la ruta de búsqueda de nuevo como una cadena cuando se crea un objeto de documento nuevo gerente o llamar al método setMetadataSearchPath().

15.5.6. Insertar un razonador al modelo

```
Model data = ModelFactory.createDefaultModel();
data.read("file:./etc/inf/naturally-occurring.nt", "N-TRIPLE");

OntModel ontology=ModelFactory.createOntologyModel(OntModelSpec.OWL_DL_MEM);
ontology.read("file:./etc/inf/naturally-occurring.owl");

Reasoner reasoner = ReasonerRegistry.getOWLReasoner();
InfModel infModel= ModelFactory.createInfModel(reasoner, ontology, data);

// O análogamente
Reasoner owlReasoner = ReasonerRegistry.getOWLReasoner();
Reasoner wnReasoner = owlReasoner.bindSchema(ontology);
InfModel infModel= ModelFactory.createInfModel(wnReasoner, data);
```

- **COMBINACIÓN DEL RAZONADOR GENÉRICO Y OWL**

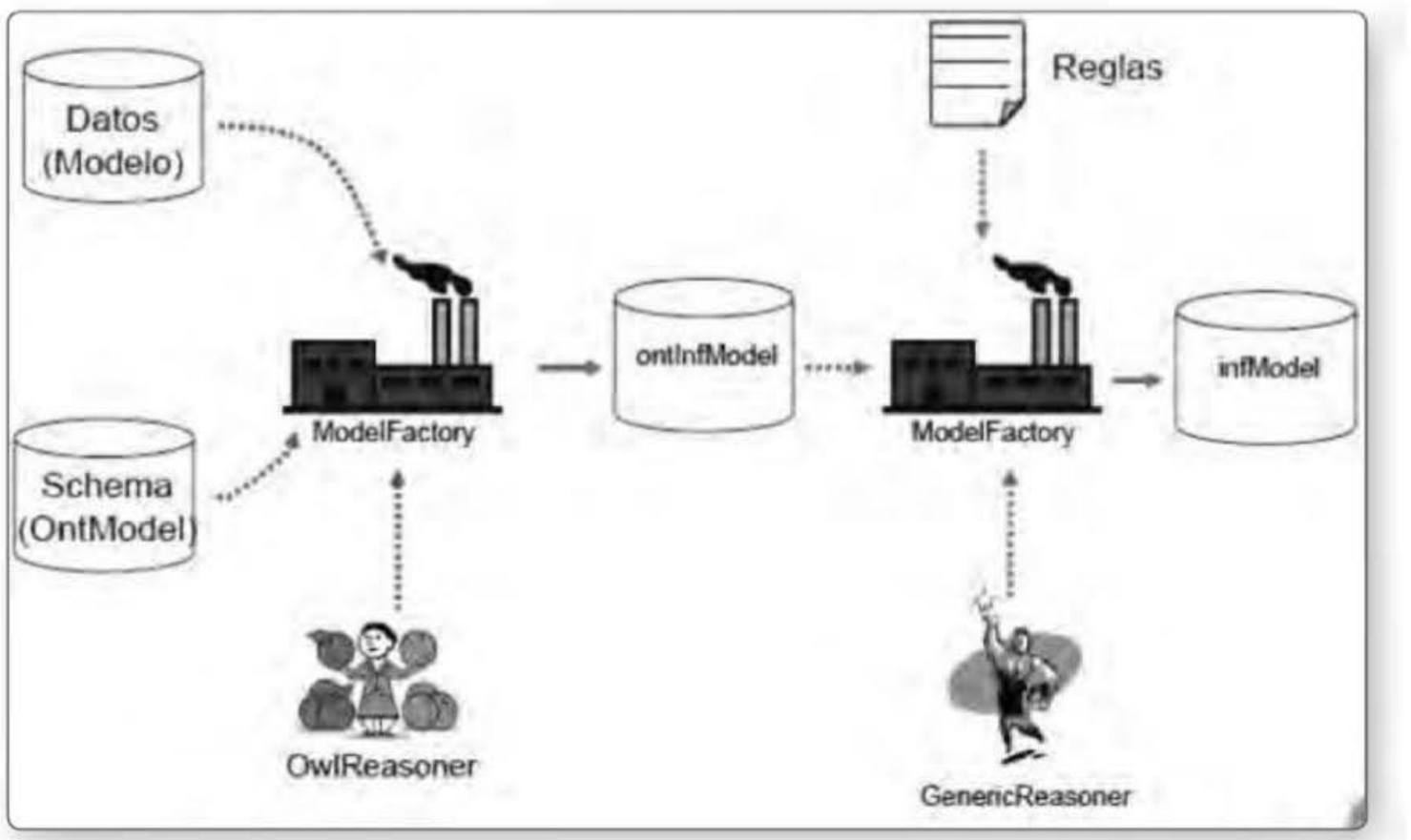


Fig. 16. El razonador genérico y el OWL.

```
OntModel schema =
ModelFactory.createOntologyModel(OntModelSpec.OWL_DL_MEM_TRANS_INF);
schema.read("file:./etc/inf/schema-inf-owl.owl");
Model data = FileManager.get().loadModel("file:./etc/inf/data.rdf");
Reasoner owlReasoner = ReasonerRegistry.getOWLReasoner();
InfModel owlInfModel = ModelFactory.createInfModel(owlReasoner, schema,data);
GenericRuleReasoner reasoner =new GenericRuleReasoner(Rule.rulesFromURL("file:./etc/
inf/myrule.rule"));
reasoner.setDerivationLogging(true);
InfModel infModel = ModelFactory.createInfModel(reasoner, owlInfModel);
```

◉ EJEMPLO DEL RAZONADOR PELLET

```
// creamos el razonador de tipo PELLET
Reasoner reasoner = PelletReasonerFactory.theInstance().create();
//Crear un modelo vacio.
Model emptyModel = ModelFactory.createDefaultModel( );

//Creamos un modelo inferenciado usando el razonador PELLET
InfModel model = ModelFactory.createInfModel(reasoner, emptyModel);

// Leer el archivo
 model.read(inFoafInstance, "");

// Imprime un reporte de validación del modelo
ValidityReport report = model.validate();
printIterator(report.getReports(), "Validation Results");

//ejecuta las reglas a todos los individuos
model.prepare();
```

```
public static void printIterator(Iterator<?> i, String header) {
   System.out.println(header);
   for(int c = 0; c < header.length(); c++)
     System.out.print("=");
   System.out.println();

   if(i.hasNext()) {
       while (i.hasNext())
         System.out.println( i.next() );
   }
   else
     System.out.println("<EMPTY>");

  System.out.println();
}
```

El método prepare de la clase InfModel() es el que ejecuta las reglas contenidas en el modelo ontológico. Esto da como resultado una inferencia la cual es colocada en el modelo ya sea como instancia, propiedad o valor.

Ejemplo:

Si el modelo se tratara de pacientes y enfermedades una de las reglas podría ser la de determinar qué enfermedad posee un determinado paciente; la cual sería agregada al momento de ejecutar este método en el modelo.

15.5.7. Resumen de razonadores

- *Razonador transitivo:* Básico, implementa propiedades transitivas y simétricas de *rdfs:subPropertyOf y rdfs: subClassOf.* consultas de jerarquías de clase y sus propiedades; además, da soporte de almacenamiento.
- *Razonador RDFS:* Implementa un subconjunto configurable de la funcionalidad ofrecida por RDFS.
- Razonadores OWL, OWL Mini, OWL Micro: Una implementación útil (aunque incompleta) de OWL/ Lite.
- *Razonador DAML:* Se utiliza internamente para proporcionar un soporte mínimo (a escala RDFS) a la inferencia mediante DAML.
- *Razonador genérico basado en reglas:* Un razonador basado en reglas.
- Jena permite incorporar nuevos razonadores externos (Pellet y otros).

15.5.8. Consultar un modelo

- Consultar directamente al modelo mediante list o selectores.
- Mediante consultas SPARQL o RDQL para RDF.
- Mediante SPARQL con JENA.

```
String queryString =""
+ "PREFIX owl: <http://www.w3.org/2002/07/owl#>"
+ "PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>"
+ "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>"

+ "PREFIX base:<http://www.owl-ontologies.com/Ontology1282851039.owl#>"

+ "SELECT ?s"
+ " WHERE{?s rdf:type base:Profesor}";

//Se establece la conexión a la base de datos por medio del archivo ttl capitulo 5
Store store = SDBFactory.connectStore( "archivo/miconexion.ttl") ;
    if(store != null){
     System.out.println("Conexión al store SDB ... Ok y consulta a la base de datos");
    }
    Dataset ds = DatasetStore.create(store) ;

// Ejecutar la consulta y obtener los resultados
    QueryExecution qe = QueryExecutionFactory.create(queryString, ds) ;
    try {
      ResultSet rs = qe.execSelect();
      ResultSetFormatter.out(rs);
    } finally { qe.close() ; }
  store.getConnection().close();
  store.close() ;
```

15.5.9. Jena persistencia

Jena tiene tres formas de dar persistencia a una base de conocimiento.

- Jena genera una persistencia de una OWL a una base relacional.
- Jena/SDB genera una persistencia a un RDF/OWL a una base relacional orientada a transacciones.
- Jena/TDB genera una persistencia a un RDF/OWL a una base relacional no transaccional.

15.5.10. Jena SDB

Permite el almacenamiento y consulta de datos RDF y OWL por medio de bases datos relacionales.

15.5.11. Instalación en Windows

Para el sistema Windows, se requiere del software Cygwin; el cual, puede descargarlo de la siguiente dirección:

http://www.cygwin.com/

- Java jdk: **http://www.oracle.com/technetwork/java/javase/downloads/index.html**
- Jena SDB:

http://openjena.org/wiki/SDB link home
http://sourceforge.net/projects/jena/files/SDB/ link descarga

- Base de datos que soporta:

Oracle 10g	Including OracleXE
Microsoft SQL Server 2005	Including MS SQL Express
DB2 9	Including DB2 9 Express
PostgreSQL	v8.2
MySQL	v5.0.22
Apache Derby	v10.2.2.0
H2	1.0.71
HSQLDB	1.8.0

DATABASE ENGINE	JDBC DRIVER
HSQLDB 1.8.0	
MySQL 4.1.11 MySQL 5.0.18	JDBC driver versions: 3.0, 3.1, 5.0
PostgreSQL 7.3 PostgreSQL 8.0	JDBC driver 7.3 JDBC driver 8.0
Apache Derby 10.1	
Oracle 10 XE	Oracle ojdbc14 driver (thin driver) 10.2.0.2
Oracle 9i Release 2	Oracle ojdbc14 driver (thin driver) 10.2.0.2
Oracle 10g Release 2	Oracle ojdbc14 driver (thin driver) 10.2.0.2
Microsoft SQL Server 2005 Express SP1	Microsoft SQL Server 2005 JDBC Driver
Microsoft SQL Server 2000 Microsoft SQL Server Desktop Edition	Microsoft SQL Server 2005 JDBC Driver jTDS version 1.2

Antes de comenzar debe tener todo instalado como: Java, el jdbc de MySQL (protege modelado de ontologias) y luego instale el cygwin para configurar el Cygwin; el cual es un aplicativo que permite ejecutar programas de la plataforma Linux en Windows. Este manual enseñará como navegar en la consola del "cygwin" que es lo mismo que navegar en Linux.

Cuando se instala el programa se creará una carpeta "../ cygwin/home/user" donde user es el usuario actual de Windows.

- *cd /cygdrive/[unidad de disco]/[nombre carpeta]* permite ir a una unidad del disco distinta dentro de una carpeta. Es necesario introducir una carpeta.
- *ls /cygdrive/[unidad de disco]/[nombre carpeta]* lista los archivos de esa carpeta.

EJECUTANDO LA APLICACIÓN

Paso 1: *Escriba el path.*

Existen diversas formas de escribir los path, cualquiera de estas es válida.

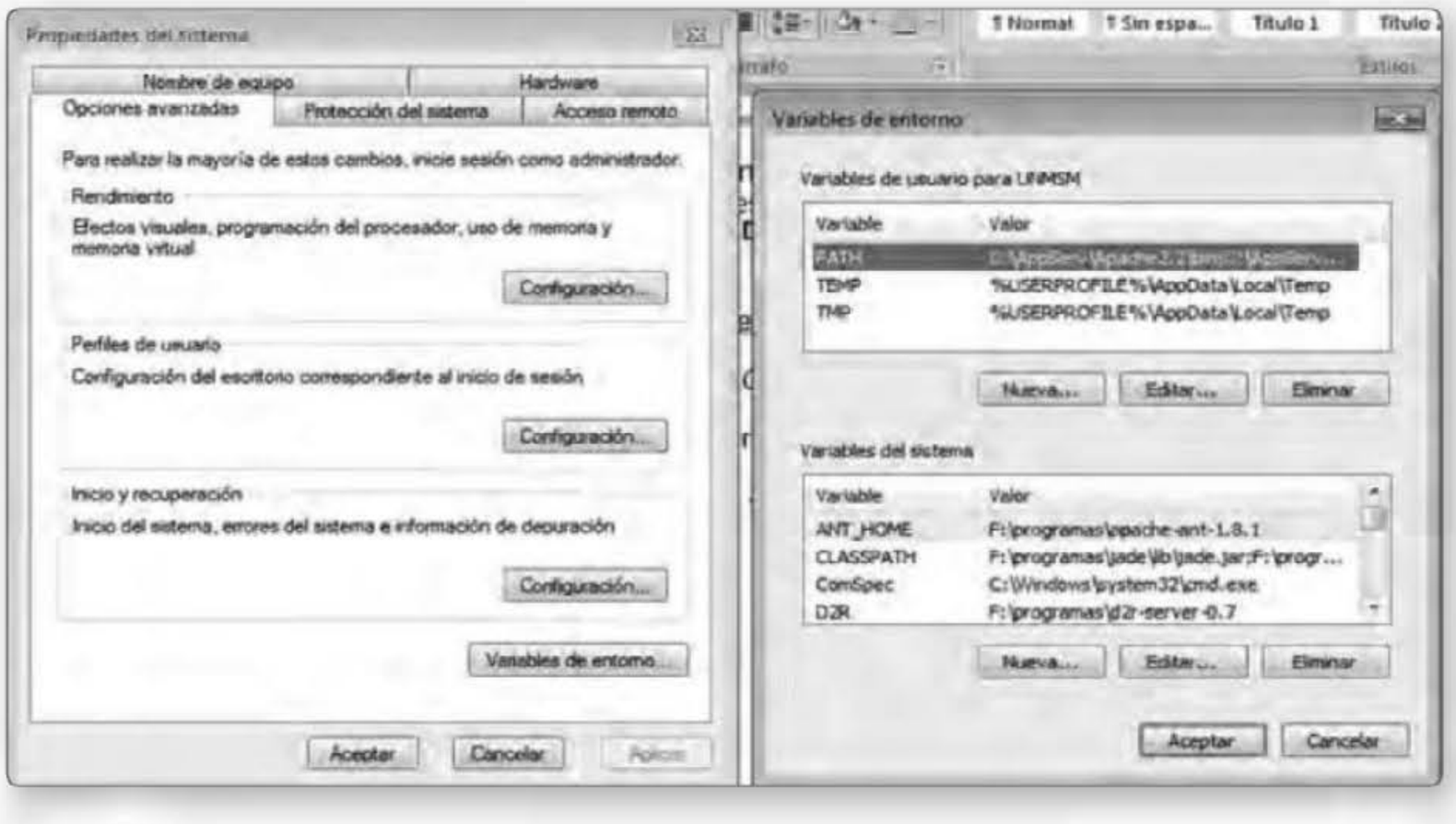
```
ls \c/window
ls c:/windows
ls c:\\windows
ls \c\\windows
ls /cygdrive/c/Windows
```

Paso 2: *Se va a exportar una aplicación de Linux para su cygwin, en este caso el SDB.*

- *SDBROOT: Se coloca la dirección de la carpeta raíz del SDB.*
- *SDB_JDBC: Se coloca la dirección del jdb (F:\programas\mysql-connector-java-5.1.13\mysql-connector-java-5.1.13-bin.jar).*

Donde SDBROOT será la variable a usar en el path de Windows.

Paso 3: *Diríjase a Propiedades del sistema > Opciones avanzadas > Variables de entorno.*



Paso 4: *Se creará una nueva variable de sistema para que esté habilitada para todos los usuarios de ese ordenador.*



Paso 5: *Coloque en el path de Windows Propiedades de mi PC > Opciones avanzadas > Variables de entorno > variables de sistema > buscar Path (editar) > Valor de variable y escriba en la última línea ";%SDBROOT%\bin".*



Paso 6: *Para la nueva variable de entorno SDB_JDBC, se realiza de la misma manera con excepción del path y se coloca el jar del driver de su jdbc.*

Paso 7: *Se obtiene como nombre de variable la dirección del .jar. Para este ejemplo, se usó el de MySQL "F:\programas\mysql-connector-java-5.1.13\mysql-connector-java-5.1.13-bin.jar".*

15.5.12. Construcción del modelo persistente en la base datos Mysql y carga de los datos por la consola Cygwin

Una vez instalado, el primer paso es crear el almacén de tripletas, para lo cual creará un archivo con el nombre store.ttl, por ejemplo, con el siguiente contenido:

```
@prefix rdfs:   <http://www.w3.org/2000/01/rdf-schema#> .
 @prefix rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
 @prefix ja:    <http://jena.hpl.hp.com/2005/11/Assembler#> .
 @prefix sdb:   <http://jena.hpl.hp.com/2007/sdb#> .

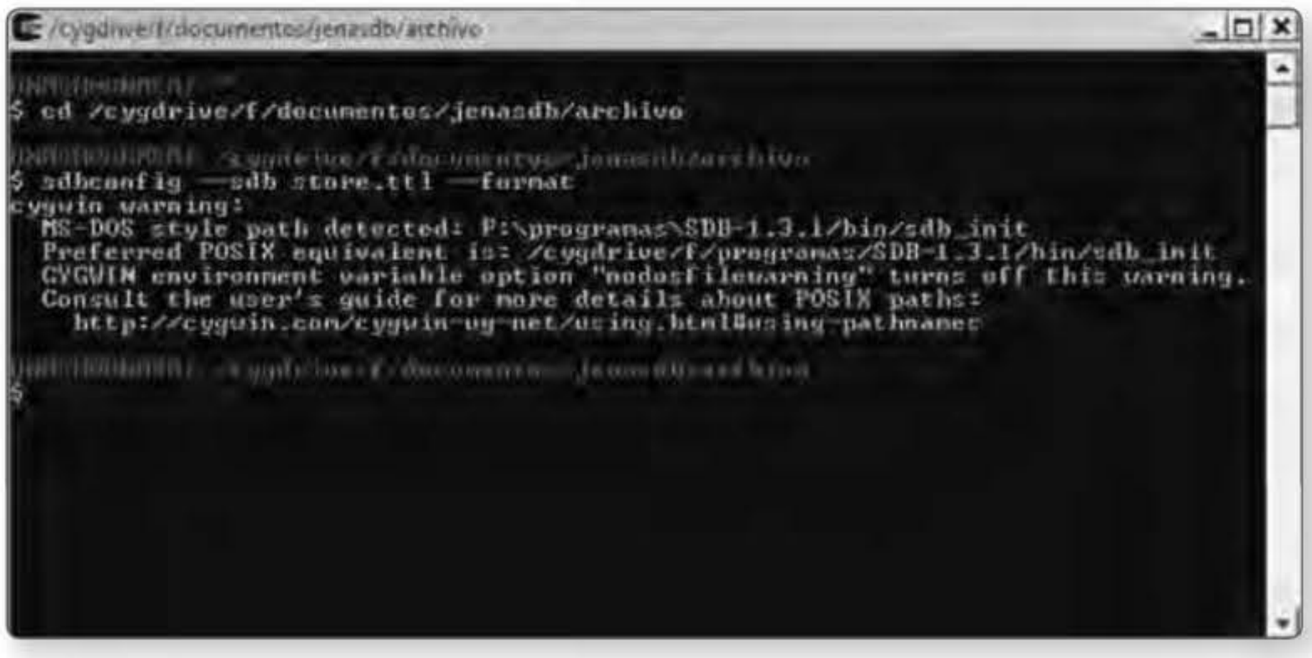
  _:c rdf:type sdb:SDBConnection ;
    sdb:sdbType    “mysql” ;
    sdb:sdbName    “rdf_testdb” ;
    sdb:sdbHost    “localhost” ;
    sdb:sdbUser  “userdb”;
    sdb:sdbPassword  “pwddb”;
    sdb:driver     “com.mysql.jdbc.Driver” ;
    .

   [] rdf:type sdb:Store ;
     S
  db:layout     “layout2/index” ;
    sdb:sdbName    “rdf_testdb” ;
    sdb:connection   _:c ;
    .
```

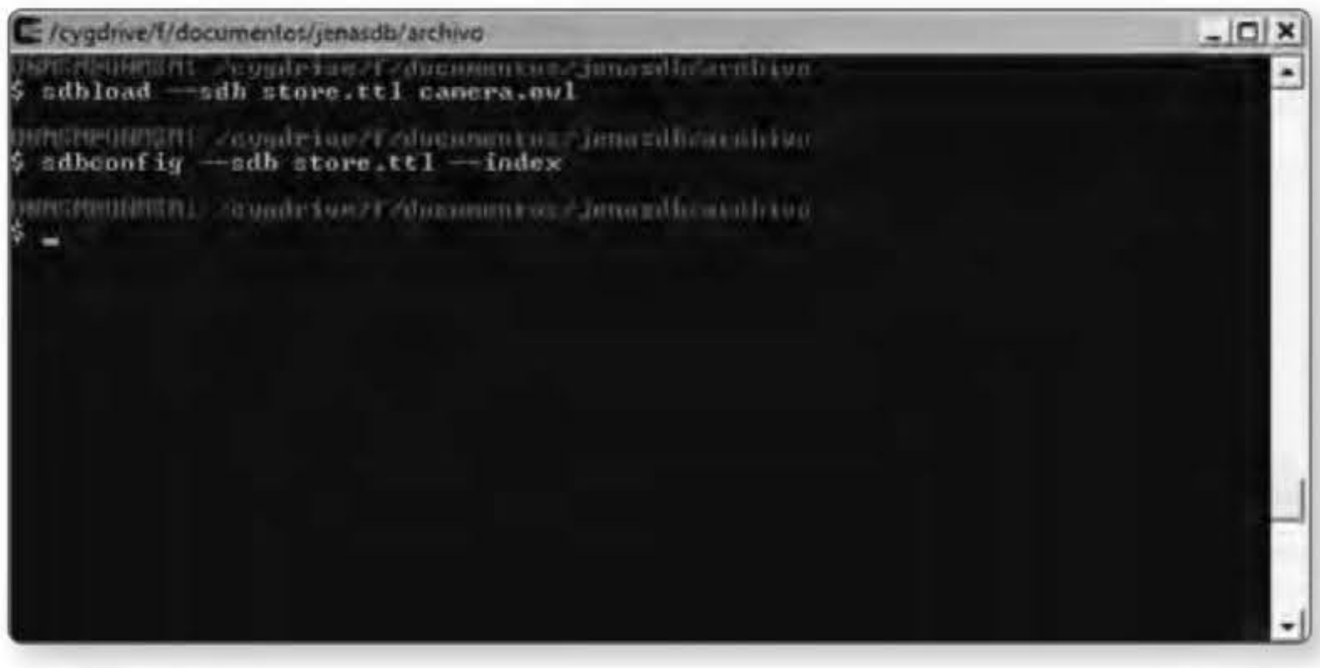
En él se indica que se utilizará MySQL como almacenamiento y cómo acceder a la base de datos que contendrá la información:

- servidor *(localhost)*,
- usuario *(userdb)*,
- contraseña *(pbddb)* y
- nombre de la base de datos *(rdf_testdb).*

La base de datos (en la cual se están almacenando los valores de los archivos OWL o RDF) puede contener toda una base de datos relacionados. Una vez hecho esto se puede crear el almacén de tripletas RDF sobre la base de datos indicada. Ubique la carpeta que contenga el archivo *.ttl.



El paso siguiente consiste en cargar el archivo OWL y seguidamente cargar los índices. Es conveniente hacerlo en este orden sobre todo si se va a cargar muchos datos.



Una vez cargada la información, consulte en SPARQL.

Por ejemplo, se ha cargado una ontología con estilos de música y autores. Se puede solicitar obtener todos los individuos o instancias de la clase Window; para ello, se digita lo siguiente:

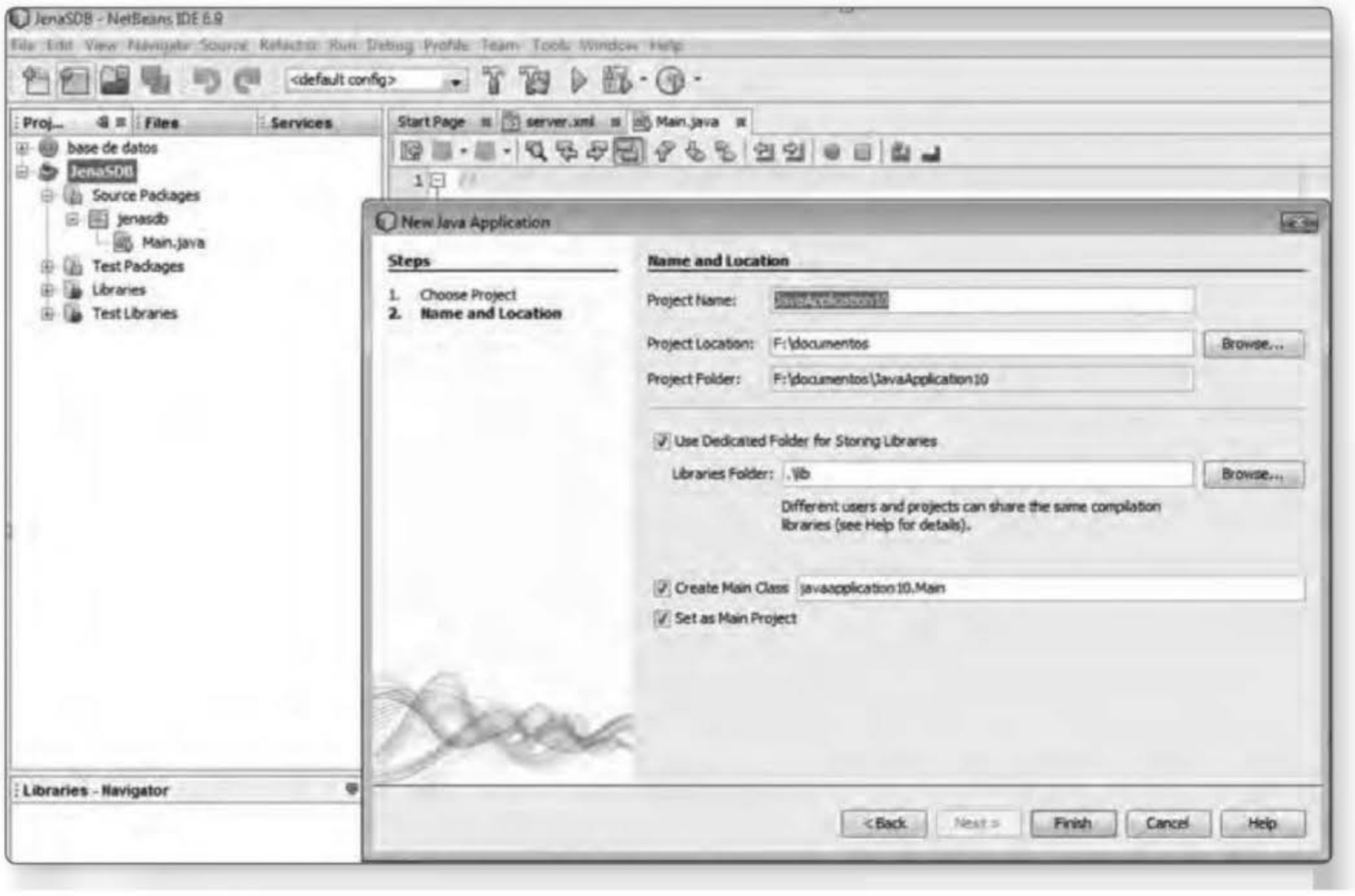
```
"PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>" +
 "PREFIX base: <http://www.xfront.com/owl/ontologies/camera/#>" +
 "SELECT ?subject "+
 "WHERE {"+
 "?subject rdf:type base:Window ."+
 "}";
```

15.5.13. Aplicación Java utilizando la API Jena/SDB

Cree un nuevo proyecto, por ejemplo con NetBeans.

Paso 1: Añada las librerías .jar de la carpeta lib de SDB. El siguiente código muestra cómo conectar con el almacén OWL y realizar la consulta recorriendo el recordset, registro a registro. Existe también un método específico que muestra por pantalla directamente los resultados en forma de tabla sin tener que recorrerlos.

Paso 2: Cree una nueva aplicación de java.



Paso 3: Cargue las librerías necesarias haciendo clic derecho en el proyecto > clic en propiedades. En la parte de categorías, haga clic para selecionar Librerías > clic en Add JAR/Folder. Luego, la siguiente ventana aparece.



Paso 4: Dele clic a *Cargar el archivo MySQL-connector-java-5.1.13-bin.jar*.

Paso 5: En el archivo Main del proyecto, copie el siguiente código:

```
/**
 *
 */
public class Main {
  /**
   * @param args the command line arguments
   */
  public static void main(String[] args) {
       String queryString =”PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>”
        + “PREFIX base: <http://www.xfront.com/owl/ontologies/camera/#>”
        + “SELECT ?subject “+
        “WHERE {“+
        “?subject rdf:type base:Window .”+
        “}”;
    QueryOWL a=new QueryOWL();
    a.query(queryString);
    System.out.println();
    QueryDataBase b= new QueryDataBase();
    b.query(queryString);
  }
}
```

Paso 6: Cree dos objetos uno que realiza la consulta a la OWL y el otro realiza la consulta a la base de datos:

```
import com.hp.hpl.jena.query.*;
import com.hp.hpl.jena.sdb.SDBFactory;
import com.hp.hpl.jena.sdb.Store;
import com.hp.hpl.jena.sdb.store.DatasetStore;

/**
 *
 */
public class QueryDataBase {
  public void query (String query){
    Store store = SDBFactory.connectStore( “archivo/store.ttl”);
    if(store != null){
      System.out.println(“Conexión al store SDB ... Ok”);
    }

    Dataset ds = DatasetStore.create(store) ;
    QueryExecution qe = QueryExecutionFactory.create(query, ds);

    try {
      ResultSet rs = qe.execSelect();
    } finally { qe.close() ; }

  store.getConnection().close();
  store.close() ;
 }
}
```

Paso 7: Con la siguiente instrucción:

```
sdbconfig --sd [mi_archivo].ttl --format
```

Se agregan las tablas necesarias a la base de datos; las cuales contendrán el modelo de la OWL y las instancias que posea. Se hará lo mismo; pero con una clase de java, de la siguiente manera:

```
import com.hp.hpl.jena.sdb.SDBFactory;
import com.hp.hpl.jena.sdb.Store;
import com.hp.hpl.jena.sdb.StoreDesc;
import com.hp.hpl.jena.sdb.sql.JDBC;
import com.hp.hpl.jena.sdb.sql.SDBConnection;
import com.hp.hpl.jena.sdb.store.DatabaseType;
import com.hp.hpl.jena.sdb.store.LayoutType;
import java.sql.SQLException;
public class Create {
  public void Crear_store () throws SQLException{
   StoreDesc storeDesc =new StoreDesc (LayoutType.LayoutTripleNodesHash, DatabaseType.
MySQL);

    storeDesc=StoreDesc.read("archivo/miconexion.ttl");
    Store store =SDBFactory.connectStore(storeDesc);

    //crea las tablas e indices que necesita el sdb
    store.getTableFormatter().create();
    //Vacia las tablas
    store.getTableFormatter().truncate();

    store.getConnection().close() ;
    store.close()
  }
}
```

Si no se contase con un archivo **.ttl** el código seria de la siguiente manera.

```
import com.hp.hpl.jena.sdb.SDBFactory;
import com.hp.hpl.jena.sdb.Store;
import com.hp.hpl.jena.sdb.StoreDesc;
import com.hp.hpl.jena.sdb.sql.JDBC;
import com.hp.hpl.jena.sdb.sql.SDBConnection;
import com.hp.hpl.jena.sdb.store.DatabaseType;
import com.hp.hpl.jena.sdb.store.LayoutType;
import java.sql.SQLException;
public class Create {
  public void Crear_store () throws SQLException{
   StoreDesc storeDesc =new StoreDesc (LayoutType.LayoutTripleNodesHash, DatabaseType.
MySQL);

    JDBC.loadDriverMySQL();
    String jdbcURL =”jdbc:mysql://localhost:3306/java_jenaSDB”;
    SDBConnection conn= new SDBConnection(jdbcURL, “root”, “root”);
    Store store =SDBFactory.connectStore(conn,storeDesc);
```

```
        //crea las tablas e índices que necesita el sdb
        store.getTableFormatter().create();
        //Vacía las tablas
        store.getTableFormatter().truncate();

        store.getConnection().close() ;
        store.close();
    }
}
```

Por último, en el main.

```
import CrearStore.Create;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;

public class Main {
  /**
   * @param args the command line arguments
   */
  public static void main(String[] args) {
        Create sdbfile=new Create();
    try {
      sdbfile.Crear_store();
    } catch (SQLException ex) {
      Logger.getLogger(Main.class.getName()).log(Level.SEVERE,null,ex);
    }
  }
}

sdbload --sd [mi_archivo].ttl -[Mi_archivo].owl
sdbconfig --sd [mi_archivo].ttl --index
```

Para modificar el código anterior, realice lo siguiente:

```
import com.hp.hpl.jena.ontology.OntModel;
import com.hp.hpl.jena.ontology.OntModelSpec;
import com.hp.hpl.jena.rdf.model.Model;
import com.hp.hpl.jena.rdf.model.ModelFactory;
import com.hp.hpl.jena.sdb.SDBFactory;
import com.hp.hpl.jena.sdb.Store;
import com.hp.hpl.jena.sdb.StoreDesc;
import com.hp.hpl.jena.sdb.sql.JDBC;
import com.hp.hpl.jena.sdb.sql.SDBConnection;
import com.hp.hpl.jena.sdb.store.DatabaseType;
import com.hp.hpl.jena.sdb.store.LayoutType;
import com.hp.hpl.jena.util.FileManager;
import java.io.InputStream;
import java.sql.SQLException;


public class Create {
  public void Crear_store () throws SQLException{
    StoreDesc storeDesc =new StoreDesc (LayoutType.LayoutTripleNodesHash, DatabaseType.
```

```
MySQL);
    storeDesc=StoreDesc.read("archivo/miconexion.ttl");
    Store store =SDBFactory.connectStore(conn, storeDesc);

    //crea todas las tablas e indices necesarios para la sdb
    store.getTableFormatter().create();
    //Vacia las tablas
    store.getTableFormatter().truncate();

/*
 * las siguientes lineas se encargaran de realizar lo escrito en la consola
 * sdbload --sdb miconexion.ttl camera.owl
 * sdbconfig --sdb miconexion.ttl --index
 */
    Model base = SDBFactory.connectDefaultModel( store);
    OntModel ontmodel = ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM, base);
    InputStream in = FileManager.get().open("archivo/camera.owl");
    ontmodel.read( in, "");
/*
 * cierra la conexion a la base de datos y al store
 */
    store.getConnection().close() ;
    store.close() ;
  }
}
```

15.5.14. Jena/SDB diseños de base de datos

SDB no tiene un diseño de base de datos único. Esta página es un resumen informal de los dos tipos principales ("layout2/hash" y "layout2/index"). En SDB, una tienda es un conjunto de datos RDF es una base de datos SQL. Bases de datos de tipo Layout2 tiene una tabla Triples de la gráfica; por defecto, una tabla Quads para los nombres de gráficos. En la tabla Triples y en la Quads, las columnas son enteras referenciando a una tabla Nodes.

- En la forma de hash, los números enteros son los hashes de 8 bytes del nodo.
- En la forma de índice, los números enteros son identificadores de secuencia de 4 bytes en la tabla de nodo.

- **TRIPLES**

```
+-----------+
| S | P | O |
+-----------+
```

Clave principal: SPO
Índices: PO, OS

- **QUADS**

```
+---------------+
G | | S | P | S |
+---------------+
```

Clave principal: GSPO
Índices: GPO, GOS, SPO, OS, PO.

- **NODES**

  En el diseño de índice basado en la tabla es la siguiente:

  ```
  +-----------------------------------------+
  | Id | Hash | lex | lang | datatype | value type |
  +-----------------------------------------+
  Clave principal: Id.
  Índice: Hash
  ```

- **HASH**

  ```
  +---------------------------------------+
  | Hash | lex | lang | datatype | value type |
  +---------------------------------------+

  Hash: Primaria clave
  ```

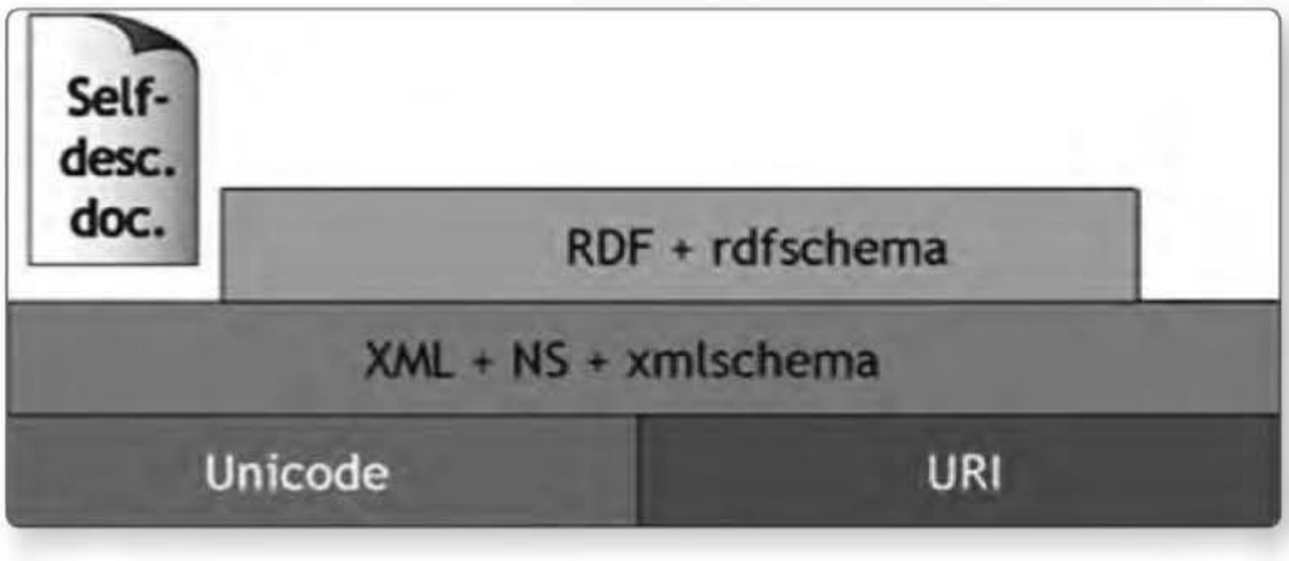
Todos los campos de caracteres son Unicode, soporta cualquier conjunto de caracteres, incluyendo el uso de lenguaje mixto.

SWRL: *Semantic Web Rule Language* sirve de puente para las reglas de negocios y la producción de sistemas de reglas lógicas más expresivas.

15.6. ANEXOS

15.6.1. RDF (Resource Description Framework)

Se construyó sobre tecnologías Unicode, URI y XML + NS + xmlschema.



- **UNICODE**

  Estándar para representar y manipular texto en cualquier idioma del mundo. Brinda la capacidad de intercambiar informacion entre las maquinas.

- **URI (UNIFORM RESOURCE IDENTIFIERS)**

  Identificadores de recursos únicos, sin posibilidad de ambigüedad. Los documentos en la web actual tienen una URL.

  - Localizador único del recurso: http://www.pacis.com

Cada objeto en la web semántica tiene una URI.

- Identificador único de un recurso.

Algunas URI pueden ser URL:

- El paciente posee una URI (http://www.w3.org/2000/01/rdf-schema#Paciente).

- **XML (eXtensible Markup Language)**

  Metalenguaje de etiquetas extensibles que se puede "acomodar" a las necesidades de cada usuario. Asimismo, describe una estructura de árbol capaz de ser procesada por las máquinas.

- **XML + NS (espacio de nombres, Name Space)**

  Etiqueta una determina URI para su utilización en una aplicación o consulta:

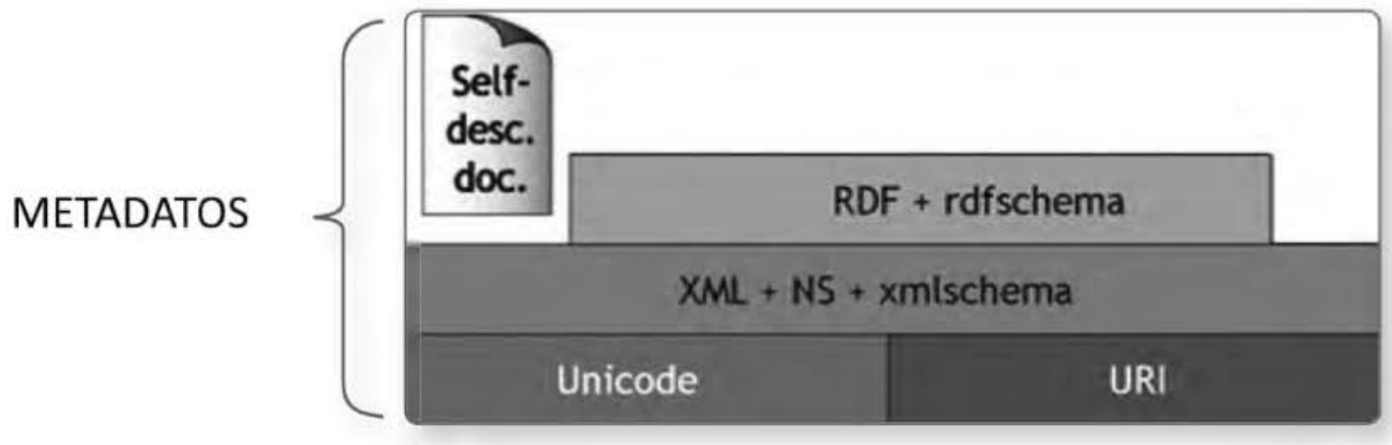
```
<Jesus:price xmlns:Jesus='http://www.pacis.com/schema' units='Soles'>
32.18
</Jesus:price>
```

- **XML + Schema**

  Permite definir documentos de tipo XML. Ejemplo obtenido de la ref[9]:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2000/08/XMLSchema">
<xsd:element name="persona" type="tipoPersona"/>
<xsd:element name="comentario" type="xsd:string"/>
<xsd:complexType name="tipoPersona">
<xsd:sequence>
<xsd:element name="nombre" type="xsd:string"/>
<xsd:element ref="comentario" minOccurs="0"/>
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

- **RDF**



Se trata de una descripción de un modelo de datos; en lugar, de la descripción de un vocabulario de datos específicos; le agrega semántica a un XML. Se trata de un modelo de metadatos que se basa en la sintaxis XML y otras como N3, N-Triple, Turtle, RxR, Trix.

No ofrece un significado formal, computable a las propiedades. Sigue el modelo de la tripleta:

Se trata de una descripción de un modelo de datos; en lugar, de la descripción de un vocabulario de datos específicos; le agrega semántica a un XML. Se trata de un modelo de metadatos que se basa en la sintaxis XML y otras como N3, N-Triple, Turtle, RxR, Trix.

No ofrece un significado formal, computable a las propiedades. Sigue el modelo de la tripleta:

```
sujeto-propiededad-objeto
```

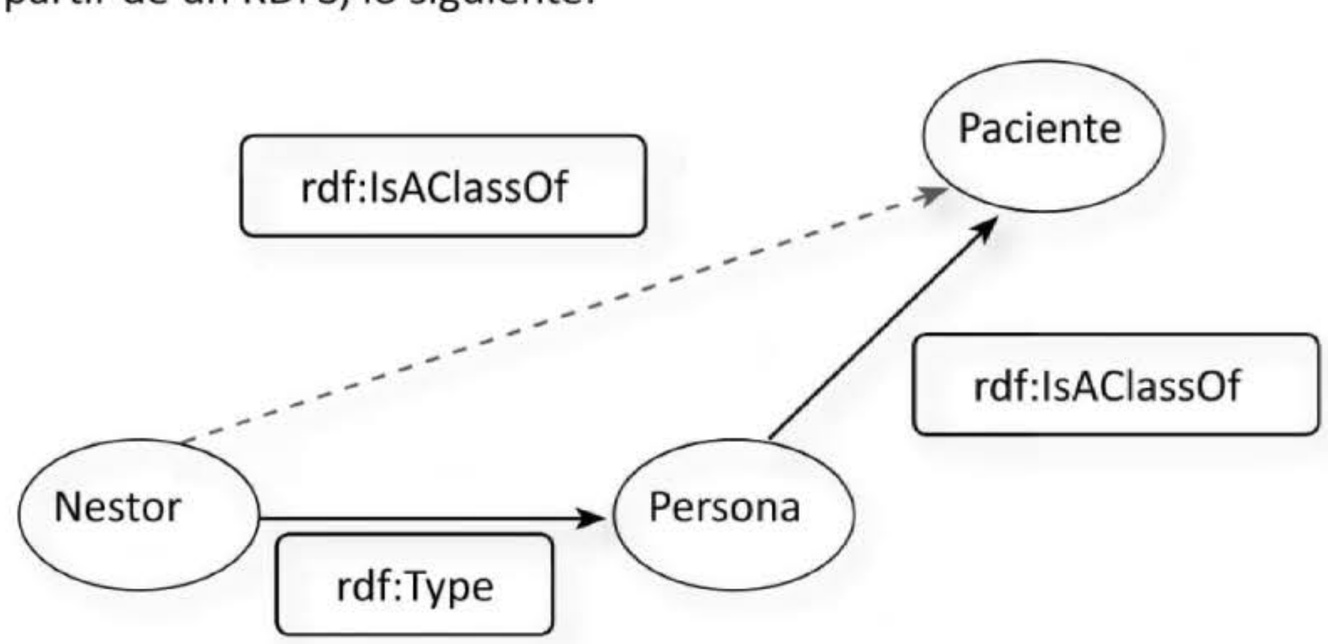
Sujeto: Se trata de un identificador de un recurso.
Propiedad: Define la relación o tipo de valor.
Objeto: Es el valor o literal de la propiedad.

15.6.2. RDFS (rdfschema)

Otorga un framework para describir clases y propiedades. Es una especie de type system para RDF (comparable a los type systems usados en OOP como Java).

Define vocabularios, con términos y relaciones entre los mismos. Brinda cierta semántica a los recursos y propiedades; además, permite a las computadoras hacer ciertas tareas Class, SubClassof, Type, Propety.

Es posible inferir a partir de un RDFS, lo siguiente:



15.6.3. D2r Server

Publica la base de datos relacionales en la web semántica.

A. Descargar

d2rserver

B. Requerimientos

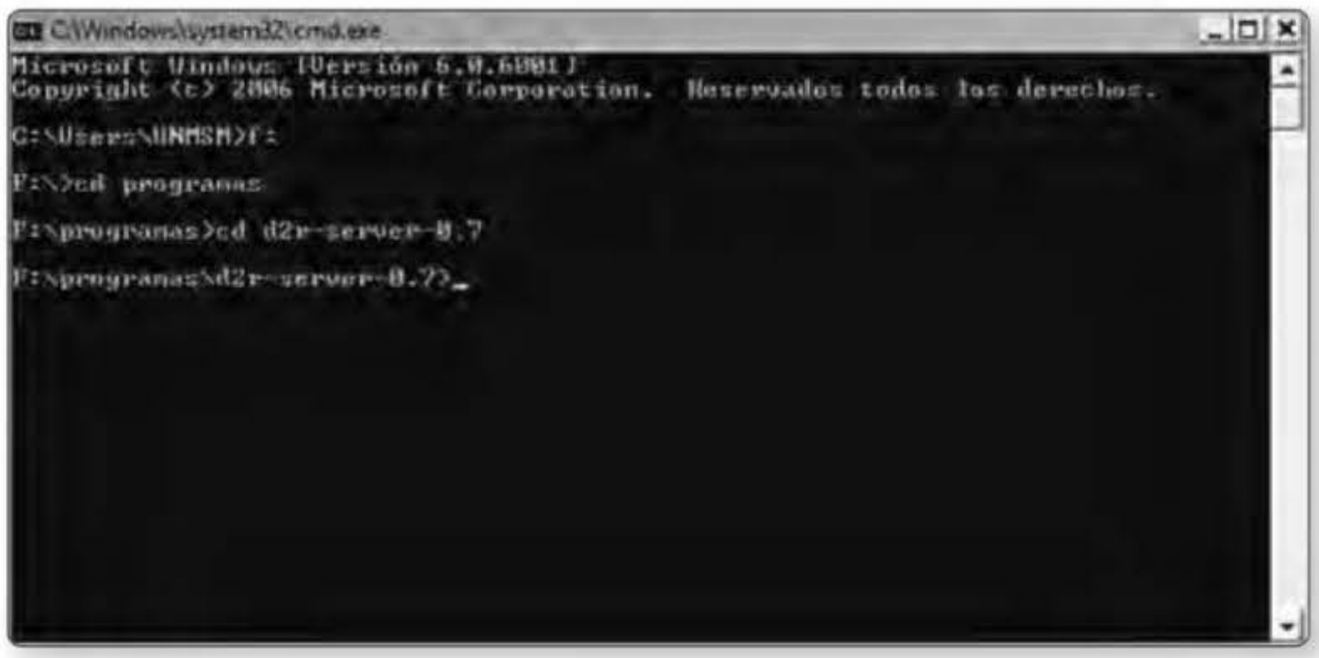
- Java,
- jdbc drivers,
- opcionalmente tener un servlet (como toncat, apache, J2EE, Glasfiship u otro).

C. Instalación

- Descomprima el archivo descargado.
- Genere un mapa de la base de datos; para este caso es una base creada en MySQL.
  - En Windows:

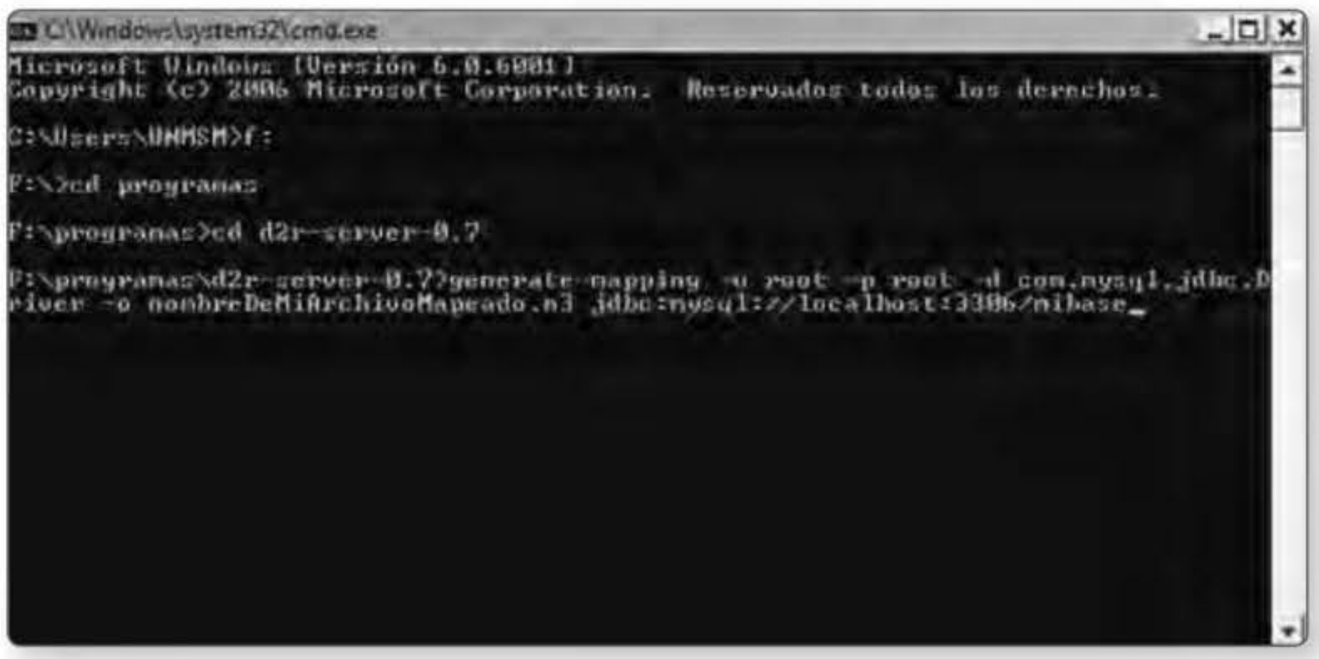
Paso 1: Abra cmd.

Paso 2: Diríjase a la carpeta donde se descomprimió el archivo.



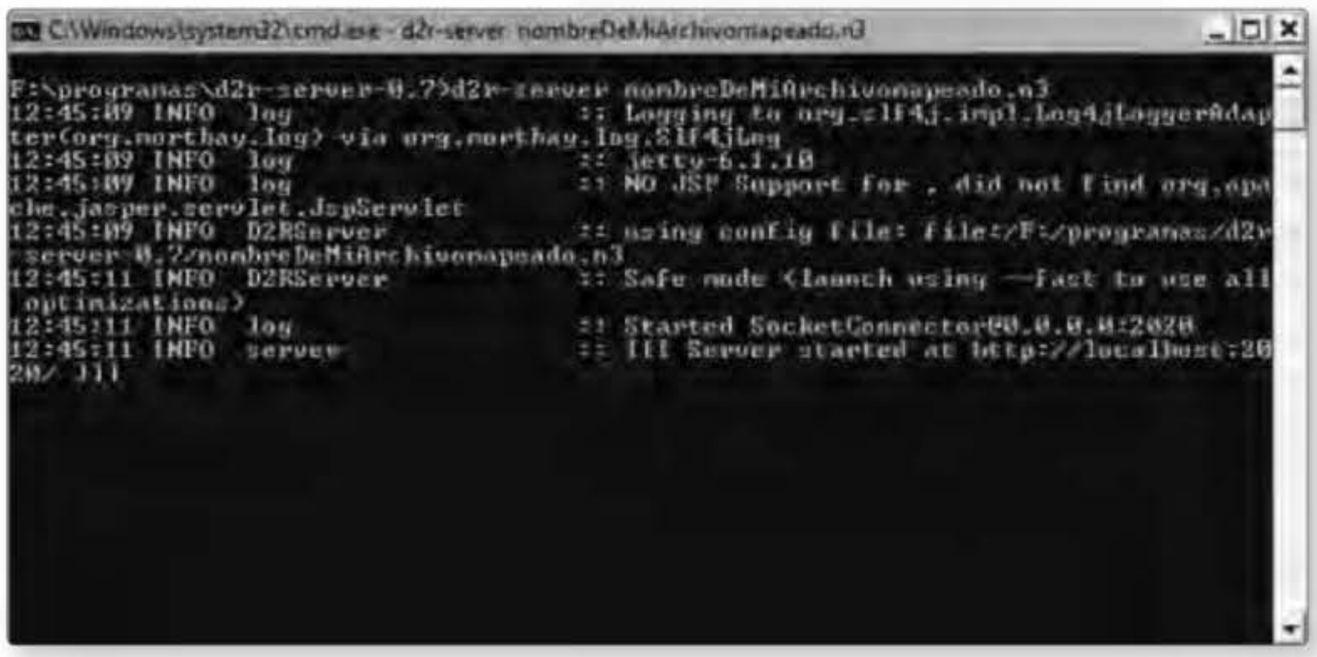
Paso 3: Escriba lo siguiente:

```
generate-mapping -u root -p root -d com.mysql.jdbc.Driver -o nombreMapeado.n3
jdbc:mysql://localhost:3306/mibase
```



Paso 4: Corra el servidor con el archivo mapeado.

```
d2r-server nombreDeMiArchivomapeado.n3
```



Paso 5: Ejecute en un explorador web: http://localhost:2020



15.6.4. D2RQ con Jena

Para el tiempo en que se escribió este manual, se usó la versión d2rq-0.7.jar. Para el siguiente ejemplo, se creará un proyecto en Netbeans y se usará el archivo n3, el cual es generado por el d2r-server, donde se agregará todos los jar de la carpeta \d2r-server-0.7\lib. También es necesario el jdbc driver, para este proyecto se utilizó el MYSQL.

```
import com.hp.hpl.jena.rdf.model.Model;
import com.hp.hpl.jena.rdf.model.Resource;
import com.hp.hpl.jena.rdf.model.StmtIterator;
import com.hp.hpl.jena.sparql.vocabulary.FOAF;
import com.hp.hpl.jena.vocabulary.DC;
import com.hp.hpl.jena.vocabulary.RDF;
import de.fuberlin.wiwiss.d2rq.ModelD2RQ;
public class Main {
  public static void main(String[] args) {
    // Se crea un modelo en memoria
    Model m =new ModelD2RQ("doc/test.n3");

    StmtIterator paperIt=m.listLiteralStatements(null, RDF.type, true);
    while (paperIt.hasNext()){
      Resource paper =paperIt.nextStatement().getSubject();
      System.out.println("Paper: " + paper.getProperty(DC.title).getString());
      StmtIterator authorIt =paper.listProperties(DC.creator);
       while (authorIt.hasNext()) {
         Resource author = authorIt.nextStatement().getResource();
         System.out.println("Author: " + author.getProperty(FOAF.name).getString());
       }
         System.out.println();
      }
    }

  }
```



HTML5

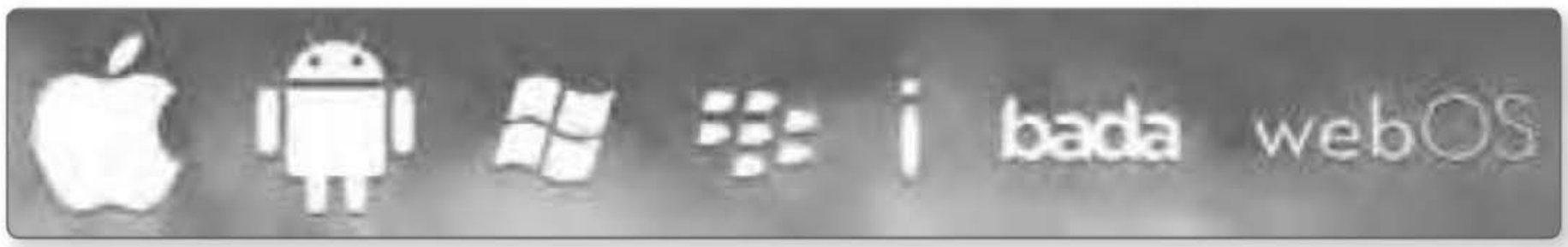
16.1. INTRODUCCIÓN



Gartner predice que HTML5 aumentará su dominio en el mercado de aplicaciones web móviles; lo que no significa que las aplicaciones nativas desaparezcan; siempre ofrecerán la mejor experiencia al usuario.

HTML5 es una especificación preliminar para la próxima versión importante de HTML. Representa un cambio con sus predecesores, HTML4 y XHTML. Algunos elementos se han eliminado y ya no se basa en SGML, un estándar antiguo para marcas .Hay muchas adiciones notables a HTML, como el soporte nativo de dibujo y elementos audiovisuales.

HTML5 es la plataforma ideal para la Web Móvil. Tanto si eres un desarrollador de contenido web para móviles como una empresa con unas determinadas necesidades comerciales o un desarrollador de juegos profesional, que quiere explorar las posibilidades de la Web como nueva plataforma.



16.2. REQUERIMIENTO DE SOFTWARE

Una vez instalado el JDK , SDK e IDE Eclipse, solo queda descargarlo. Se necesitará los siguientes requerimientos de software que se detallarán en los siguientes apartados.

16.2.1. PhoneGap

PhoneGap es un framework para el desarrollo de aplicaciones móviles producido por Nitobi y comprado posteriormente por Adobe Systems. Permite a los programadores desarrollar aplicaciones para dispositivos móviles utilizando herramientas genéricas tales como JavaScript, HTML5 y CSS3. Las aplicaciones resultantes son híbridas; es decir, que no son realmente aplicaciones nativas al dispositivo (ya que el renderizado es realizado mediante vistas web y no con interfaces gráficas específicas a cada sistema), pero no se trata tampoco de aplicaciones web (teniendo en cuenta que son aplicaciones empaquetadas, para poder ser desplegadas en el dispositivo incluso trabajando con el API del sistema nativo).



PhoneGap puede ser considerado como una distribución de Apache Cordova (software de código abierto). La aplicación fue primero llamada “PhoneGap” y luego “Apache Callback”.

A continuación deberá seguir los siguientes pasos:

Paso 1: Ingrese al link http://www.phonegap.com.

Paso 2: Haga doble clic en el ícono siguiente:

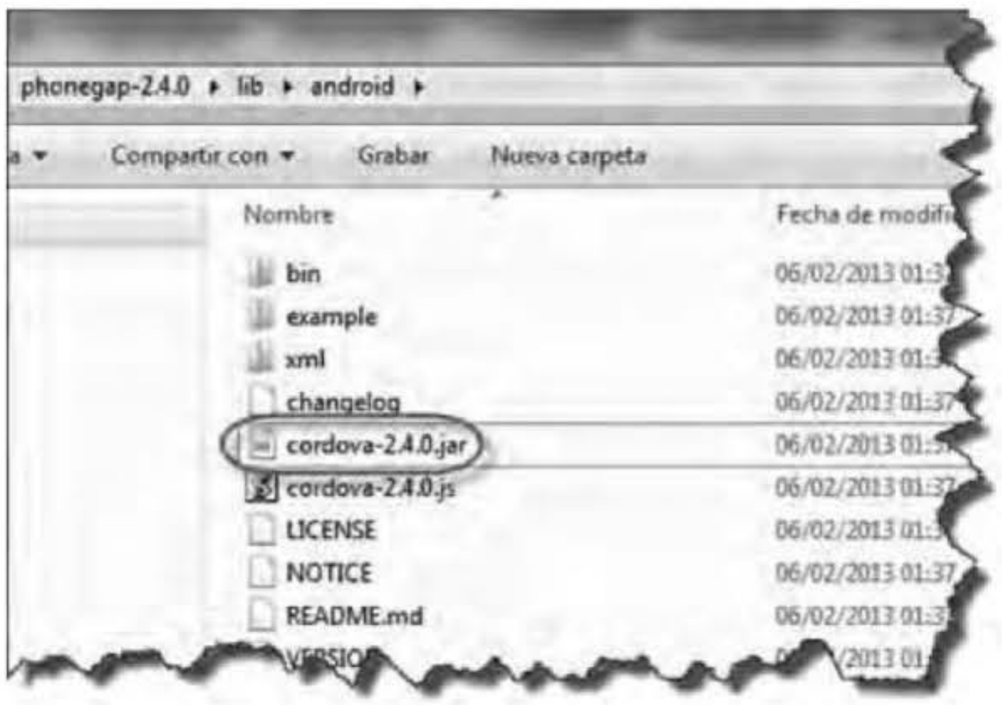


Paso 3: Presione clic en:



Paso 4: Guarde el archivo phonegap-2.4.0.zip, en alguna carpeta específica y luego descomprima.

Paso 5: Al descomprimir ubique la librería que más adelante utilizará.

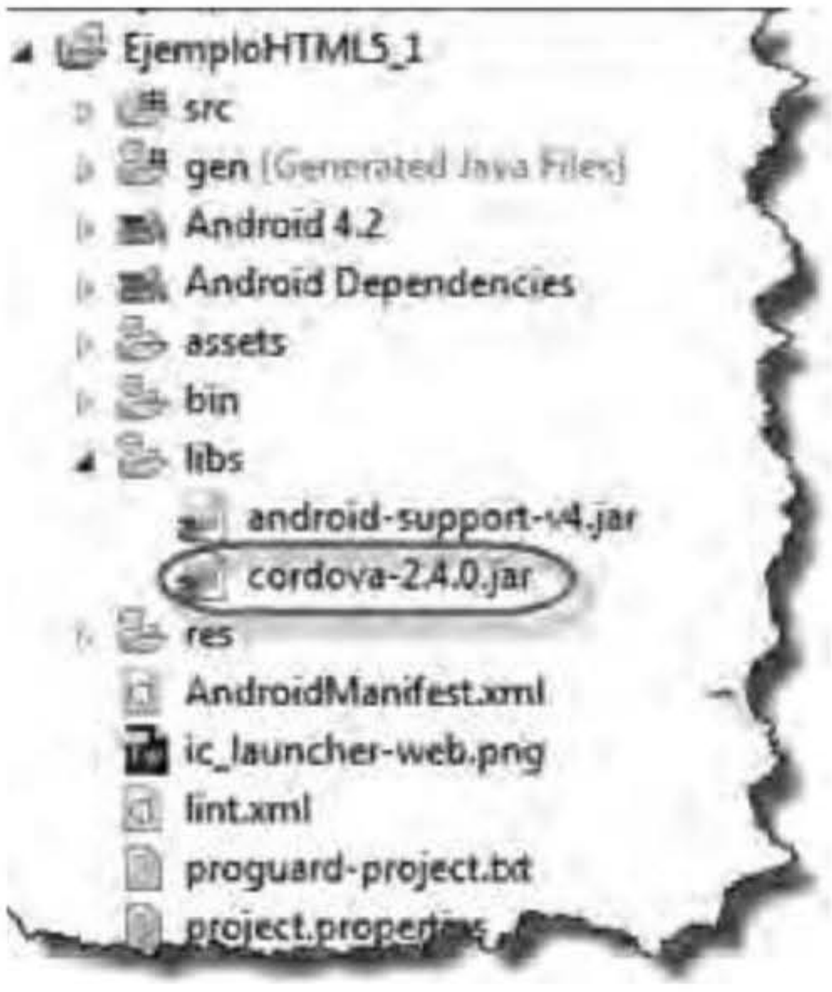




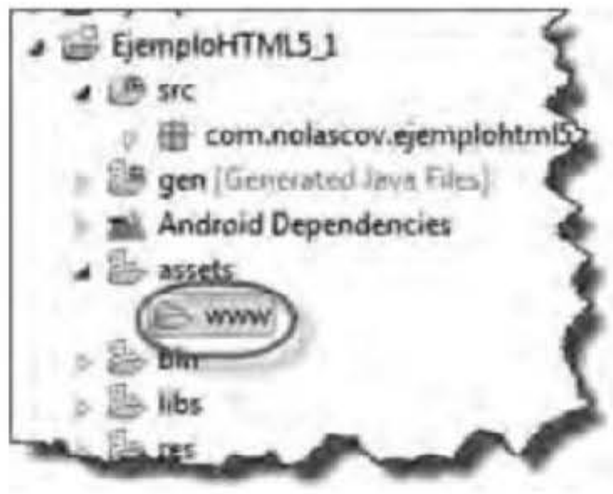
Paso 6: Diseñe una aplicación; la cual permita mostrar el texto "Hola mundo".

Paso 7: Después de haber creado la primera aplicación para Android, ahora cree un nuevo proyecto seleccionando Archivo > Nuevo > Android proyecto, el nombre de proyecto: EjemploHTML5_1 (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

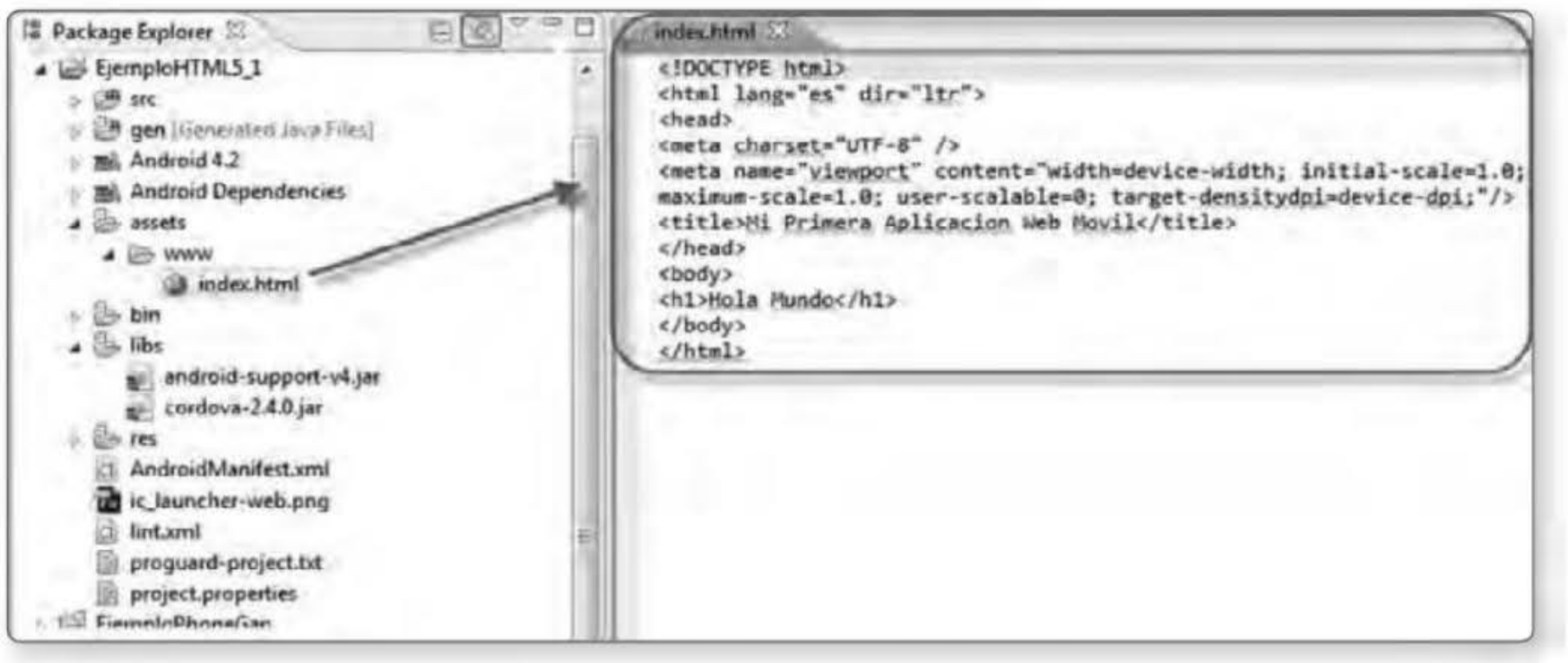
Paso 8: Almacene el fichero .jar en una carpeta y añádala a su proyecto como una librería externa (cordova-2.4.0.jar).



Paso 9: Cree dentro de la carpeta assets , la carpeta www.



Paso 10: El siguiente archivo se encontrará dentro de la carpeta www, index.html.



Index.html

```
<!DOCTYPE html>
<html lang=”es” dir=”ltr”>
<head>
<meta charset=”UTF-8” />
<meta name=”viewport” content=”width=device-width; initial-scale=1.0;
maximum-scale=1.0; user-scalable=0; target-densitydpi=device-dpi;”/>
<title>Mi Primera Aplicacion Web Movil</title>
</head>
<body>
<h1>Hola Mundo</h1>
</body>
</html>
```

Paso 11: Codifique en la actividad MainActivity.java lo siguiente:

```
package com.nolascov.ejemplohtml5_1;

import android.os.Bundle;
import org.apache.cordova.*;

public class MainActivity extends DroidGap
{
  public void onCreate(Bundle savedInstanceState)
  {
    super.onCreate(savedInstanceState);
    super.loadUrl(“file:///android_asset/www/index.html”);
  }
}
```

EJECUTANDO LA APLICACIÓN

Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:



16.3. CANVAS



Canvas es un elemento HTML incorporado en HTML5 que permite la generación de gráficos, dinámicamente por medio del scripting. Permite generar gráficos estáticos y animaciones. Fue implementado por Apple para su navegador Safari. Más tarde, fue adoptado por otros navegadores, como Firefox a partir de su versión 1.5, Opera y es adoptado por WHATWG.

Posee dos atributos width (ancho) y height (alto). El tamaño por defecto es 160. Para acceder al objeto canvas, es a través de Javascript; permitiendo generar gráficos 2D, juegos, animaciones y composición de imágenes. Asimismo, existe otra etiqueta SVG que cumple con funciones similares.

16.3.1. Conceptos Basicos sobre Canvas

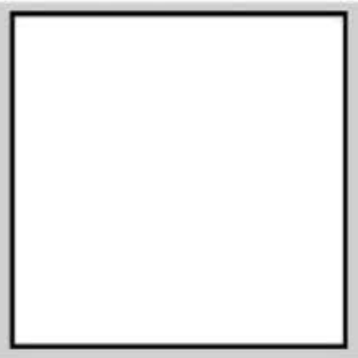
A. Crear un Canvas

```
<canvas id=”primerCanvas” width=”100” height=”100”></canvas>
```

Como se mencionó anteriormente, Canvas posee dos atributos width (ancho) y height (alto), el tamano por defecto es 150.

B. Definir bordes en un Canvas

```
<canvas id=”primerCanvas” width=”100” height=”100”
style=”border:1px solid #0000FF;”>
</canvas>
```

CÓDIGO FUENTE	VISTA PRELIMINAR
<html> <body> <canvas id=”primerCanvas” width=”100” height=”100” style=”border:1px solid #0000FF;”> </canvas> </body> </html>	

C. Dibujar con Canvas

CÓDIGO FUENTE	VISTA PRELIMINAR

<!DOCTYPE html>
<html>
 	
<body>
 	
<canvas id=”segundoCanvas” width=”100” height=”100”
style=”border:3px solid #FFFF00;”>
 	
</canvas>
 	
<script>
 	
var c=document.getElementById(“segundoCanvas”);
var ctx=c.getContext(“2d”);
 	
//define el color de relleno
ctx.fillStyle=”#FF0000”;
 	
// Dibuja un rectángulo con relleno
ctx.fillRect(0,0,100,100);
 	
</script>
 	
</body>
 	
</html>
	

Nota:

- *El método document.getElementById permite obtener la referencia a un elemento de la página mediante el ID de dicho elemento (en este caso al Canvas).*

```
var c=document.getElementById(“segundoCanvas”);
```

- *El método getContext y el segundo permite acceder al lienzo del Canvas, sobre el cual se va a dibujar un rectángulo con HTML5.*

```
var ctx=c.getContext(“2d”);
```

D. Dibujar sombras con Canvas

CÓDIGO FUENTE	VISTA PRELIMINAR

```
<!DOCTYPE html>

<html>

<body>

<canvas id=”tercerCanvas” width=”270” height=”120”
style=”border:3px solid #d3d3d3;”>
</canvas>

<script>

//referencia a un elemento de la pagina mediante
un id
var c=document.getElementById(“tercerCanvas”);

//acceder al lienzo del canvas
var ctx=c.getContext(“2d”);

//define el valor del tamaño de desenfoque
ctx.shadowBlur=30;

//define el color de relleno
ctx.fillStyle=”red”;

//define el color de sombra
ctx.shadowColor=”#FFFF00”;

//Dibuja un rectángulo con relleno
ctx.fillRect(20,20,100,80);

//define el color de sombra
ctx.shadowColor=”#FFFF00”;

// Dibuja un rectángulo con relleno
ctx.fillRect(140,20,100,80);

</script>

</body>

</html>
```

E. Dibujar texto gradiente con Canvas

CÓDIGO FUENTE	VISTA PRELIMINAR

```
<!DOCTYPE html>
<html>
<body>
<canvas id=”cuartoCanvas” width=”420” height=”100”
style=”border:1px solid #d3d3d3;”>
</canvas>
<script>
var c=document.getElementById(“cuartoCanvas”);
var ctx=c.getContext(“2d”);

ctx.font=”30px Verdana”;
// Create gradient
var       gradient=ctx.createLinearGradient(0,0,c.
width,0);
gradient.addColorStop(“0”,”magenta”);
gradient.addColorStop(“0.5”,”blue”);
gradient.addColorStop(“1.0”,”red”);
// asigna el degradado al strokeStyle
ctx.strokeStyle=gradient;
ctx.strokeText(“Texto con color gradiente”,10,50);

</script>

</body>
</html>
```


Para degradados, utilice este objeto para asignar a las propiedades fillStyle o strokeStyle. Cree un objeto canvasGradient mediante uno de los métodos siguientes:

```
El método createLinearGradient toma cuatro argumentos que representan el punto de
partida o inicial (x1, y1) y el punto final (x2, y2) del gradiente.
var gradient=ctx.createLinearGradient(0,0,c.width,0);
```

```
El método AddColorStop(position, color) Este método toma dos argumentos. La posición
debe ser un número entre 0.0 y 1.0 y define la posición relativa de los colores en el
degradado. El argumento de color debe ser una cadena que representa un color CSS (es
decir, # FFF, rgba (0,0,0,1))

gradient.addColorStop(“0”,”magenta”);
gradient.addColorStop(“0.5”,”blue”);
gradient.addColorStop(“1.0”,”red”);
```

```
asigna el degradado al strokeStyle
ctx.strokeStyle=gradient;
```

F. Manejo de texto en Canvas

CÓDIGO FUENTE	VISTA PRELIMINAR

```
<!DOCTYPE html>

<html>
<body>

<canvas id=”quintoCanvas” width=”230” height=”80”
style=”border:3px solid #90EE90;”>
</canvas>

<script>

var c=document.getElementById(“quintoCanvas”);
var ctx=c.getContext(“2d”);
ctx.font=”40px Arial”;
ctx.fillText(“Hola Amigos”,2,50);

</script>

</body>
</html>
```


◉ PROPIEDAD FONT

```
ctx.font=”40px Arial”;
```

TAMAÑO Y TIPO DE LETRAS	40px Arial (Valores por defecto 10px sans-serif)
SINTAXIS JAVASCRIPT	`context.font=”italic small-caps bold 12px arial”;`

◉ ALGUNOS VALORES DE LA PROPIEDAD FONT

VALOR	DESCRIPCIÓN
font-style	Especifica los estilos de Fuentes: • normal • italic • oblique
font-variant	Especifica la Variante de la fuente: • normal • small-caps
font-weight	Especifica el grosor de la fuente: • normal • bold • bolder • lighter • 100 • 200 • 300 • 400 • 500 • 600 • 700 • 800 • 900
font-size/line-height	Especifica el tamaño de fuente y la altura de la línea, en píxeles
font-family	Especifica la familia de fuentes
caption	Utilizado para los controles de fuente subtítulos (como botones, menús desplegables, etc)-
icon	Utilice la fuente utilizada para etiquetar iconos
Menu	Utilice el tipo de letra utilizado en los menús (menús desplegables y las listas de menú)
message-box	Utilice la fuente utilizada en los cuadros de diálogo
small-caption	

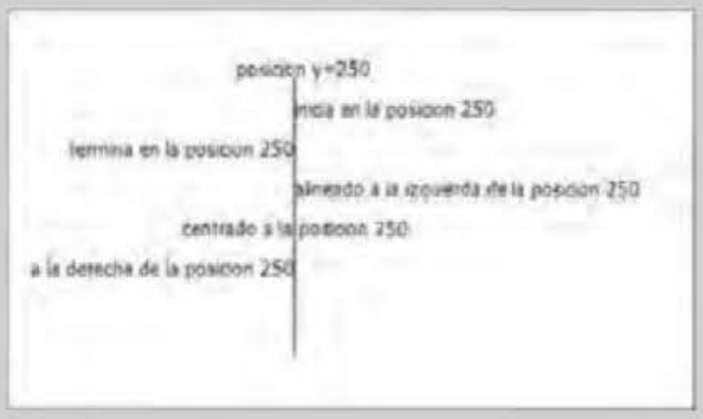
G. Alineación de texto

La propiedad textAlign establece o devuelve la alineación actual de contenido de texto, de acuerdo con el punto de anclaje. Normalmente, el texto comenzará en la posición especificada; sin embargo, si se establece textAlign = “right” y coloca el texto en la posición 200. Esto significa que el texto debe terminar en la posición 200.

ALINEACIÓN DE TEXTO EN CANVAS

CÓDIGO FUENTE	VISTA PRELIMINAR

```
<!DOCTYPE html8
<html>
<body>
<canvas id=”sextoCanvas” width=”600”
height=”300” style=”border:1px solid
#00FF7F;”>
</canvas>
<script>
var c=document.getElementById(“sextoCanvas”);
var ctx=c.getContext(“2d”);
//crea una linea en la posicion 250
ctx.strokeStyle=”blue”;
ctx.moveTo(250,50);
ctx.lineTo(250,260);
ctx.stroke();
ctx.font=”15px Verdana”;
ctx.fillText(“posicion y=250”,200,50);
//diferentes alineaciones
ctx.textAlign=”start”;
ctx.fillText(“inicia en la posicion
250”,250,80);
ctx.textAlign=”end”;
ctx.fillText(“termina en la posicion
250”,250,110);
ctx.textAlign=”left”;
ctx.fillText(“alineado a la izquierda de la
posicion 250”,250,140);
ctx.textAlign=”center”;
ctx.fillText(“centrado a la posicion
250”,250,170);
ctx.textAlign=”right”;
ctx.fillText(“a la derecha de la posicion
250”,250,200);
</script>
</body>
</html>
```



H. Escribir texto en Canvas

CÓDIGO FUENTE	VISTA PRELIMINAR

```
<!DOCTYPE html>
<html>
<body>

<canvas id=”septimoCanvas” width=”300” height=”150”
style=”border:3px solid #00BFFF;”>
</canvas>

<script>

var c=document.getElementById(“septimoCanvas”);

var ctx=c.getContext(“2d”);

ctx.font=”22px Verdana”;

ctx.fillText(“Texto Numero 1”,11,55);

ctx.font=”33px Verdana”;

// Crear gradiente
var       gradient=ctx.createLinearGradient(0,0,c.
width,0);
gradient.addColorStop(“0”,”red”);
gradient.addColorStop(“0.5”,”blue”);
gradient.addColorStop(“1.0”,”white”);

// llenado con el gradiente
ctx.fillStyle=gradient;
ctx.fillText(“Texto Numero 2”,10,90);

</script>

</body>

</html>
```


16.3.2. drawimage()

Método utilizado para dibujar imágenes. Para ello, se utiliza el método drawImage(), que pertenece al objeto contexto del Canvas, con la siguiente sintaxis:

```
drawImage(objeto_imagen, x, y)
```

Envíe tres parámetros: El primero, es el objeto Javascript de la imagen que se desea incluir en el Canvas. Los dos siguientes, son las coordenadas donde situar la imagen, siendo (X, Y) el punto donde se colocará la esquina superior izquierda de la imagen.

Cabe advertir que este método pertenece al objeto del Canvas; por lo que antes de poder invocarlo debe haber obtenido el contexto del Canvas.

CÓDIGO FUENTE	VISTA PRELIMINAR

```
<!DOCTYPE html>
<html>
<body>

<p>Imagen:</p>
<img id="inkadroid" src="inkadroid.jpg" alt="The
Scream" width="210" height="254">

<p>Canvas:</p>
<canvas id="octavoCanvas" width="260" height="300"
style="border:3px solid #EEE8AA;">
</canvas>

<script>

var c=document.getElementById("octavoCanvas");
var ctx=c.getContext("2d");
var img=document.getElementById("inkadroid");
ctx.drawImage(img,10,10);

</script>

</body>
</html>
```



16.4. MANEJO DE EVENTOS EN HTML5

Existen tres diferentes formas de registrar un evento para un elemento HTML:

- Agregar un nuevo atributo al el evento.
- Registrar un manejador de evento como una propiedad del evento.
- Usar el nuevo método estándar addEventListener().

16.4.1. Método addEventListener()

El método addEventListener() es la técnica ideal y la que es considerada como estándar por la especificación de HTML5. Por otro lado, este método tiene tres argumentos:

- Nombre del evento
- Función a ser ejecutada
- Valor lógico (falso o verdadero)

```
//escuchar eventos en el momento de cargar la pantalla
window.addEventListener('load', load, false);
```

La sintaxis del método addEventListener() es como sigue:

1. El primer atributo es el nombre del evento.
2. El segundo es la función a ser ejecutada; la cual puede ser una referencia a una función (como en este caso) o una función anónima.
3. El tercer atributo especificará usando true (verdadero) o false (falso), como múltiples eventos serán disparados.

16.5. EjemploHTML5_basico

Ahora se diseñará una aplicación que cual permita realizar algunos trazos.

Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando Archivo > Nuevo > Android proyecto.

Paso 2: Ingrese el nombre del proyecto: EjemploHtml5 (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 3: Cree dentro de la carpeta assets, la carpeta www.



Asimismo, dentro de la carpeta www, almacenará los javascrip (js).



Paso 4: Dentro de esta carpeta tendrá los siguientes ficheros:

- **MAIN.JS**

```
/*** MAIN ***/
//es el programa principal
//escuchar eventos en el momento de cargar la pantalla
window.addEventListener('load', load, false);

var WIN_WDT = window.innerWidth;
var WIN_HGT = window.innerHeight;
var TOOLBAR_HGT = 40;
var ctxToolbar;
var ctxCanvas;
var ctxPalette;
var ctxBrushes;
var mousedown = false;
var colorIdx = 16;
var colorTmp = colorIdx;
var bgcolorIdx = 0;
var bgcolorTmp = bgcolorIdx;
var lineWidth = 16;
var lineTmp = lineWidth;
var radius = lineWidth >> 1;
var currX;
var currY;
var lastX;
var lastY;
var img_button;
var buttonIdx = 0;
var BUTTON_WDT = 80;
var BUTTON_HGT = 31;
var MAX_RES = 1;
var countRes = 0;
//carga todas los recursos , imagenes y botones
function load() {
    loadResources();
//crear el objeto imagen
    img_button = new Image();
    img_button.onload = init;
//ruta de la iamgen
    img_button.src = 'images/button.png';
}


function loadResources() {
//  console.log('loadResources()');
    img_button = new Image();
    img_button.onload = countResources;
    img_button.src = 'images/button.png';
}
//cargador de imagenes
function countResources() {
//  console.log('countResources()');
    countRes++;
    if(countRes = MAX_RES) {
            init();
    }
}
//incializa la pantalla
function init() {
```

```
        BUTTON_WDT = img_button.width;
        BUTTON_HGT = img_button.height;

        WIN_WDT = window.innerWidth;
        WIN_HGT = window.innerHeight;
        initToolbar();
        initCanvas();
        initPalette();
        initBrushes();
    }



    // dibujo un boton
    function drawButton(ctx, x, y, text, hover) {
        var px;
        if(hover) {
                ctx.fillStyle = '#FF8000';
        } else {
                ctx.fillStyle = '#404040';
        }
        ctx.fillRect(x-2, y-2, BUTTON_WDT+4, BUTTON_HGT+4);
        ctx.drawImage(img_button, x, y);
        px = x + ((BUTTON_WDT - ctx.measureText(text).width) >> 1)
        ctx.fillStyle = '#000000';
        ctx.fillText(text,px+1,y+8);
        ctx.fillStyle = '#FFFFFF';
        ctx.fillText(text,px,y+7);
    }

    // pregunta si x y esta dentro de un rectangulo ( boton)
    function inside( px, py, x, y, wdt, hgt ) {
        if( (px >= x) && (px < (x+wdt)) && (py >= y) && (py < (y+hgt)) ) {
                return true;
        } else {
                return false;
        }
    }
    // rellena con color la brocha actual
    function ctxSetColor(ctx) {
        imgData = ctx.getImageData(0, 0, ctx.canvas.width, ctx.canvas.height);
        size = imgData.width * imgData.height;
        pixel = imgData.data;
        pos = 0;
        r = paleta[colorIdx][0];
        g = paleta[colorIdx][1];
        b = paleta[colorIdx][2];
        for(i=0; i < size; i++ ) {
                pixel[pos + 0] = r;
                pixel[pos + 1] = g;
                pixel[pos + 2] = b;
                pos += 4;
        }
        ctx.putImageData(imgData, 0, 0);
    }
    // establece los niveles de alfa de la brocha
    function ctxSetAlpha(ctx) {
        imgData = ctx.getImageData(0, 0, ctx.canvas.width, ctx.canvas.height);
        size = imgData.width * imgData.height;
        pixel = imgData.data;
```

```
        pos = 0;
        for(i=0; i < size; i++ ) {
                gray = (pixel[pos + 0] * 0.299) + (pixel[pos + 1] * 0.587) + (pixel[pos +
2] * 0.114);
                // gray = Math.floor((pixel[pos + 0] * 0.299) + (pixel[pos + 1] * 0.587) +
(pixel[pos + 2] * 0.114));
                // if( gray > 255 ) {
                //      console.log(gray);
                // }
                if( pixel[pos + 3] > 0 ) {
//                      pixel[pos + 3] = (255 - gray) >> 1;
                        pixel[pos + 3] = 255 - gray;
                }
                pos += 4;
        }
        ctx.putImageData(imgData, 0, 0);
}
//paleta de colores
var paleta =
[[255,255,255], // white
[245,181,107],
[236,166,92],
[227,151,78],
[217,136,65],
[208,123,53],
[199,108,41],
[192,95,30],
[186,92,28],
[179,87,26],
[173,84,23],
[167,77,22],
[160,72,20],
[154,69,18],
[147,63,16],
[141,59,14],
[135,56,13],
[128,52,11],
[123,47,10],
[116,44,9],
[110,41,7],
[104,38,6],
[97,33,5],
[91,31,4],
[84,28,3],
[78,24,3],
[72,21,2],
[65,19,1],
[59,17,0],
[52,14,0],
[46,12,0],
[0,0,0], // black
[246,161,71],
[237,162,82],
[229,159,92],
[220,159,101],
[202,140,85],
[186,126,71],
[170,114,58],
[154,99,46],
[138,86,35],
```

```
    [123,73,26],
    [107,62,18],
    [91,50,11],
    [75,41,6],
    [59,31,2],
    [43,22,0],
    [255,214,173],
    [241,196,153],
    [228,181,134],
    [215,163,118],
    [202,148,102],
    [184,137,100],
    [168,128,97],
    [152,118,93],
    [136,108,88],
    [121,97,82],
    [106,87,75],
    [90,75,67],
    [74,63,58],
    [58,51,47],
    [42,37,35],
    [26,24,23],
    [16,16,16],
    [117,19,0],
    [82,7,0],
    [46,0,0],
    [218,255,255],
    [255,198,124],
    [245,181,107],
    [236,166,92],
    [227,151,78],
    [217,136,65],
    [208,123,53],
    [199,108,41],
    [192,95,30],
    [186,92,28],
    [179,87,26],
    [173,84,23],
    [167,77,22],
    [160,72,20],
    [154,69,18],
    [147,63,16],
    [141,59,14],
    [135,56,13],
    [128,52,11],
    [123,47,10],
    [116,44,9],
    [110,41,7],
    [104,38,6],
    [97,33,5],
    [91,31,4],
    [84,28,3],
    [78,24,3],
    [72,21,2],
    [65,19,1],
    [59,17,0],
    [52,14,0],
    [46,12,0],
    [255,165,59],
    [246,161,71],
```

```
    [237,162,82],
    [229,159,92],
    [220,159,101],
    [202,140,85],
    [186,126,71],
    [170,114,58],
    [154,99,46],
    [138,86,35],
    [123,73,26],
    [107,62,18],
    [91,50,11],
    [75,41,6],
    [59,31,2],
    [43,22,0],
    [255,214,173],
    [241,196,153],
    [228,181,134],
    [215,163,118],
    [202,148,102],
    [184,137,100],
    [168,128,97],
    [152,118,93],
    [136,108,88],
    [121,97,82],
    [106,87,75],
    [90,75,67],
    [74,63,58],
    [58,51,47],
    [42,37,35],
    [26,24,23],
    [152,39,0],
    [117,19,0],
    [82,7,0],
    [46,0,0],
    [218,255,255],
    [255,198,124],
    [245,181,107],
    [236,166,92],
    [227,151,78],
    [217,136,65],
    [208,123,53],
    [199,108,41],
    [192,95,30],
    [186,92,28],
    [179,87,26],
    [173,84,23],
    [167,77,22],
    [160,72,20],
    [154,69,18],
    [147,63,16],
    [141,59,14],
    [135,56,13],
    [128,52,11],
    [123,47,10],
    [116,44,9],
    [110,41,7],
    [104,38,6],
    [97,33,5],
    [91,31,4],
    [84,28,3],
```

```
    [78,24,3],
    [72,21,2],
    [65,19,1],
    [59,17,0],
    [52,14,0],
    [46,12,0],
    [255,165,59],
    [246,161,71],
    [237,162,82],
    [229,159,92],
    [220,159,101],
    [202,140,85],
    [186,126,71],
    [170,114,58],
    [154,99,46],
    [138,86,35],
    [123,73,26],
    [107,62,18],
    [91,50,11],
    [75,41,6],
    [59,31,2],
    [43,22,0],
    [255,214,173],
    [241,196,153],
    [228,181,134],
    [215,163,118],
    [202,148,102],
    [184,137,100],
    [168,128,97],
    [152,118,93],
    [136,108,88],
    [121,97,82],
    [106,87,75],
    [90,75,67],
    [74,63,58],
    [58,51,47],
    [42,37,35],
    [26,24,23],
    [152,39,0],
    [117,19,0],
    [82,7,0],
    [46,0,0],
    [218,255,255],
    [255,198,124],
    [245,181,107],
    [236,166,92],
    [227,151,78],
    [217,136,65],
    [208,123,53],
    [199,108,41],
    [192,95,30],
    [186,92,28],
    [179,87,26],
    [173,84,23],
    [167,77,22],
    [160,72,20],
    [154,69,18],
    [147,63,16],
    [141,59,14],
    [135,56,13],
```

```
    [128,52,11],
    [123,47,10],
    [116,44,9],
    [110,41,7],
    [104,38,6],
    [97,33,5],
    [91,31,4],
    [84,28,3],
    [78,24,3],
    [72,21,2],
    [65,19,1],
    [59,17,0],
    [52,14,0],
    [46,12,0],
    [255,165,59],
    [246,161,71],
    [237,162,82],
    [229,159,92],
    [220,159,101],
    [202,140,85],
    [186,126,71],
    [170,114,58],
    [154,99,46],
    [138,86,35],
    [123,73,26],
    [107,62,18],
    [91,50,11],
    [75,41,6],
    [59,31,2],
    [43,22,0],
    [255,214,173],
    [241,196,153],
    [228,181,134],
    [215,163,118],
    [202,148,102]];
```

- **BRUSHES.JS**

```
/*** BRUSHES DIALOG ***/
//brushes la pantalla para definir el tamaño de la brocha
// la posiciones y el numero maximo de botones
BRUSHES_BTN_Y = 32;
BRUSHES_BTN_MAX = 4;
BRUSHES_BTN_LEFT = 0;
BRUSHES_BTN_RIGHT = 1;
BRUSHES_BTN_OK = 2;
BRUSHES_BTN_CANCEL = 3;

var brushes_btn_pos = new Array(BRUSHES_BTN_MAX);
var brushes_btn_label = ['-','+','Ok','Cancel'];

//inicializa la pantalla
function initBrushes() {
    temp=document.getElementById('brushes');
    ctxBrushes=temp.getContext('2d');
    //crear un canvas de toda la pantalla
    ctxBrushes.canvas.width = WIN_WDT;
    ctxBrushes.canvas.height = WIN_HGT;
```

```
    //el tipo de letra para el canvas
    ctxBrushes.font = 'bold 16px Myriad Pro';
    //la linie base del texto
    ctxBrushes.textBaseline = 'top';
    //dibuja la brocha
    drawBrushes();
    //inicializar los botones
    initBrushesButtons();
    //los lintener para los diferentes eventos
    ctxBrushes.canvas.addEventListener('mousedown', brushes_mousedown, false);
    ctxBrushes.canvas.addEventListener('mousemove', brushes_mousemove, false);
    ctxBrushes.canvas.addEventListener('mouseup', brushes_mouseup, false);
    ctxBrushes.canvas.addEventListener('touchstart', brushes_touchstart, false);
    ctxBrushes.canvas.addEventListener('touchmove', brushes_touchmove, false);
    ctxBrushes.canvas.addEventListener('touchend', brushes_touchend, false);
    ctxBrushes.canvas.addEventListener('touchcancel', brushes_touchend, false);
}

//inicializa  los botones en las posiciones XY
function initBrushesButtons() {
    brushes_btn_pos[0] = [200,BRUSHES_BTN_Y];
    brushes_btn_pos[1] = [340,BRUSHES_BTN_Y];
    brushes_btn_pos[2] = [600,BRUSHES_BTN_Y];
    brushes_btn_pos[3] = [720,BRUSHES_BTN_Y];
    drawButton(ctxBrushes,    brushes_btn_pos[BRUSHES_BTN_LEFT][0],    brushes_btn_
pos[BRUSHES_BTN_LEFT][1], brushes_btn_label[BRUSHES_BTN_LEFT], false);
    drawButton(ctxBrushes,    brushes_btn_pos[BRUSHES_BTN_RIGHT][0],    brushes_btn_
pos[BRUSHES_BTN_LEFT][1], brushes_btn_label[BRUSHES_BTN_RIGHT], false);
    drawButton(ctxBrushes,    brushes_btn_pos[BRUSHES_BTN_OK][0],    brushes_btn_
pos[BRUSHES_BTN_LEFT][1], brushes_btn_label[BRUSHES_BTN_OK], false);
    drawButton(ctxBrushes,    brushes_btn_pos[BRUSHES_BTN_CANCEL][0],    brushes_btn_
pos[BRUSHES_BTN_LEFT][1], brushes_btn_label[BRUSHES_BTN_CANCEL], false);
}

//pinta el texto de color blanco
function drawBrushes() {
    ctxBrushes.fillStyle = '#FFFFFF';
    ctxBrushes.fillText('Brush Size', 120, 38);
    drawWidth();
}

//tamaño de la brocha
function drawWidth() {
    ctxBrushes.fillStyle = '#404040';
    ctxBrushes.fillRect(288, 38, 48, 20);
    ctxBrushes.fillStyle = '#FFFFFF';
    ctxBrushes.fillText(lineTmp+' px', 288, 38);
}

//evento mouse
function brushes_mousedown(event) {
  brushes_pressed(event.pageX, event.pageY);
}

// evento touch
function brushes_touchstart(event) {
    event.preventDefault();
    var touch = event.changedTouches[0];
    brushes_pressed(touch.pageX, touch.pageY);
}
```

```
//evento manejo de teclas
function brushes_pressed(x,y) {
    if(x || y) {
            mousedown = true;
            // if( inside(x, y, BRUSHES_X, BRUSHES_Y, BRUSHES_WDT, BRUSHES_HGT) ) {
            //      selectColor(x,y);
            // } else {
                    buttonIdx = -1;
                    // que bonto ha sido presionado 0,1,2,3
                    for(i=0; i < BRUSHES_BTN_MAX; i++) {
                        if(  inside(x,  y,  brushes_btn_pos[i][0],  brushes_btn_
pos[i][1], BUTTON_WDT, BUTTON_HGT) ) {
                            buttonIdx = i;
                            break;
                        }
                    }
                    if( buttonIdx > -1 ) {
                        drawButton(ctxBrushes,    brushes_btn_pos[buttonIdx][0],
brushes_btn_pos[buttonIdx][1], brushes_btn_label[buttonIdx], true);
                    }
            // }
    }
}

// evento mousemove
function brushes_mousemove(event) {
  brushes_dragged(event.pageX, event.pageY);
}

//evento touchmove
function brushes_touchmove(event) {
    event.preventDefault();
    var touch = event.changedTouches[0];
    brushes_dragged(touch.pageX, touch.pageY);
}

// evento dragged
function brushes_dragged(x,y) {
}

// evento mouseup
function brushes_mouseup(event) {
    brushes_released(0,0);
}

//Evento dejo de tocar la pantalla
function brushes_touchend(event) {
    brushes_released(0,0);
}

// Evento cuando existe inactividad
function brushes_released(x,y) {
    mousedown = false;
    if( buttonIdx > -1 ) {
            drawButton(ctxBrushes,    brushes_btn_pos[buttonIdx][0],    brushes_btn_
pos[buttonIdx][1], brushes_btn_label[buttonIdx], false);
            brushes_action(buttonIdx);
    }
}
```

```
//Manejo de acciónes de cada boton
function brushes_action(idx) {
    switch(idx) {
            case BRUSHES_BTN_LEFT:
                    if( lineTmp > 2 ) {
                            lineTmp -= 2;
                            drawWidth();
                    }
                    break;
            case BRUSHES_BTN_RIGHT:
                    if( lineTmp < 32 ) {
                            lineTmp += 2;
                            drawWidth();
                    }
                    break;
            case BRUSHES_BTN_OK:
                    lineWidth = lineTmp;
                    radius = Math.floor(lineWidth /2);
                    document.getElementById('brushes').style.display = 'none';
            break;
            case BRUSHES_BTN_CANCEL:
                    document.getElementById('brushes').style.display = 'none';
            break;
    }
}
```

CANVAS.JS

```
/*** CANVAS ***/
//canvas es la pantalla de dibujo
var img_ball = new Image();
img_ball.onload = initBrush;
//img_ball.src = 'gfx/commodore64.png';
//img_ball.src = 'images/brushes/ball.png';
img_ball.src = 'images/brushes/brush2.png';
ofsX = 0;
ofsY = 0;
//inicializa la pantalla donde dibujar o area de dibujo
function initCanvas() {
    temp=document.getElementById('canvas');
    ctxCanvas=temp.getContext('2d');
    ctxCanvas.canvas.width = WIN_WDT;
    ctxCanvas.canvas.height = WIN_HGT - TOOLBAR_HGT;
    ctxCanvas.canvas.style.background = '#FFFFFF';
    ctxCanvas.canvas.addEventListener('mousedown', canvas_mousedown, false);
    ctxCanvas.canvas.addEventListener('mousemove', canvas_mousemove, false);
    ctxCanvas.canvas.addEventListener('mouseup', canvas_mouseup, false);
    ctxCanvas.canvas.addEventListener('touchstart', canvas_touchstart, false);
    ctxCanvas.canvas.addEventListener("touchmove", canvas_touchmove, false);
    ctxCanvas.canvas.addEventListener('touchend', canvas_touchend, false);
    ctxCanvas.canvas.addEventListener('touchcancel', canvas_touchend, false);
    ctxCanvas.textBaseline = 'top';
    ctxCanvas.fillStyle = '#000';


}
//inicializa la pantalla
function initBrush() {
```

```
        brush = document.createElement('canvas');
        ctxBrush = brush.getContext('2d');
        ctxBrush.canvas.width = img_ball.width;
        ctxBrush.canvas.height = img_ball.height;
        ctxBrush.drawImage(img_ball, 0, 0);
        ctxBrush.transform(2.0, 0, 0, 2.0, 0, 0);
        ctxSetAlpha(ctxBrush);
        ctxSetColor(ctxBrush);
        ofsX = -(ctxBrush.canvas.width >> 1);
        ofsY = -(ctxBrush.canvas.height >> 1);
    }

    function canvas_mousedown(event) {
      canvas_pressed(event.pageX, event.pageY);
    }

    function canvas_touchstart(event) {
        event.preventDefault();
        var touch = event.changedTouches[0];
        canvas_pressed(touch.pageX, touch.pageY);
    }
    // dibujar el punto incial
    function canvas_pressed(x,y) {
        if(x || y) {
                mousedown = true;
                currX = x;
                currY = y - TOOLBAR_HGT;
                lastX = currX;
                lastY = currY;
                drawPixel(currX, currY);
        }
    }

    function canvas_mousemove(event) {
        if( mousedown ) {
          canvas_dragged(event.pageX, event.pageY);
        }
    }
    //evento arrastre del mouse
    function canvas_touchmove(event) {
        if( mousedown ) {
                event.preventDefault();
          var touch = event.changedTouches[0];
          canvas_dragged(touch.pageX, touch.pageY);
        }
    }

    function canvas_dragged(x,y) {
        if(x || y) {
                currX = x;
                currY = y - TOOLBAR_HGT;
                drawLine();
                lastX = currX;
                lastY = currY;
        }
    }

    function canvas_mouseup(event) {
        canvas_release(0,0);
    }
```

```
function canvas_touchend(event) {
    canvas_release(0,0);
}

function canvas_release(x,y) {
    mousedown = false;
}

function drawPixel(x,y) {
    ctxCanvas.drawImage(brush, x + ofsX, y + ofsY);
}
//dibuja una linea desde el punto actual hasta el incial
function drawLine() {
    lenX = currX - lastX;
    lenY = currY - lastY;
    vx = 0;
    if(lenX > 0) {
            vx = 1;
    } else if(lenX < 0) {
            vx = -1;
    }
    vy = 0;
    if(lenY > 0) {
            vy = 1;
    } else if(lenY < 0) {
            vy = -1;
    }
    len = 0;
    lenX = Math.abs(lenX);
    lenY = Math.abs(lenY);
    if(lenX > lenY) {
            vy = (lenY / lenX) * vy;
            len = lenX;
    } else {
            vx = (lenX / lenY) * vx;
            len = lenY;
    }
    x = lastX;
    y = lastY;
    for(i=0; i < len; i++) {
            drawPixel(Math.floor(x),Math.floor(y));
            x += vx;
            y += vy;
    }
}
```

- **PALETTE.JS**

```
/*** PALETTE DIALOG ***/

//paleta de colores

COLOR_WDT = 30;
COLOR_HGT = 40;
PALETTE_WDT = COLOR_WDT*32;
PALETTE_HGT = COLOR_HGT*8;
PALETTE_X = (WIN_WDT - PALETTE_WDT) >> 1;
PALETTE_Y = 248;
```

```
SLIDER_WDT = 254;
SLIDER_HGT = COLOR_HGT - 2;

PALETTE_BTN_Y = 32;
PALETTE_BTN_MAX = 2;
PALETTE_BTN_SPACING = 0;
PALETTE_BTN_OK = 0;
PALETTE_BTN_CANCEL = 1;

var palette_btn_pos = new Array(TOOLBAR_BTN_MAX);
var palette_btn_label = ['Ok','Cancel'];
var palX;
var palY;
var palLx;
var palLy;
// crear canvas o la pantalla de la paleta
function initPalette() {
    temp=document.getElementById('palette');
    ctxPalette=temp.getContext('2d');
    ctxPalette.canvas.width = WIN_WDT;
    ctxPalette.canvas.height = WIN_HGT;
    ctxPalette.font = 'bold 16px Myriad Pro';
    ctxPalette.textBaseline = 'top';
    drawPalette();
    initPaletteButtons();

    ctxPalette.canvas.addEventListener('mousedown', palette_mousedown, false);
    ctxPalette.canvas.addEventListener('mousemove', palette_mousemove, false);
    ctxPalette.canvas.addEventListener('mouseup', palette_mouseup, false);
    ctxPalette.canvas.addEventListener('touchstart', palette_touchstart, false);
    ctxPalette.canvas.addEventListener('touchmove', palette_touchmove, false);
    ctxPalette.canvas.addEventListener('touchend', palette_touchend, false);
    ctxPalette.canvas.addEventListener('touchcancel', palette_touchend, false);
}
// posicionar los botones
function initPaletteButtons() {

    palette_btn_pos[0] = [600,PALETTE_BTN_Y];
    palette_btn_pos[1] = [720,PALETTE_BTN_Y];
    //dibujar los botones
    drawButton(ctxPalette,      palette_btn_pos[PALETTE_BTN_OK][0],      palette_btn_
pos[PALETTE_BTN_OK][1],palette_btn_label[PALETTE_BTN_OK], false);
    drawButton(ctxPalette,    palette_btn_pos[PALETTE_BTN_CANCEL][0],    palette_btn_
pos[PALETTE_BTN_OK][1],palette_btn_label[PALETTE_BTN_CANCEL], false);
}
//dibuja un cuadrado del color principal
function drawColor(index) {
    ctxPalette.fillStyle = 'rgb('+paleta[index]+')';
    ctxPalette.fillRect(320,32,128,128);
}
//dibuja un cuadrado del color secundario o fondo
function drawBgColor(index) {
    ctxPalette.fillStyle = 'rgb('+paleta[index]+')';
    ctxPalette.fillRect(368,80,128,128);
}
//dibuja la paleta de colores
function drawPalette() {
    // ctxPalette.lineWidth = 1;
    // ctxPalette.strokeStyle = '#FFFFFF';
    // ctxPalette.strokeRect(32, 32, SLIDER_WDT, SLIDER_HGT);
```

```
// ctxPalette.strokeRect(32, 80, SLIDER_WDT, SLIDER_HGT);
// ctxPalette.strokeRect(32, 128, SLIDER_WDT, SLIDER_HGT);
// ctxPalette.strokeRect(32, 176, SLIDER_WDT, SLIDER_HGT);
pal = 0;
y = PALETTE_Y;
for( j=0; j<8; j++ ) {
        x = PALETTE_X;
        for( i=0; i<32; i++ ) {
                ctxPalette.fillStyle = 'rgb('+paleta[pal]+')';
                ctxPalette.fillRect(x, y, COLOR_WDT, COLOR_HGT);
                x += COLOR_WDT;
                pal++;
        }
        y += COLOR_HGT;
    }
    drawBgColor(bgcolorTmp);
    drawColor(colorTmp);
}

function palette_mousedown(event) {
  palette_pressed(event.pageX, event.pageY);
}

function palette_touchstart(event) {
    event.preventDefault();
    var touch = event.changedTouches[0];
    palette_pressed(touch.pageX, touch.pageY);
}
// Evento de manejo de botones
function palette_pressed(x,y) {
    if(x || y) {
            mousedown = true;
            buttonIdx = -1;
            if( inside(x, y, PALETTE_X, PALETTE_Y, PALETTE_WDT, PALETTE_HGT) ) {
                    selectColor(x,y);
            } else if( inside(x, y, 368, 160, 128, 48) || inside(x, y, 448, 80, 48, 80)
) {
                    swapColors();
            } else {
                    for(i=0; i < PALETTE_BTN_MAX; i++) {
                            if(  inside(x,  y,  palette_btn_pos[i][0],  palette_btn_
pos[i][1], BUTTON_WDT, BUTTON_HGT) ) {
                                    buttonIdx = i;
                                    break;
                            }
                    }
                    if( buttonIdx > -1 ) {
                            drawButton(ctxPalette,    palette_btn_pos[buttonIdx][0],
palette_btn_pos[buttonIdx][1], palette_btn_label[buttonIdx], true);
                    }
            }
    }
}

function palette_mousemove(event) {
  palette_dragged(event.pageX, event.pageY);
}

function palette_touchmove(event) {
    event.preventDefault();
```

```
    var touch = event.changedTouches[0];
    palette_dragged(touch.pageX, touch.pageY);
}

function palette_dragged(x,y) {
    if(x || y) {
            if( mousedown && inside(x, y, PALETTE_X, PALETTE_Y, PALETTE_WDT, PALETTE_
HGT) ) {
                    selectColor(x,y);
            }
    }
}
//cambia el color del cuadrado principal al presionar la paleta
function selectColor(x,y) {
    x = Math.floor((x - PALETTE_X) / COLOR_WDT);
    y = Math.floor((y - PALETTE_Y) / COLOR_HGT);
    colorTmp = (y << 5) | x;
    drawColor(colorTmp);
}
//intercambio del color principal y secundario
function swapColors() {
    temp = colorTmp;
    colorTmp = bgcolorTmp;
    bgcolorTmp = temp;
    drawBgColor(bgcolorTmp);
    drawColor(colorTmp);
}

function palette_mouseup(event) {
    palette_released(0,0);
}

function palette_touchend(event) {
    palette_released(0,0);
}
// Evento de inactividad
function palette_released(x,y) {
    mousedown = false;
    if( buttonIdx > -1 ) {
            drawButton(ctxPalette,    palette_btn_pos[buttonIdx][0],    palette_btn_
pos[buttonIdx][1], palette_btn_label[buttonIdx], false);
            palette_action(buttonIdx);
    }
}
//acciones de los botones
function palette_action(idx) {
    switch(idx) {
            case PALETTE_BTN_OK:
                    colorIdx = colorTmp;
                    bgcolorIdx = bgcolorTmp;
                    ctxSetColor(ctxBrush);
                    document.getElementById('palette').style.display = 'none';
            break;
            case PALETTE_BTN_CANCEL:
                    drawBgColor(bgcolorIdx);
                    drawColor(colorIdx);
                    document.getElementById('palette').style.display = 'none';
            break;
    }
}
```

TOOLBAR.JS

```
/*** TOOLBAR ***/
//la barra superior de botones

TOOLBAR_BTN_Y = 4;
TOOLBAR_BTN_MAX = 7;
TOOLBAR_BTN_SPACING = 0;
TOOLBAR_BTN_NEW = 0;
TOOLBAR_BTN_LOAD = 1;
TOOLBAR_BTN_SAVE = 2;
TOOLBAR_BTN_SHARE = 3
TOOLBAR_BTN_BRUSHES = 4;
TOOLBAR_BTN_COLORS = 5;
TOOLBAR_BTN_HELP = 6;

var toolbar_btn_posx = new Array(TOOLBAR_BTN_MAX);
var  toolbar_btn_label  =  ['Nuevo','Abrir','Guardar','Redes','Paleta','Colores','Ayu
da'];

function initToolbar() {
    temp=document.getElementById('toolbar');
    ctxToolbar=temp.getContext('2d');
    ctxToolbar.canvas.width = WIN_WDT;
    ctxToolbar.canvas.height = TOOLBAR_HGT;
    ctxToolbar.canvas.style.background = '#404040';
    ctxToolbar.font = 'bold 16px Myriad Pro';
    ctxToolbar.textBaseline = 'top';
    ctxToolbar.fillStyle = '#000000';
    ctxToolbar.fillRect(0, TOOLBAR_HGT-1, WIN_WDT, TOOLBAR_HGT-1);
    initToolbarButtons();

    ctxToolbar.canvas.addEventListener('mousedown', toolbar_mousedown, false);
    ctxToolbar.canvas.addEventListener('mousemove', toolbar_mousemove, false);
    ctxToolbar.canvas.addEventListener('mouseup', toolbar_mouseup, false);
    ctxToolbar.canvas.addEventListener('touchstart', toolbar_touchstart, false);
    ctxToolbar.canvas.addEventListener("touchmove", toolbar_touchmove, false);
    ctxToolbar.canvas.addEventListener('touchend', toolbar_touchend, false);
}
//inicializa los botones de la barra principal
function initToolbarButtons() {
    free = WIN_WDT - (BUTTON_WDT * TOOLBAR_BTN_MAX);
    margin = (free % (TOOLBAR_BTN_MAX + 1)) >> 1;
    TOOLBAR_BTN_SPACING = Math.floor( free / (TOOLBAR_BTN_MAX + 1) );
    pos = margin + TOOLBAR_BTN_SPACING;
    for(i=0; i<TOOLBAR_BTN_MAX; i++) {
            toolbar_btn_posx[i] = pos;
            pos += (BUTTON_WDT + TOOLBAR_BTN_SPACING);
    }
    drawButton(ctxToolbar, toolbar_btn_posx[TOOLBAR_BTN_NEW], TOOLBAR_BTN_Y, toolbar_
btn_label[TOOLBAR_BTN_NEW], false);
    drawButton(ctxToolbar, toolbar_btn_posx[TOOLBAR_BTN_LOAD], TOOLBAR_BTN_Y, toolbar_
btn_label[TOOLBAR_BTN_LOAD], false);
    drawButton(ctxToolbar, toolbar_btn_posx[TOOLBAR_BTN_SAVE], TOOLBAR_BTN_Y, toolbar_
btn_label[TOOLBAR_BTN_SAVE], false);
    drawButton(ctxToolbar,    toolbar_btn_posx[TOOLBAR_BTN_SHARE],    TOOLBAR_BTN_Y,
toolbar_btn_label[TOOLBAR_BTN_SHARE], false);
    drawButton(ctxToolbar,   toolbar_btn_posx[TOOLBAR_BTN_BRUSHES],   TOOLBAR_BTN_Y,
toolbar_btn_label[TOOLBAR_BTN_BRUSHES], false);
```

```
    drawButton(ctxToolbar,    toolbar_btn_posx[TOOLBAR_BTN_COLORS],    TOOLBAR_BTN_Y,
toolbar_btn_label[TOOLBAR_BTN_COLORS], false);
    drawButton(ctxToolbar, toolbar_btn_posx[TOOLBAR_BTN_HELP], TOOLBAR_BTN_Y, toolbar_
btn_label[TOOLBAR_BTN_HELP], false);
}

function toolbar_mousedown(event) {
    toolbar_pressed(event.pageX, event.pageY);
}

function toolbar_touchstart(event) {
    event.preventDefault();
    var touch = event.changedTouches[0];
    toolbar_pressed(touch.pageX, touch.pageY);
}

function toolbar_pressed(x,y) {
    mousedown = true;
    buttonIdx = -1;
    for(i=0; i < TOOLBAR_BTN_MAX; i++) {
            if( inside(x, y, toolbar_btn_posx[i], TOOLBAR_BTN_Y, BUTTON_WDT, BUTTON_
HGT) ) {
                    buttonIdx = i;
                    break;
            }
    }
    if( buttonIdx > -1 ) {
            drawButton(ctxToolbar,    toolbar_btn_posx[buttonIdx],    TOOLBAR_BTN_Y,
toolbar_btn_label[buttonIdx], true);
    }
}

function toolbar_mousemove(event) {
    toolbar_dragged(event.pageX, event.pageY);
}

function toolbar_touchmove(event) {
    if( mousedown ) {
            event.preventDefault();
      var touch = event.changedTouches[0];
      toolbar_dragged(touch.pageX, touch.pageY);
    }
}

function toolbar_dragged(x,y) {

}

function toolbar_mouseup(event) {
    toolbar_released(0,0);
}

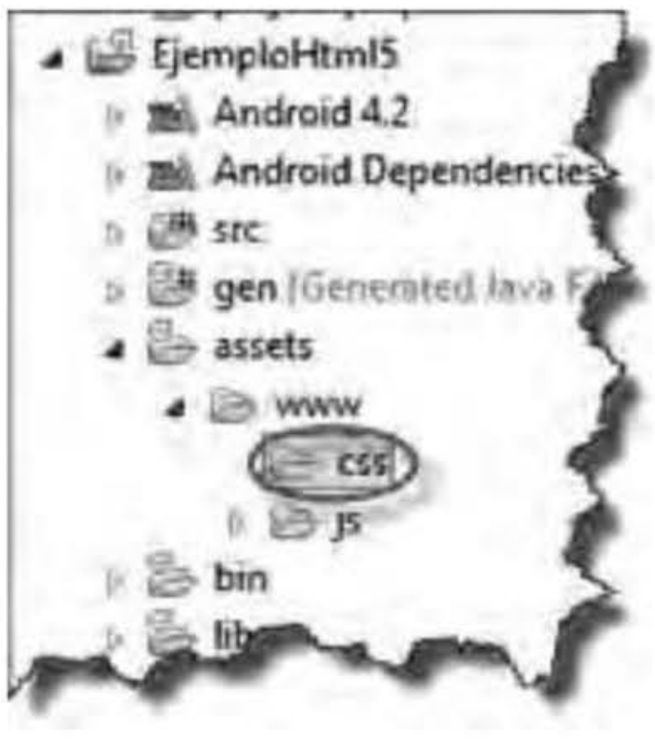
function toolbar_touchend(event) {
    toolbar_released(0,0);
}

function toolbar_released(x,y) {
    mousedown = false;
    if( buttonIdx > -1 ) {
            drawButton(ctxToolbar,    toolbar_btn_posx[buttonIdx],    TOOLBAR_BTN_Y,
```

```
toolbar_btn_label[buttonIdx], false);
            toolbar_action(buttonIdx);
    }
}

function toolbar_action(idx) {
    switch(idx) {
            case TOOLBAR_BTN_NEW:
                    ctxCanvas.fillStyle = 'rgb('+paleta[bgcolorIdx]+')';
                    ctxCanvas.fillRect( 0, 0, ctxCanvas.canvas.width, ctxCanvas.canvas.
height );
                    break;
            case TOOLBAR_BTN_SHARE:
                    var strDataURI = ctxCanvas.canvas.toDataURL('image/jpeg');
                    console.log(strDataURI);
                    break;
            case TOOLBAR_BTN_BRUSHES:
                    lineTmp = lineWidth;
                    document.getElementById('brushes').style.display = 'block';
                    break;
            case TOOLBAR_BTN_COLORS:
                    colorTmp = colorIdx;
                    bgcolorTmp = bgcolorIdx;
                    document.getElementById('palette').style.display = 'block';
                    break;
    }
}
```

Paso 5: Seleccione la hoja de estilo a utilizar. Para ello, cree la siguiente carpeta:



- **STYLES.CSS**

```
body {
    width: 100%;
    height: 100%;
    margin: 0px;
    background-color: #000000;
}

/*#toolbar {
    background-color: #404040;
}

#canvas {
    background-color: #FFFFFF;
}*/

#palette, #brushes {
    background-color: #404040;
    position: absolute;
    left: 0px;
    top: 0px;
    display: none;
}

table, tr, td {
    margin: 0px;
    padding: 0px;
    border: 0px;
    border-spacing: 0px;
}

canvas {
    margin: 0px;
    padding: 0px;
    border: 0px;
    border-spacing: 0px;
    display: block;
    /*opacity: 0.5;
    /*-moz-opacity: 0.5;
    filter:alpha(opacity=5);*/
}
```

La pagina principal index.html

```
<!DOCTYPE html>
<html lang="es">
    <head>
        <meta charset="UTF-8" />
        <title>Pintura</title>
        <link href="css/styles.css" rel="stylesheet" type="text/css" />
        <script type="application/javascript" src="js/main.js"></script>
        <script type="application/javascript" src="js/toolbar.js"></script>
        <script type="application/javascript" src="js/canvas.js"></script>
        <script type="application/javascript" src="js/palette.js"></script>
        <script type="application/javascript" src="js/brushes.js"></script>
    </head>
    <body>
```

```
            <div>
                <canvas id="toolbar" />
            </div>
            <div>
                <canvas id="canvas" />
            </div>
            <div>
                <canvas id="palette" />
            </div>
            <div>
                <canvas id="brushes" />
            </div>
        </body>
    </html>
```

EJECUTANDO LA APLICACIÓN

Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha.

- *Pantalla principal*



- *Manejo de la paleta*



- *Manejo de Colores*





Google Maps Android API V3

17.1. INTRODUCCIÓN

Google Maps fue desarrollado originalmente por los hermanos daneses, Lars y Jens Rasmussen, cofundadores de Where 2 Technologies, una empresa dedicada a la creación de soluciones de mapeo. La empresa fue adquirida por Google en octubre de 2004 y los dos hermanos posteriormente crearon Google Maps (también son los que están detrás de Google Wave).

Antes de que hubiera una API pública, algunos desarrolladores descubrieron la manera de hackear Google Maps para incorporar los mapas en sus propios sitios web. Esto llevó a Google a la conclusión de que había una necesidad de una API pública y en junio de 2005 fue lanzado públicamente Google Maps.

El mashup por primera vez en Internet es, a menudo, considerado que lo ejerció Housingmaps.com, una combinación de Google Maps con los listados de bienes raíces de Craiglist.org. Fue creado de hecho antes de la API pública fuera puesto en libertad y fue hackeado por el desarrollador Paul Rademacher. En mayo de 2010, se anunció la versión 3 del API. Ahora es la opción recomendada para las nuevas aplicaciones de Google Maps y el siguiente paso en la historia de Google Maps.

«Versión 3 del API de JavaScript de Google Maps

Cómo declarar tu aplicación como HTML5

Te recomendamos que declares un DOCTYPE verdadero en tu aplicación web. En los ejemplos que te mostramos aquí, hemos declarado nuestras aplicaciones como HTML5 mediante el sencillo DOCTYPE HTML5, tal y como se muestra a continuación:

```
<!DOCTYPE html>
```

Los navegadores más habituales mostrarán contenido declarado con este DOCTYPE en "modo estándar", lo que significa que tu aplicación deberá ser más compatible con los navegadores. El DOCTYPE también está diseñado para ajustarse de la mejor manera; así, los navegadores que no lo entiendan lo ignorarán y utilizarán el "modo chapucero" para mostrar el contenido.

Ten en cuenta que algunas de las CSS que funcionan en el modo chapucero no son válidas en el modo estándar. En concreto, todos los tamaños basados en porcentajes deben heredarse de los elementos de bloque principales, y si cualquiera de estos antecesores no puede especificar un tamaño, se supondrá un tamaño de 0 x 0 píxeles. Por esta razón, incluimos la siguiente declaración <style>:

```
<style type="text/css">
 html { height: 100% }
 body { height: 100%; margin: 0; padding: 0 }
 #map_canvas { height: 100% }
</style>
```

Esta declaración de CSS indica que el contenedor del mapa (denominado map_canvas) debe ocupar el 100% de la altura del cuerpo de HTML. Ten en cuenta que debemos declarar de forma específica estos porcentajes tanto para <body> como para <html>.

Cómo cargar el API de Google Maps

```
<html>
 <head>
  <script type="text/javascript"
        src="http://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&sensor=SET_TO_
TRUE_OR_FALSE">
  </script>
```

La URL incluida en la etiqueta script indica la ubicación de un archivo JavaScript que carga todos los símbolos y las definiciones que necesitas para utilizar el API de Google Maps. La etiqueta script es obligatoria.

El parámetro key incluye la clave de API de tu aplicación. Para obtener más información, consulta la sección Cómo obtener una clave de API. Ten en cuenta que esta clave no es la misma clave que se utiliza en la versión 2 del API y que se debe generar a partir de la consola de las API.

El parámetro sensor de la URL es obligatorio e indica si esta aplicación utiliza un sensor (por ejemplo, un localizador GPS) para determinar la ubicación del usuario. En este ejemplo hemos dejado el parámetro como la variable set_to_true_or_false para hacer hincapié en que se debe definir este valor en true o false de forma explícita.

Los usuarios del API de Google Maps for Business deben consultar la sección Cómo cargar el API de JavaScript de Google Maps de la documentación para empresas para obtener información importante sobre cómo cargar el API de Google Maps en sus aplicaciones.

HTTPS

Si tu aplicación es una aplicación HTTPS, es posible que quieras cargar el API de JavaScript de Google Maps a través de HTTPS:

```
<script  src="https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&sensor=SET_
TO_TRUE_OR_FALSE"
 type="text/javascript"></script>
```

El uso del protocolo HTTPS para cargar el API de Google Maps es necesario en las aplicaciones SSL para evitar que aparezcan advertencias de seguridad en la mayoría de los navegadores, y se recomienda para las aplicaciones que incluyen datos importantes de los usuarios en las solicitudes (por ejemplo, su ubicación).

Bibliotecas

Al cargar el API de JavaScript de Google Maps mediante la URL http://maps.googleapis.com/maps/api/js, tienes la opción de cargar bibliotecas adicionales utilizando el parámetro libraries. Las bibliotecas son módulos de código que proporcionan funciones adicionales al API de JavaScript, pero que no se cargan a menos que las solicites de forma específica. Para obtener más información, consulta la sección Bibliotecas en la versión 3 del API de Google Maps.

Cómo cargar el API de forma asíncrona

Te recomendamos que cargues el código JavaScript del API de Google Maps después de que la página haya terminado de cargarse o en el momento. Para ello, puedes insertar tu propia etiqueta <script> en respuesta a un evento window.onload o a una llamada de función. No obstante, también debes indicar a la secuencia de comandos de inicialización del API de JavaScript de Google Maps que retrase la ejecución del código de tu aplicación hasta que el código del API de JavaScript de Google Maps se haya cargado por completo. Para ello, puedes utilizar el parámetro callback, que utiliza como argumento la función que se debe ejecutar cuando finaliza la carga del API.

A continuación se muestra un fragmento de código que permite que la aplicación cargue el API de Google Maps una vez que se haya cargado por completo la página (mediante window.onload) y que se incluya el API de JavaScript de Google Maps en una etiqueta <script> dentro de la página. También se puede hacer que el API no ejecute la función initialize() hasta que se haya cargado por completo incluyendo callback=initialize en la secuencia de comandos de inicialización del API de Google Maps:

```
function initialize() {
 var mapOptions = {
  zoom: 8,
  center: new google.maps.LatLng(-34.397, 150.644),
  mapTypeId: google.maps.MapTypeId.ROADMAP
 }
 var map = new google.maps.Map(document.getElementById("map_canvas"), mapOptions);
}

function loadScript() {
 var script = document.createElement("script");
 script.type = "text/javascript";
 script.src = "http://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&sensor=TRUE_
OR_FALSE&callback=initialize";
 document.body.appendChild(script);
}

window.onload = loadScript;
```

Elementos DOM de mapas

```
<div id="map_canvas" style="width: 100%; height: 100%"></div>
```

Para que el mapa aparezca en una página web, se debe reservar un lugar para él. Normalmente, lo hacemos mediante la creación de un elemento div con nombre y la obtención de una referencia a este elemento en el modelo de objetos de documento (DOM) del navegador.

En el ejemplo anterior, definimos un objeto denominado "map_canvas" y establecemos su tamaño mediante atributos de estilo. Ten en cuenta que este tamaño se establece en "100%", lo que amplía el mapa hasta que ocupa toda la pantalla del dispositivo móvil. Es posible que sea necesario ajustar estos valores según el desplazamiento y el tamaño de la pantalla del navegador. Ten en cuenta que el mapa siempre adoptará el tamaño del elemento en el que esté contenido, por lo que siempre debes establecer un tamaño para el elemento de forma explícita.

Opciones de mapas

```
var mapOptions = {
 center: new google.maps.LatLng(-34.397, 150.644),
 zoom: 8,
 mapTypeId: google.maps.MapTypeId.ROADMAP
};
```

Antes de inicializar un mapa, debemos crear un objeto Map options que contenga las variables de inicialización correspondientes. Este objeto no se construye, sino que se crea como un objeto literal.

```
var mapOptions = {};
```

Latitudes y longitudes

Como queremos center el mapa en un punto específico, creamos un objeto LatLng para mantener esta ubicación especificando las coordenadas de ubicación en el orden {latitud, longitud}:

```
center = new google.maps.LatLng(-34.397, 150.644)
```

El proceso de convertir una dirección en un punto geográfico se conoce como codificación geográfica. Esta versión del API de Google Maps incluye funciones de codificación geográfica. Para obtener más información, consulta la sección Codificación geográfica del capítulo Servicios de esta guía.

Niveles de zoom

La propiedad zoom especifica la resolución inicial con la que se mostrará un mapa, donde un zoom de 0 se corresponde con un mapa de la Tierra totalmente alejado y los niveles de zoom acercan el mapa con una resolución más elevada.

```
zoom: 8
```

Para ofrecer un mapa de todo el planeta como una única imagen, es necesario un mapa muy grande o un mapa pequeño con una resolución muy baja. Por consiguiente, las imágenes de mapa en Google Maps y el API de Google Maps se dividen en "mosaicos" de mapas y "niveles de zoom". A niveles bajos de zoom, un conjunto pequeño de mosaicos de mapas cubre una superficie amplia; a niveles de zoom más elevados, los mosaicos tienen una resolución mayor y cubren una superficie más pequeña [...].

Tipos de mapas

También debes establecer expresamente un tipo de mapa inicial en este momento.

```
mapTypeId: google.maps.MapTypeId.ROADMAP
```

Se admiten los siguientes tipos de mapas:

- *ROADMAP, que muestra los mosaicos normales en 2D predeterminados de Google Maps.*
- *SATELLITE muestra imágenes de satélite.*
- *HYBRID muestra una mezcla de mosaicos fotográficos y una capa de mosaicos para los elementos del mapa más destacados (carreteras, nombres de ciudades, etc.).*
- *TERRAIN muestra mosaicos de relieve físico para indicar las elevaciones del terreno y las fuentes de agua (montañas, ríos, etc.).*

Para obtener más información sobre los tipos de mapas, consulta la sección Tipos de mapas. Sin embargo, para la mayoría de los casos lo único que necesitas saber es cómo utilizar los tipos básicos descritos anteriormente.

El objeto "Map"

```
var map = new google.maps.Map(document.getElementById("map_canvas"),
  mapOptions);
```

La clase de JavaScript que representa a los mapas es Map. Cada objeto de esta clase define un único mapa en una página. (Puedes crear más de una instancia de esta clase; cada objeto definirá un mapa independiente en la página). Creamos una nueva instancia de esta clase mediante el operador new de JavaScript.

Al crear una nueva instancia de mapa, se especifica un elemento HTML en la página como contenedor para el mapa. Los nodos HTML son elementos secundarios del objeto document de JavaScript. Se obtiene una referencia a este elemento mediante el método document. getElementById().

Este código permite definir una variable (denominada map) y asignar dicha variable a un nuevo objeto Map, además de transmitir opciones definidas en el objeto mapOptions literal. Estas opciones se utilizarán para inicializar las propiedades del mapa. A continuación se muestra la definición de la función Map(), conocida como constructor:

CONSTRUCTOR	DESCRIPCIÓN
Map(mapDiv:Node, opts?:MapOptions)	Crea un mapa nuevo dentro del contenedor HTML en cuestión, que suele ser un elemento DIV, mediante los parámetros (opcional) que se especifiquen.

Cómo cargar el mapa

```
<body onload="initialize()">
```

Mientras se procesa una página HTML, se crea el modelo de objetos de documentos (DOM) y las imágenes y secuencias de comandos externas se reciben y se incorporan al objeto document. Para garantizar que nuestro mapa se añada a la página cuando se cargue por completo, solo ejecutamos la función que crea el objeto Map cuando el elemento <body> de la página HTML ha recibido un evento onload. De este modo, evitamos un comportamiento impredecible y obtenemos más control acerca del modo y del momento en que se dibuja el mapa.

El atributo onload de la etiqueta body es un ejemplo de un controlador de eventos. El API de JavaScript de Google Maps también proporciona varios eventos que se pueden controlar para determinar cambios de estado. Para obtener más información, consulta la sección Eventos de mapa».

Fuente: developers.google.com

¿Cómo funciona Google Maps?

Es sólo (sic) HTML, CSS y JavaScript trabajando junto. Los mapas son solo imágenes que se cargan en el fondo a través de peticiones ejecutadas por la tecnología de AJAX, y se insertan en un en la página HTML. Mientras navegas en el mapa, el API envía información acerca de las nuevas coordenadas y los niveles de "zoom" de el mapa a través de AJAX y esto retorna las imágenes.

El API consiste de archivos JavaScript que contienen las clases, métodos y propiedades que se usan para el comportamiento de los mapas. ¿Cómo se usan?, de eso se trata esta guía. Esta guía tratará acerca de la última versión creada al momento, la versión 3.

Fuente: www.maestrosdelweb.com



Nota.

La versión 3 del API de JavaScript de Google Maps es ahora la versión oficial del API de JavaScript. La versión 2 de API se ha descartado oficialmente, de acuerdo con nuestra política de funciones obsoletas.

17.2. COORDENADAS

Las coordenadas se utilizan para expresar ubicaciones en el mundo. Hay varios sistemas de coordenadas diferentes. Google Maps utiliza el sistema geodésico 84 (WGS 84), que es el mismo sistema de posicionamiento global (GPS) que utiliza. Las coordenadas se expresan mediante la latitud y la longitud. Está relacionado con la idea que se tiene acerca de las ejes Y y los valores de X en una cuadrícula.

Las coordenadas están expresadas usando números decimales separados por coma. La latitud siempre precede la longitud. La latitud es positiva si va después del punto mostrado en el mapa y negativo si va antes. La longitud es positiva si va arriba del punto y negativa si va debajo.
En los mapas físicos, las coordenadas están expresadas en grados, así que la posición de Puerto Rico sería:

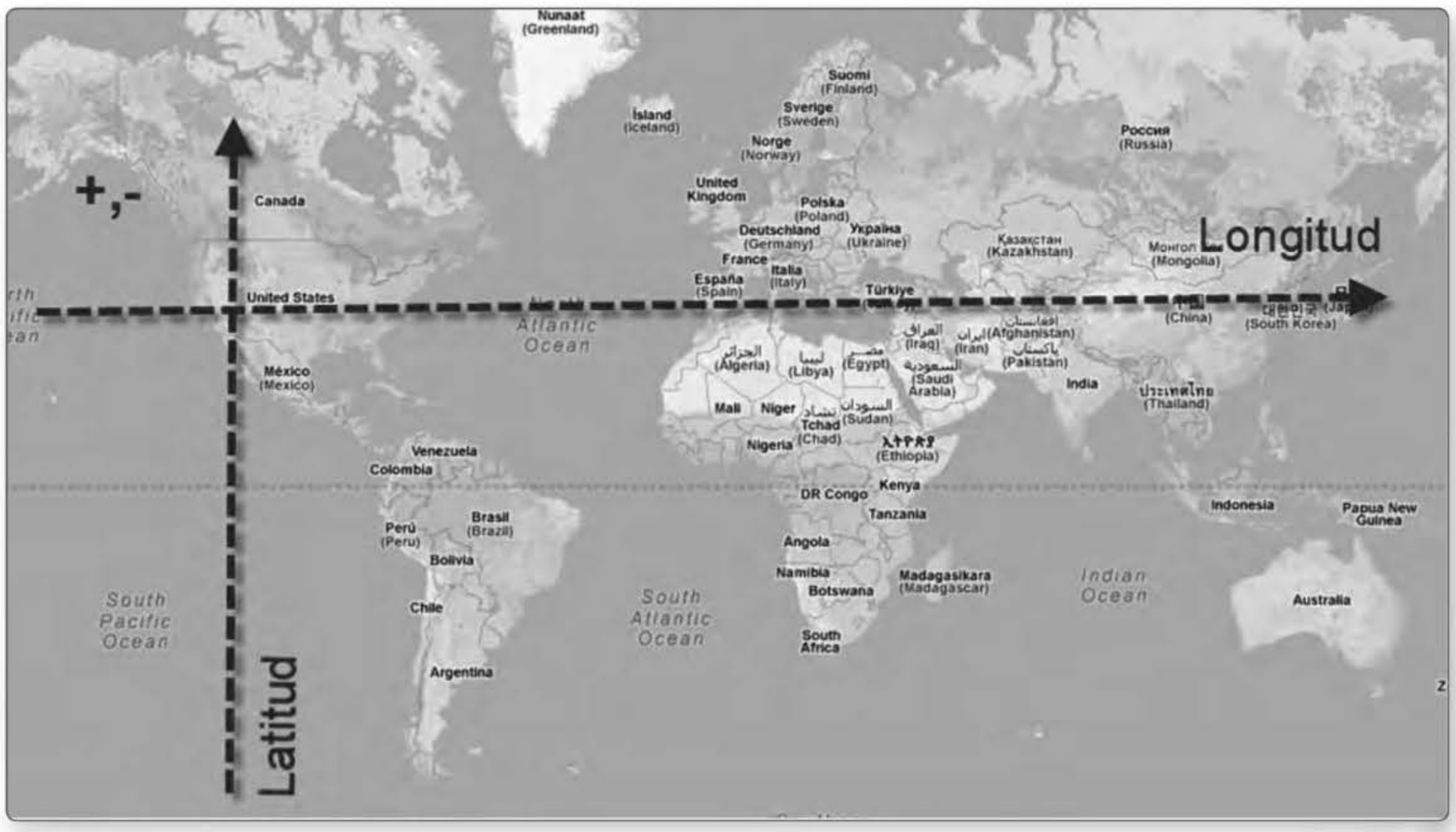
18°14'70" N 66°29'68" W

La forma de convertir estos datos a decimales sería:

(18°14'70" N) = (18 + (14 / 60) + (70 / 3600)) = 18.252
(66°29'68" W) = -(66 + (29 / 60) + (68 / 3600)) = -66.8627

La longitud se multiplica por negativo, porque está a la izquierda (oeste) del punto 0,0. Para esta guía solo vamos a estar trabajando con decimales.

Fuente: www.maestrosdelweb.com



Por ejemplo, el centro del mundo, en la latitud 0 y longitud 0 (centro de la tierra).



¿Cuánto es el máximo de decimales?

Google maps no se limita a cierta cantidad de decimales. Sin embargo, según unas pruebas hechas, se notó que números mayores a 6 decimales es una perdida de tiempo. Así´también google estableció en varios métodos que la mayor cantidad a trabajar es 6 decimales, como por ejemplo el método toUrlValue(). Es decir, cuando vayamos a establecer los decimales se puede hacer así:

- *5 a 6 decimales: es el máximo que debemos usar para ser específicos*
- *4 decimales: para algún detalle en el mapa*
- *3 decimales: es bueno para centrar ciudades*
- *2 decimales: es apropiado para centrar paises o estados, tal vez 3 por monaco*

Fuente: www.maestrosdelweb.com

17.3. Ejemplo Mapa1

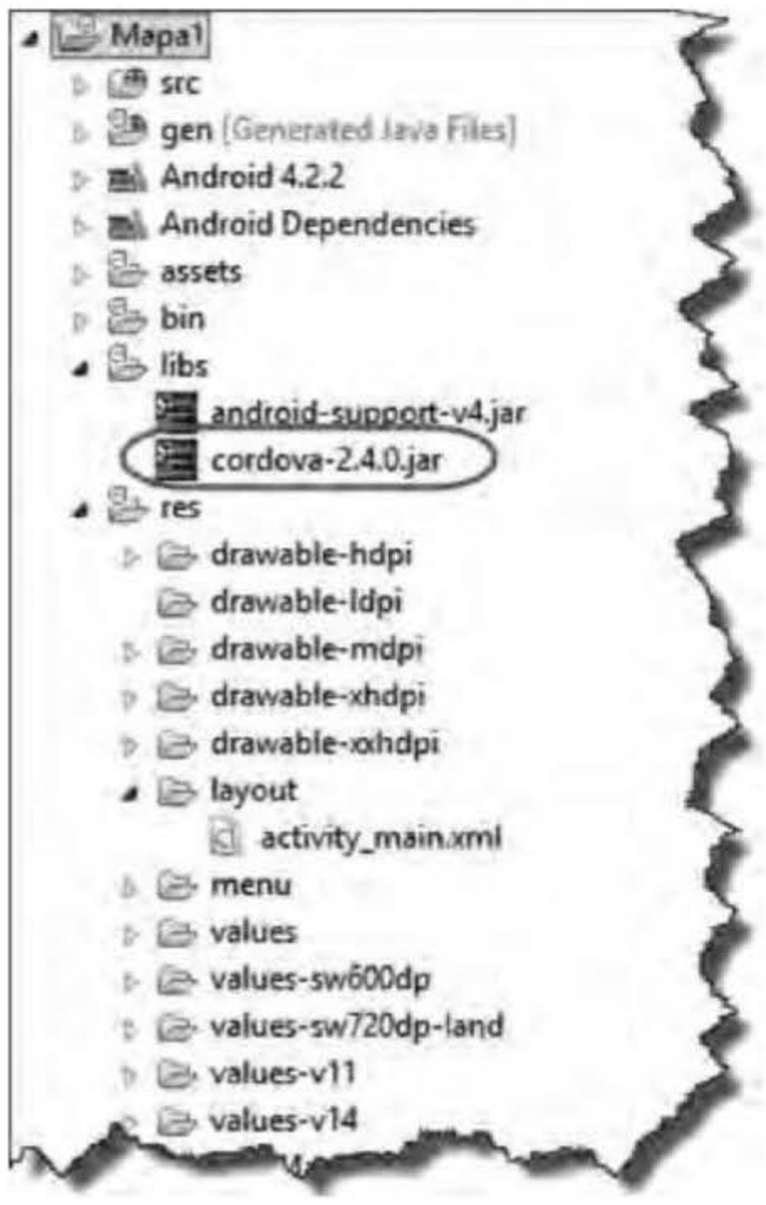


Ahora diseñaremos una aplicación la cual permitira el uso basico de los mapas,

Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto.*

Paso 2: Ingrese el nombre del proyecto: Mapa1 (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

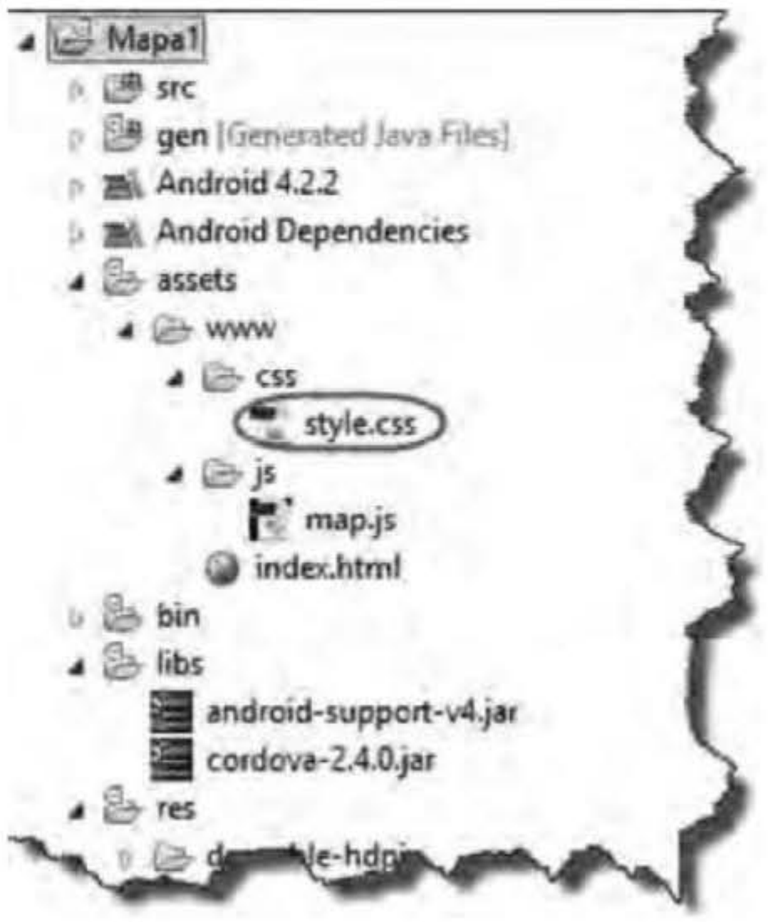
Paso 3: Almacene el fichero .jar en una carpeta y añádala a su proyecto como una librería externa (cordova-2.4.0.jar).



Paso 4: Cree dentro de la carpeta assets, la carpeta www.

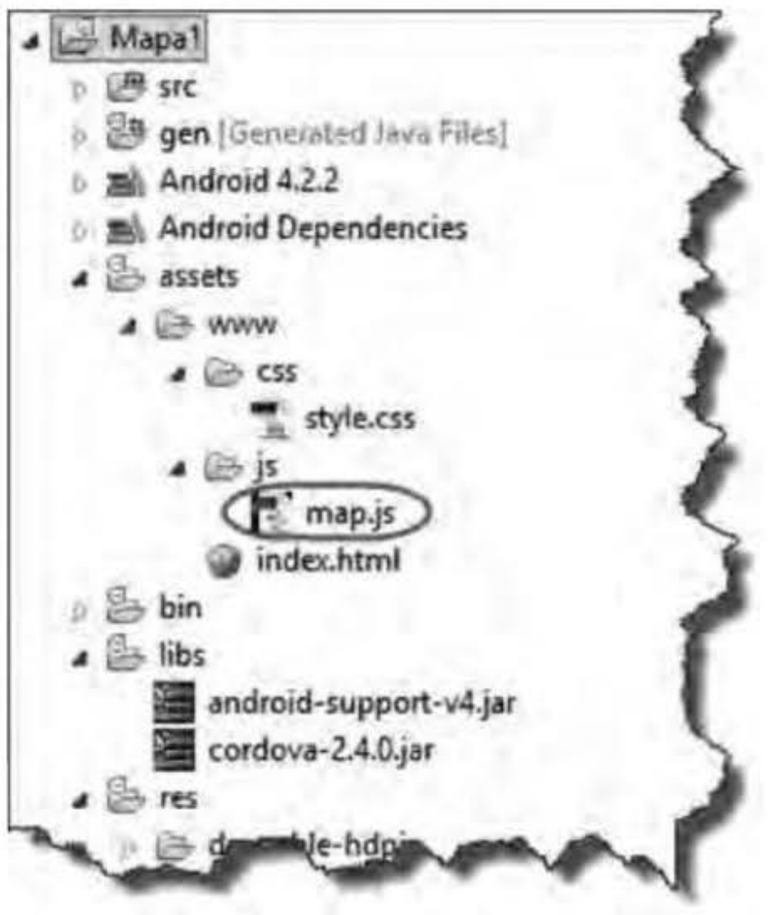


- Hoja de estilo styles.css



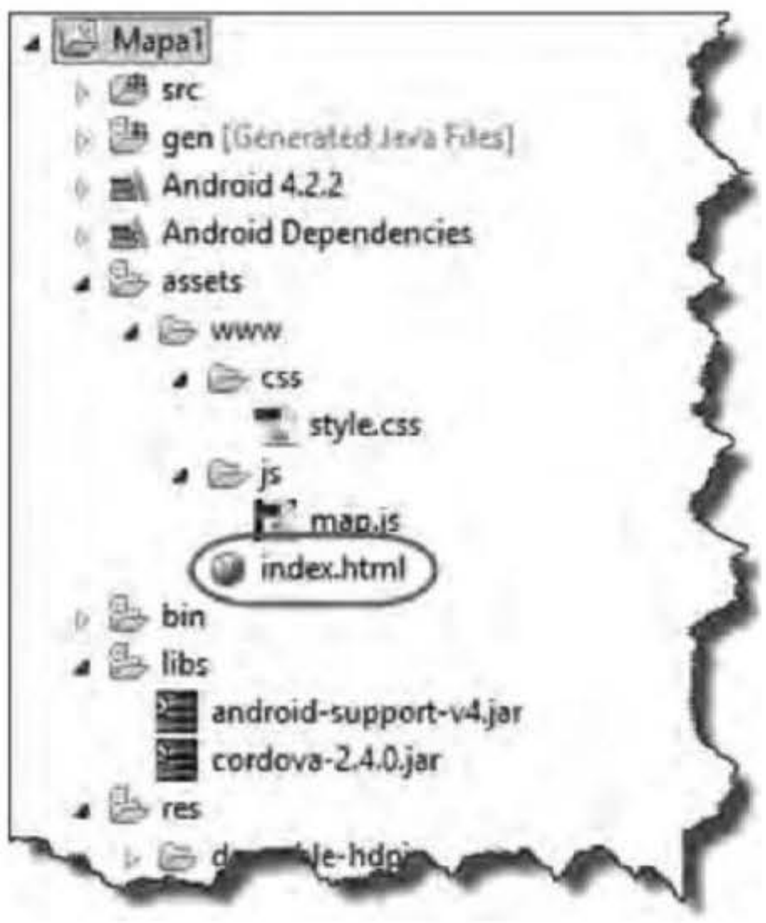

```
body {
 font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
 font-size: small;
 background: #fff;
}
#mapa_peru {
 width: 90%;
 height: 500px;
 border: 3px solid #000;
}
```

- archivo javascript : map.js



```
(function() {
 window.onload = function() {
  alert('Ubicacion del Peru....!');
    // refrencia al div que mostrara el mapa
  var mapDiv = document.getElementById('mapa_peru');
  // referencia de longitud y latitud
  var latlng = new google.maps.LatLng(-12.05,-77.05);
  // propiedades del mapa
  var options = {
   center: latlng,
   zoom: 5,
   mapTypeId: google.maps.MapTypeId.ROADMAP
  };
  // creacion del mapa y lo inserta en la capa
  var map = new google.maps.Map(mapDiv, options);
 }
})();
```

- Archivo html : index.html



- Index.html

```
<html xmlns="http://www.w3.org/1999/xhtml">
 <head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <title>Mi Primer Ejemplo</title>
  <link type="text/css" href="css/style.css" rel="stylesheet" media="all" />
        <script    type="text/javascript"    src="http://maps.google.com/maps/api/
js?sensor=false"></script>
  <script type="text/javascript" src="js/map.js"></script>
 </head>
 <body>
  <h1>Ubicacion del Peru</h1>
  <div id="mapa_peru"></div>
 </body>
</html>
```

Paso 5: Codifique lo siguiente, en la actividad MainActivity.java:

```
import android.os.Bundle;
import org.apache.cordova.*;

public class MainActivity extends DroidGap
{
  public void onCreate(Bundle savedInstanceState)
  {
    super.onCreate(savedInstanceState);
    super.loadUrl("file:///android_asset/www/index.html");
```

Paso 6: Antes de ejecutar la aplicación no se olvide de los permisos necesarios, en el archivo de manifiesto:

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:*



Nota:

Tipos de mapas que se puede utilizar en Google Maps son los siguientes:

- *MapTypeId.ROADMAP muestra la vista de mapa de carretera predeterminada.*
- *MapTypeId.SATELLITE muestra imágenes de satélite de Google Earth.*
- *MapTypeId.HYBRID muestra una mezcla de vistas normales y de satélite.*
- *MapTypeId.TERRAIN muestra un mapa físico basado en información del relieve.*

Paso 2: *Para tener un visualización del mapa tipo satelital, modifique el archivo map.js:*

```
(function() {
  window.onload = function() {
    alert('Ubicacion del Peru....!');
    // referencia al div que contiene el mapa
    var mapDiv = document.getElementById('mapa_peru');

    // referencia de longitud y latitud
    var latlng = new google.maps.LatLng(-12.05,-77.05);
    // propiedades del mapa
    var options = {
      center: latlng,
      zoom: 5,
      mapTypeId:google.maps.MapTypeId.SATELLITE
    };
    // creacion del maps
    var map = new google.maps.Map(mapDiv, options);
  }
})();
```

EJECUTANDO LA APLICACIÓN

Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:



17.4. EJEMPLO MAPA2



Ahora diseñe una aplicación que permita mostrar la ubicación de Perú y permita alternar un mapa de la ciudad de Machu Picchu y la ubicación de Perú; además que muestre los valores siguientes:

Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto.*

Paso 2: Ingrese el nombre del proyecto: Mapa2 (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

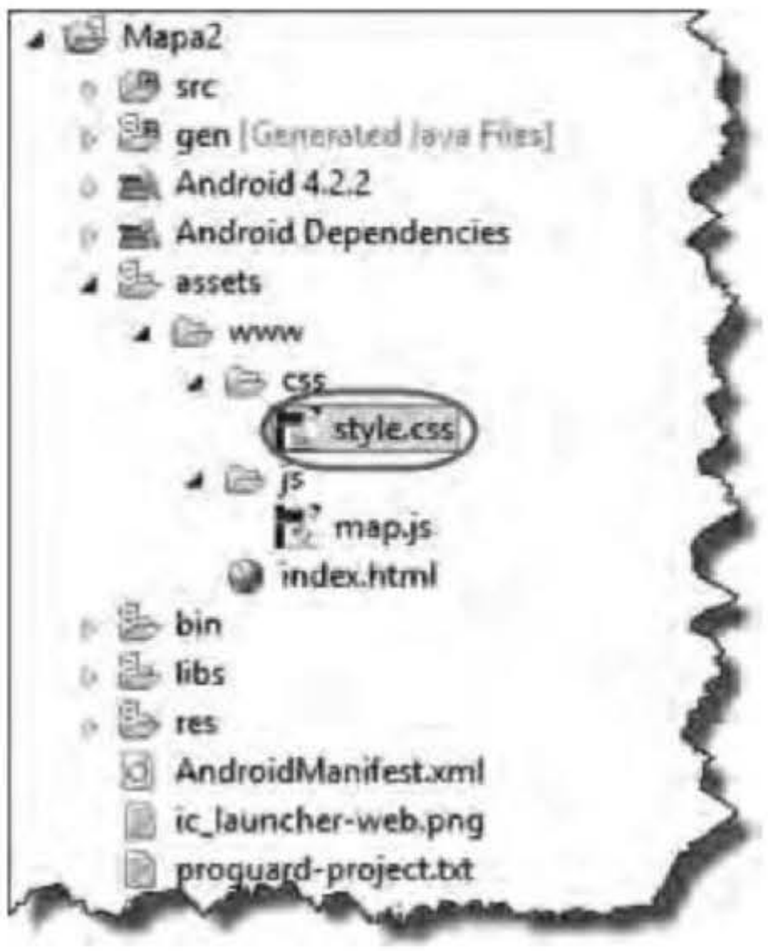
Paso 3: Almacene el fichero .jar en una carpeta y añádala a su proyecto como una librería externa (cordova-2.4.0.jar).



Paso 4: Cree dentro de la carpeta assets, la carpeta www.

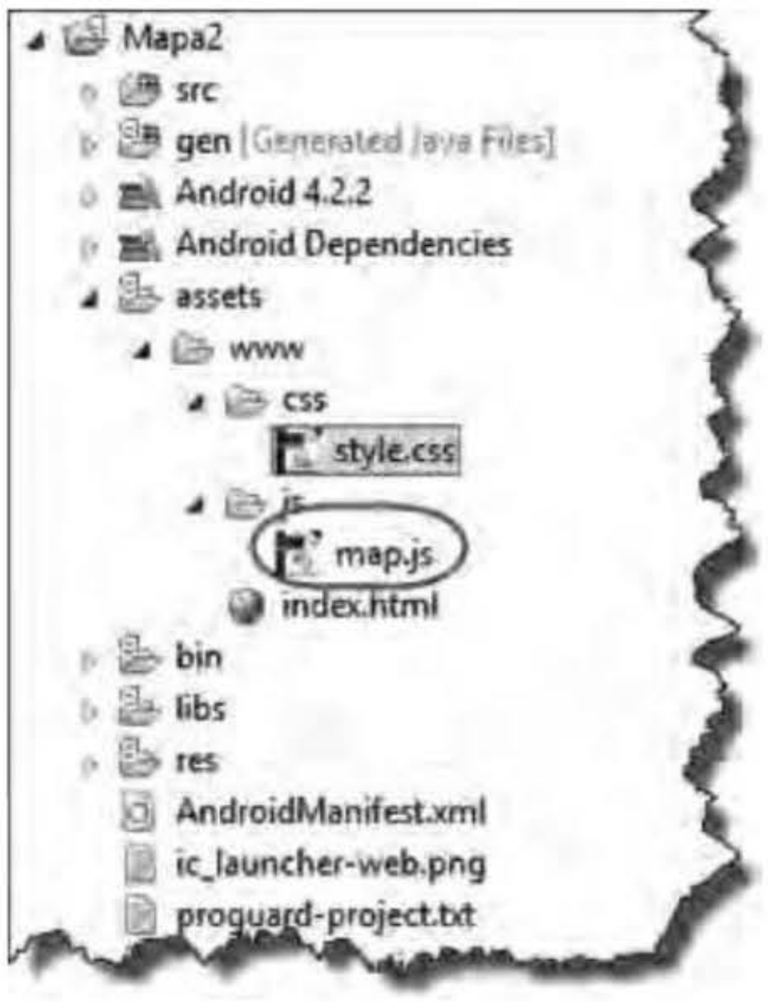


- La hoja de estilo styles.css




```
body {
 font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
 font-size: small;
 background: #fff;
}
#mapa_peru {
 width: 90%;
 height: 500px;
 border: 3px solid #000;
}
```

- El archivo javascript : map.js



```
(function() {
 window.onload = function() {

    var options = {
 zoom: 7,
 center: new google.maps.LatLng(-12.05,-77.051),
 mapTypeId: google.maps.MapTypeId.ROADMAP,
 mapTypeControl: true,
 mapTypeControlOptions: {
  mapTypeIds: [
   google.maps.MapTypeId.ROADMAP,
   google.maps.MapTypeId.SATELLITE
  ]
 }
};


  var map = new google.maps.Map(document.getElementById('mapa_peru'), options);
  document.getElementById('getValues').onclick = function() {
   alert('Nivel ' + map.getZoom());
   alert('Centrado ' + map.getCenter());
   alert('Tipo' + map.getMapTypeId());
  }
  document.getElementById('changeValues2').onclick = function() {
   map.setOptions({
    center: new google.maps.LatLng(-12.05,-77.051),
    zoom: 7,
    mapTypeId: google.maps.MapTypeId.SATELLITE
   });
  }
  document.getElementById('changeValues').onclick = function() {
   map.setOptions({
    center: new google.maps.LatLng(-13.164317,-72.545372),
    zoom: 17,
    mapTypeId: google.maps.MapTypeId.SATELLITE
   });
  }
 };
})();
```

- archivo html : index.html



- Index.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en">
 <head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <title>Segundo Ejemplo</title>
  <link rel="stylesheet" href="css/style.css" type="text/css" media="all" />
        <script    type="text/javascript"    src="http://maps.google.com/maps/api/
js?sensor=false"></script>
  <script type="text/javascript" src="js/map.js"></script>
 </head>
 <body>
  <input type="button" value="Valores" id="getValues" />
  <input type="button" value="Machu Picchu" id="changeValues" />
  <input type="button" value="Peru" id="changeValues2" />
  <div id="mapa_peru"></div>
 </body>
</html>
```

Paso 5: Codifique lo siguiente en la actividad MainActivity.java.

```
import android.os.Bundle;
import org.apache.cordova.*;

public class MainActivity extends DroidGap
{
  public void onCreate(Bundle savedInstanceState)
  {
    super.onCreate(savedInstanceState);
    super.loadUrl("file:///android_asset/www/index.html");
```

Paso 6: Antes de ejecutar la aplicación, no se olvide de los permisos necesarios, en el archivo de manifiesto:

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
```

EJECUTANDO LA APLICACIÓN

Paso 1: *Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:*



- *Vista satélite*



- *Machu Picchu*



17.5. EJEMPLO MAPA3

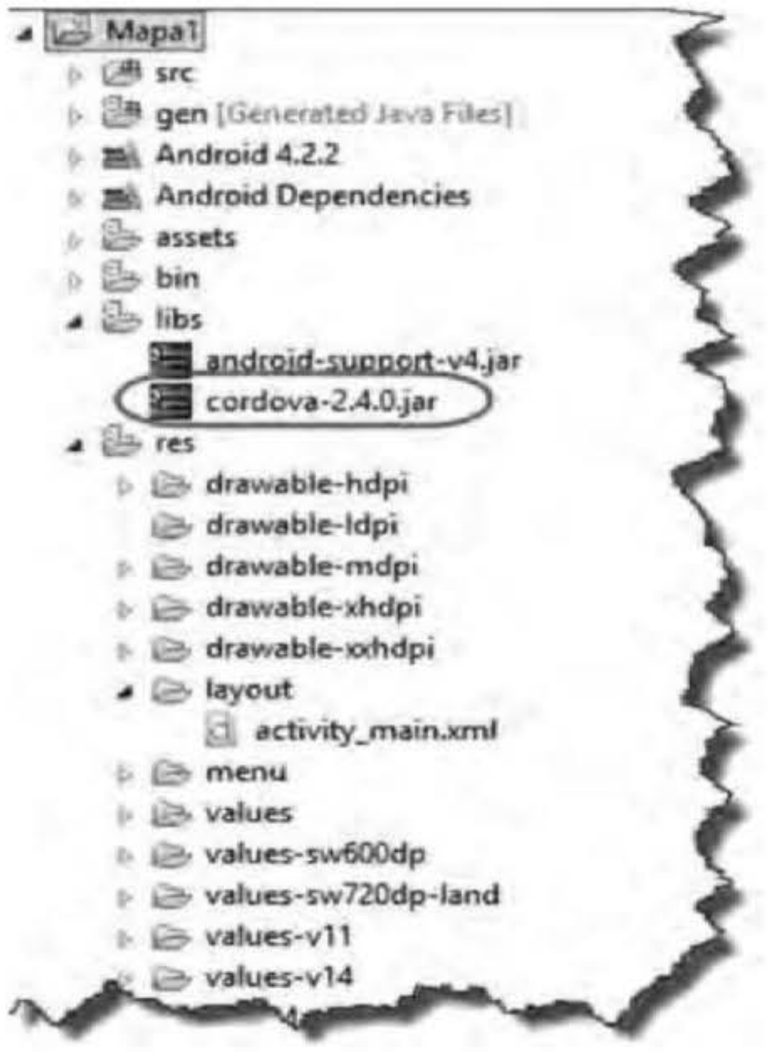


Ahora diseñe una aplicación que permita el uso básico de los mapas y rotación.

Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando Archivo > Nuevo > Android proyecto.

Paso 2: Ingrese el nombre del proyecto: Mapa3 (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

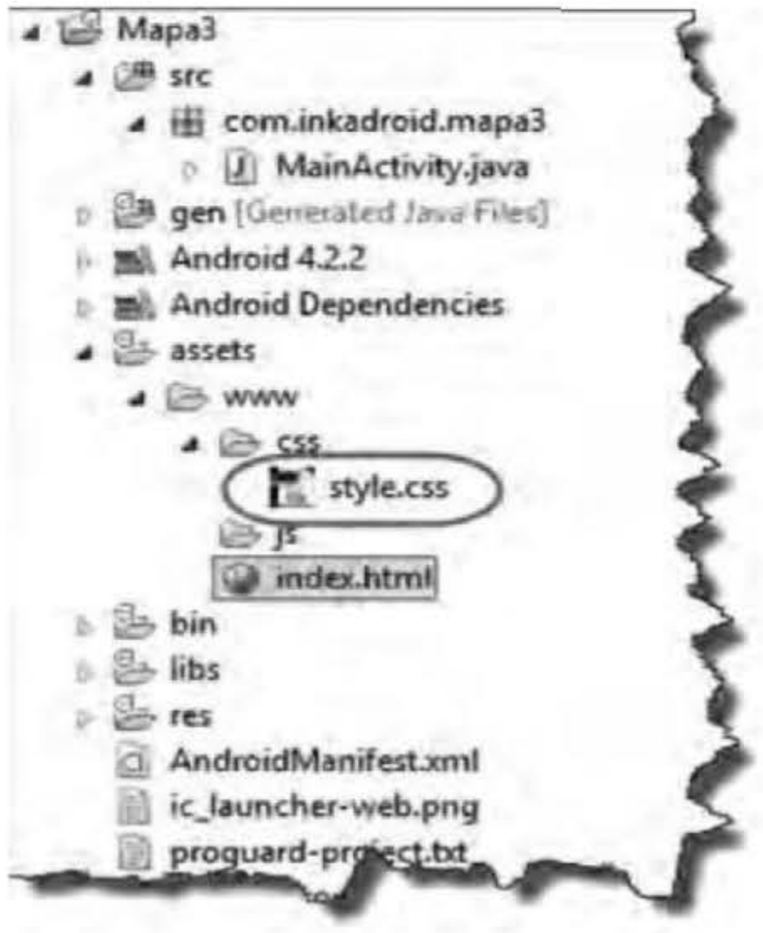
Paso 3: Almacene el fichero .jar en una carpeta y añádala a su proyecto como una librería externa (cordova-2.4.0.jar).





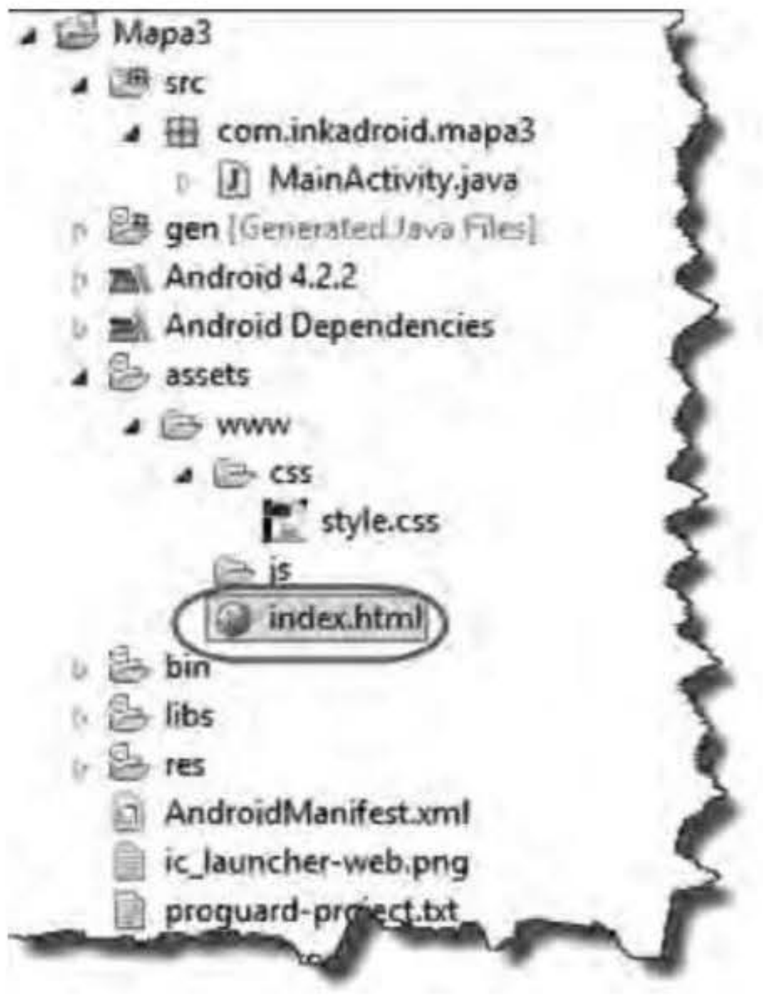
Paso 4: Cree dentro de la carpeta assets, la carpeta www.

- Hoja de estilo styles.css




```
html, body {
 height: 100%;
 margin: 0;
 padding: 0;
}
#mapa{
 height: 100%;
}
@media print {
 html, body {
  height: auto;
 }
 #mapa {
  height: 650px;
 }
}
```

- Archivo html : index.html



- Index.html

```
<!DOCTYPE html>
<html>
 <head>
  <meta name=”viewport” content=”initial-scale=1.0, user-scalable=no”>
  <meta charset=”utf-8”>
  <title>Google Maps JavaScript API v3 Example: Simple Aerials</title>

   <link type=”text/css” href=”css/style.css” rel=”stylesheet” media=”all” />
<script src=”https://maps.googleapis.com/maps/api/js?v=3.exp&sensor=false”></script>
 <script>
  var map;
  function initialize() {
   var mapOptions = {
    center: new google.maps.LatLng(45.518970, -122.672899),
    zoom: 18,
    mapTypeId: google.maps.MapTypeId.SATELLITE,
    heading: 90,
    tilt: 45
   };
   map = new google.maps.Map(document.getElementById(‘mapa’), mapOptions);
  }

  function rotate90() {
   var heading = map.getHeading() || 0;
   map.setHeading(heading + 90);
  }

    function autoRotate() {
     // Determine if we’re showing aerial imagery
     if (map.getTilt() != 0) {
      window.setInterval(rotate90, 3000);
     }
    }

    google.maps.event.addDomListener(window, ‘load’, initialize);
   </script>
  </head>
  <body>
   <input type=”button” value=”Rotar” onclick=”autoRotate();”>
   <div id=”mapa”></div>
  </body>
 </html>
```

Paso 5: Codifique lo siguiente en la actividad MainActivity.java.

```
import android.os.Bundle;
import org.apache.cordova.*;

public class MainActivity extends DroidGap
{
  public void onCreate(Bundle savedInstanceState)
  {
    super.onCreate(savedInstanceState);
    super.loadUrl(“file:///android_asset/www/index.html”);
```

Paso 6: Antes de ejecutar la aplicación, no se olvide de los permisos necesarios en el archivo de manifiesto:

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
```

EJECUTANDO LA APLICACIÓN

Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:



17.6. MARCADORES

Los marcadores identifican ubicaciones en el mapa. De manera predeterminada, utilizan un ícono estándar, aunque puede establecer un ícono personalizado dentro del constructor del marcador o mediante la ejecución de setIcon() en el marcador. El constructor google.maps.Marker toma un único objeto literal Marker options que especifica las propiedades iniciales del marcador. A continuación, se indican algunos campos especialmente importantes que se suelen definir al crear un marcador.

- *position (obligatorio):* Especifica un valor de LatLng que identifica la ubicación inicial del marcador.
- *map (opcional):* Especifica el objeto Map en el que se sitúa el marcador.

Considere que debe especificar el mapa en el que se va a añadir el marcador dentro del constructor Marker. Si no se especifica este argumento, el marcador se creará pero no se añadirá al mapa (o no se mostrará). Puede añadir el marcador más tarde, mediante la ejecución del método setMap() del marcador. Para eliminar un marcador, ejecute el método setMap() y transmita null como el argumento. Los marcadores están diseñados para ser interactivos. De forma predeterminada, reciben eventos click, por ejemplo, y se suelen utilizar dentro de detectores de eventos para abrir ventanas de información. Puede establecer la propiedad draggable de un marcador en true para que los usuarios puedan editar el marcador en el mapa.

17.7. EJEMPLO MAPA4



Ahora diseñe una aplicación que permita el uso básico de marcadores.

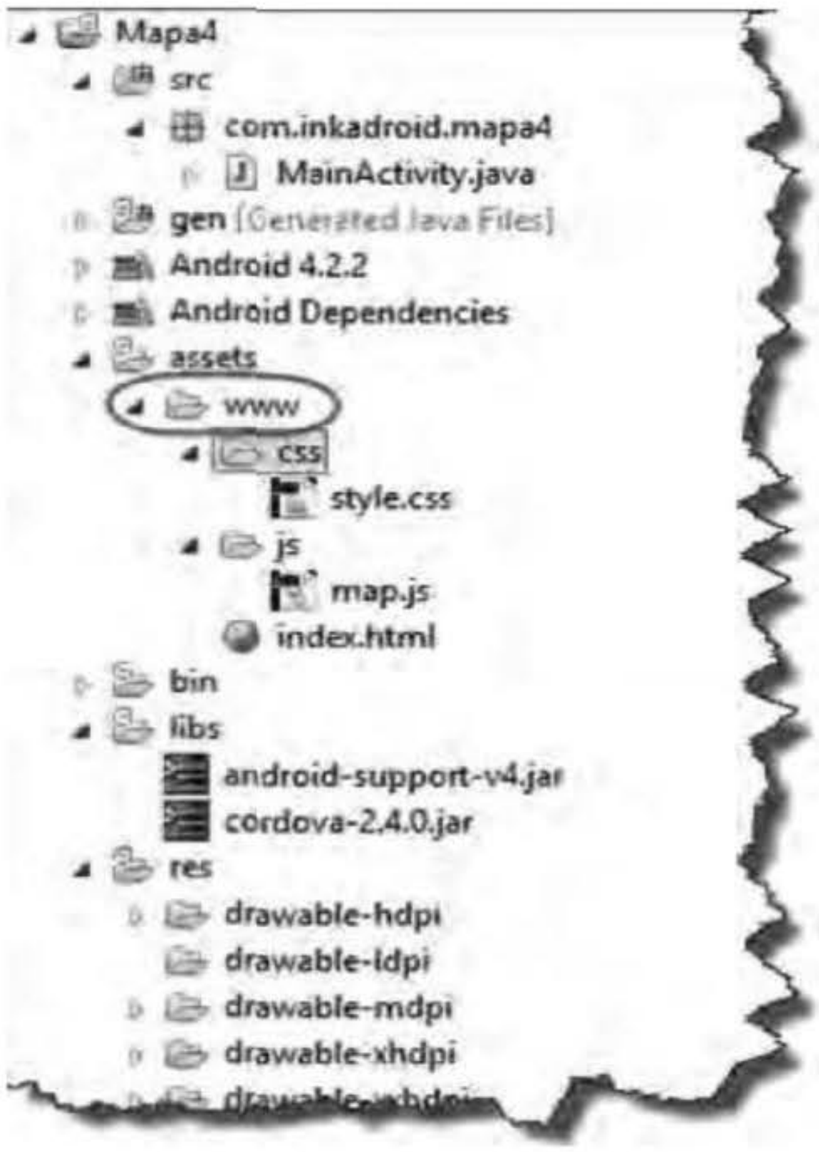
Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto.*

Paso 2: Ingrese el nombre del proyecto: Mapa4 (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

Paso 3: Almacene el fichero .jar en una carpeta y añádala a su proyecto como una librería externa (cordova-2.4.0.jar).



Paso 4: Cree dentro de la carpeta assets, la carpeta www.



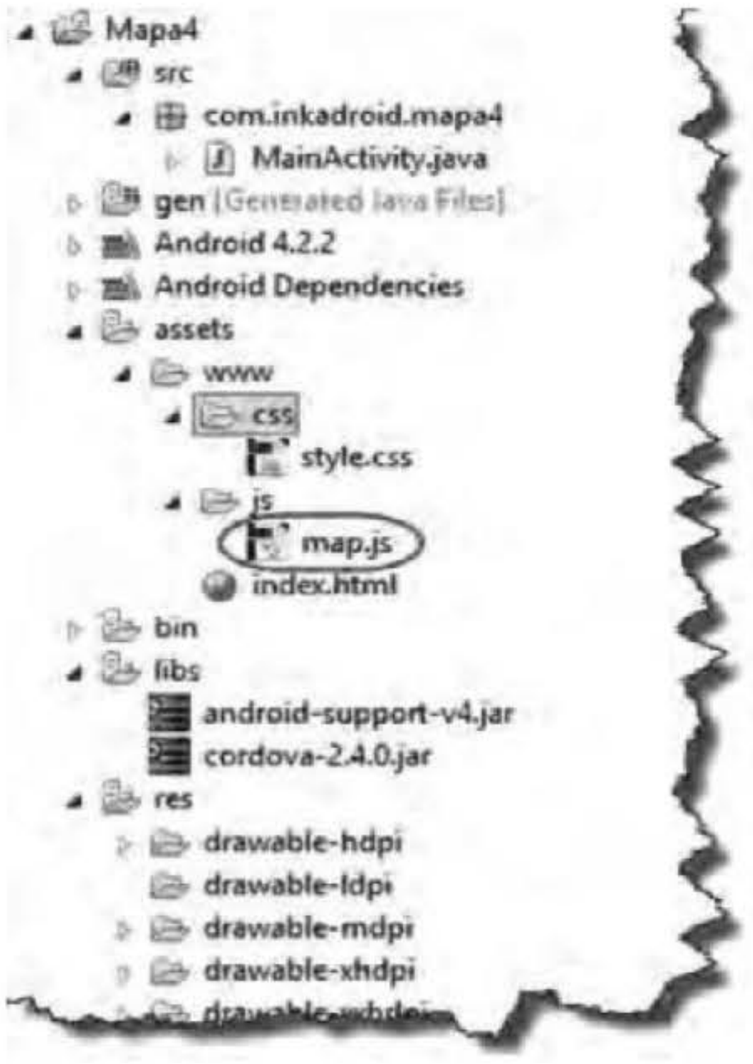
- Hoja de estilo styles.css



- Style.css

```
body {
 font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
 font-size: small;
 background: #fff;
}
#mapa_peru {
 width: 90%;
 height: 500px;
 border: 3px solid #000;
}
```

- Archivo html : map.js



- map.js

```
(function() {
window.onload = function() {
 alert('Ubicacion del Peru....!');
   // refrencia al div que mostrara el mapa
 var mapDiv = document.getElementById('mapa_peru');

 // referencia de longitud y latitud
 var latlng = new google.maps.LatLng(-12.05,-77.05);
 // propiedades del mapa
 var options = {
  center: latlng,
  zoom: 4,
  mapTypeId:google.maps.MapTypeId.ROADMAP
 };
 // creacion del mapa
 var map = new google.maps.Map(mapDiv, options);

       var marker = new google.maps.Marker({
     position: latlng,
     map: map,
       title: 'Hola Peru!'
     });

     }
})
();
```

- archivo html : index.html



- Index.html

```
<html xmlns=”http://www.w3.org/1999/xhtml”>
 <head>
  <meta http-equiv=”Content-Type” content=”text/html; charset=UTF-8” />
  <title>My first map</title>
  <link type=”text/css” href=”css/style.css” rel=”stylesheet” media=”all” />
          <script     type=”text/javascript”     src=”http://maps.google.com/maps/api/
js?sensor=false”></script>
  <script type=”text/javascript” src=”js/map.js”></script>
 </head>
 <body>
  <h1>Ubicacion del Peru</h1>
  <div id=”mapa_peru”></div>
 </body>
</html>
```

Paso 5: Codifique los siguientes en la actividad MainActivity.java.

```
import android.os.Bundle;
import org.apache.cordova.*;

public class MainActivity extends DroidGap
{
  public void onCreate(Bundle savedInstanceState)
  {
    super.onCreate(savedInstanceState);
    super.loadUrl(“file:///android_asset/www/index.html”);
```

Paso 6: Antes de ejecutar la aplicación, no se olvide de los permisos necesarios en el archivo de manifiesto:

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
```

EJECUTANDO LA APLICACIÓN

Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:



17.7. EJEMPLO MAPA5

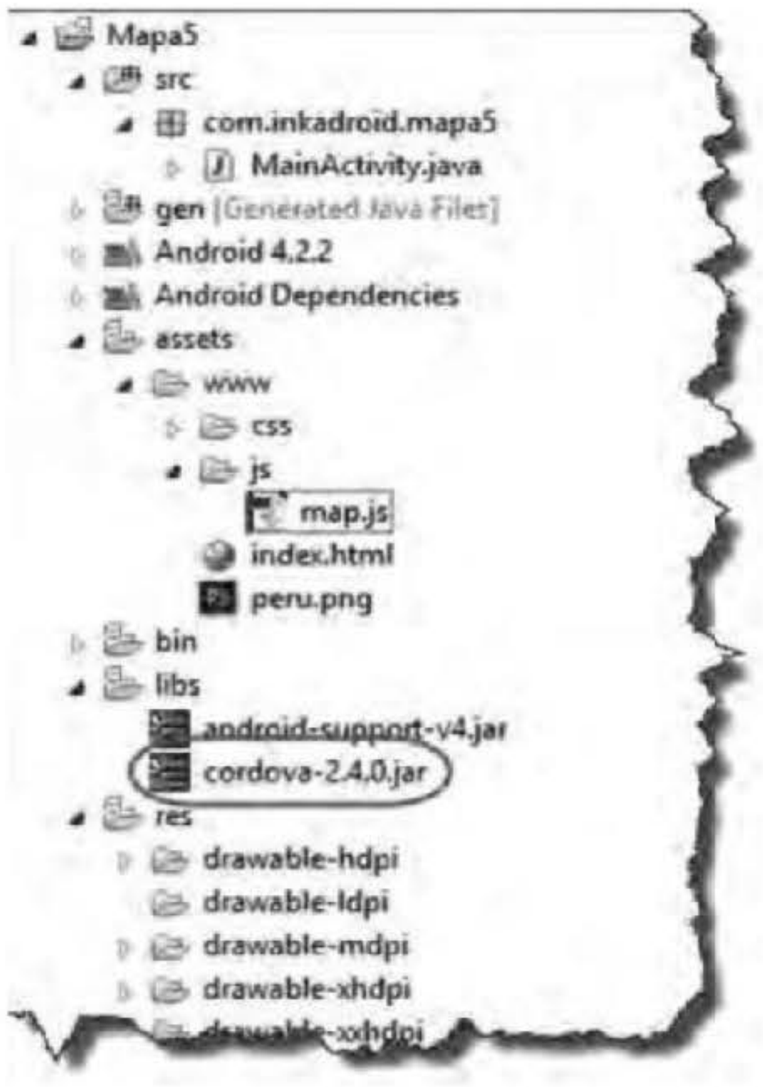


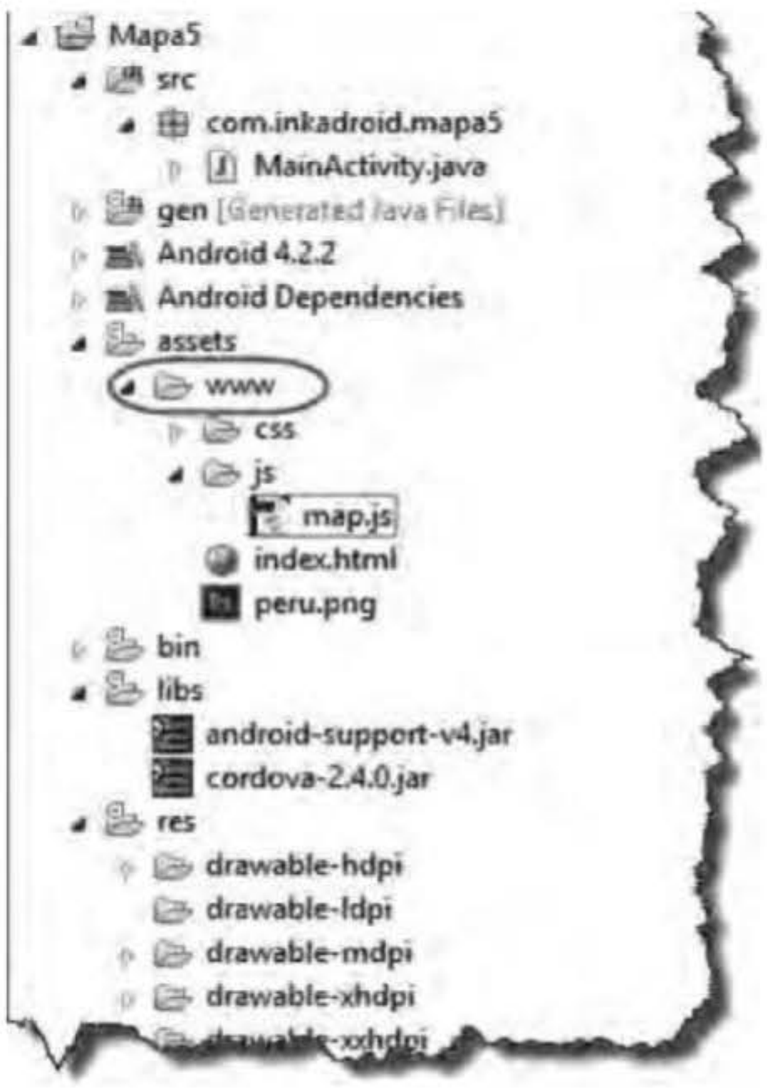
Ahora diseñe una aplicación que permita la personalizacion del marcador.

Paso 1: Después de haber creado la primera aplicación para Android, cree un nuevo proyecto seleccionando *Archivo > Nuevo > Android proyecto.*

Paso 2: Ingrese el nombre del proyecto: Mapa5 (los pasos de creación de un proyecto ya se detallaron en ejemplos anteriores).

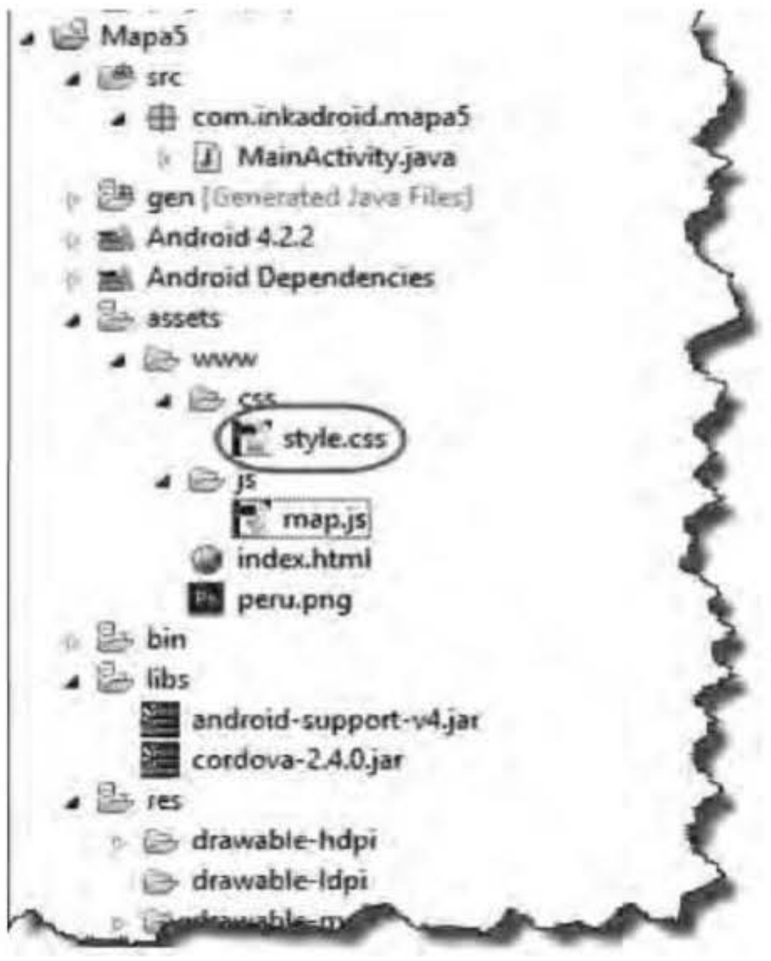
Paso 3: Almacene el fichero .jar en una carpeta y añádala a su proyecto como una librería externa (cordova-2.4.0.jar).





Paso 4: Cree dentro de la carpeta assets, la carpeta www.

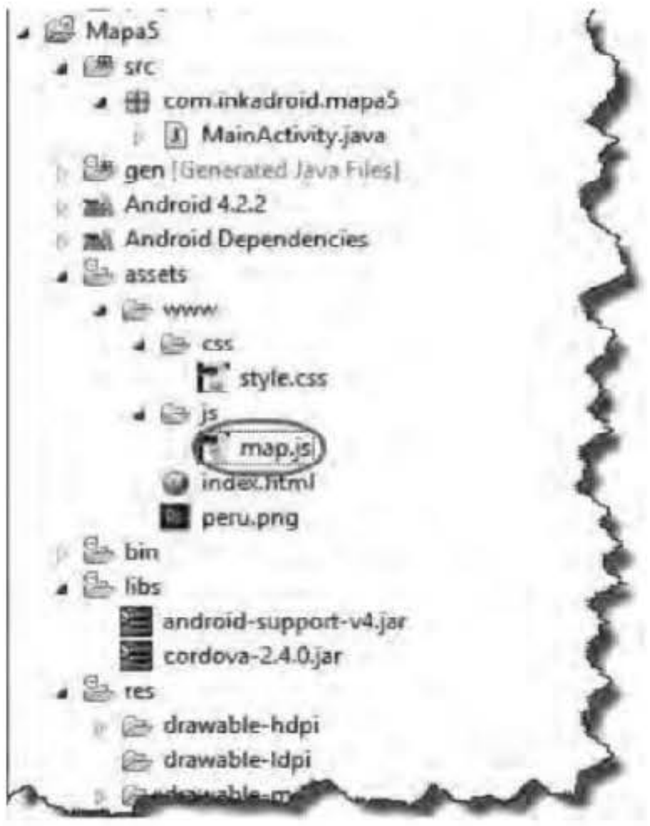
- Hoja de estilo styles.css



- Style.css

```
body {
 font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
 font-size: small;
 background: #fff;
}
#mapa_peru {
 width: 90%;
 height: 500px;
 border: 3px solid #000;
}
```

- Archivo html: map.js



- map.js

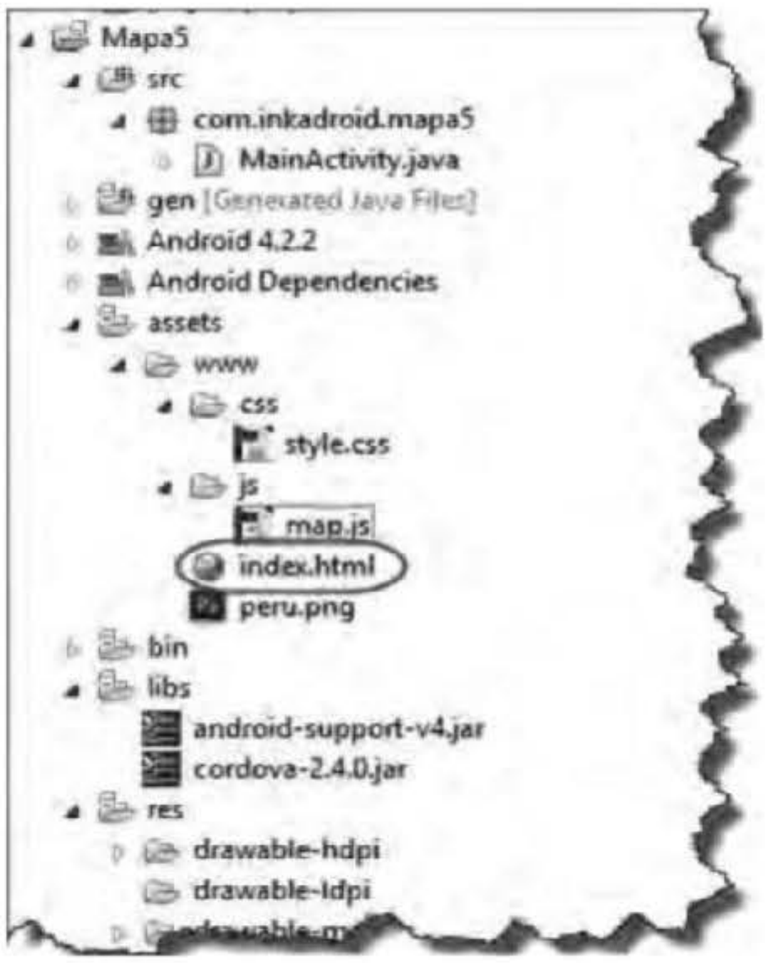
```
(function() {
 window.onload = function() {
  alert('Ubicacion del Peru....!');
    // refrencia al div que mostrara el mapa
  var mapDiv = document.getElementById('mapa_peru');

  // referencia de longitud y latitud
  var latlng = new google.maps.LatLng(-12.05,-77.05);
  // propiedades del mapa
  var options = {
   center: latlng,
   zoom: 4,
   mapTypeId:google.maps.MapTypeId.ROADMAP
  };
  // creacion del mapa
  var map = new google.maps.Map(mapDiv, options);

        var marker = new google.maps.Marker({
      position: latlng,
      map: map,
      title: 'Hola Peru!'
    });

   }
})
();
```

- *archivo html : index.html*



- Index.html

```
<html xmlns="http://www.w3.org/1999/xhtml">
 <head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <title>My first map</title>
  <link type="text/css" href="css/style.css" rel="stylesheet" media="all" />
        <script    type="text/javascript"    src="http://maps.google.com/maps/api/
js?sensor=false"></script>
  <script type="text/javascript" src="js/map.js"></script>
 </head>
 <body>
  <h1>Ubicacion del Peru</h1>
  <div id="mapa_peru"></div>
 </body>
</html>
```

Paso 4: Codifique lo siguiente en la actividad MainActivity.java.

```
import android.os.Bundle;
import org.apache.cordova.*;

public class MainActivity extends DroidGap
{
  public void onCreate(Bundle savedInstanceState)
  {
    super.onCreate(savedInstanceState);
    super.loadUrl("file:///android_asset/www/index.html");
```

Paso 5: Antes de ejecutar la aplicación, no se olvide de los permisos necesarios, en el archivo de manifiesto:

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
```

EJECUTANDO LA APLICACIÓN

Seleccione el nombre del proyecto en Eclipse. Haga clic derecho, seleccione la opción Run As > Android Application y observe que el emulador se pondrá en marcha:



CAP.

18

Códigos ocultos en su dispositivo móvil

18.1. CÓDIGO



Anteriormente, se ha revisado cómo obtener información útil del dispositivo Android, mediante MOBILedit! En este capítulo, se abordará algunos códigos secretos con los que realizar determinadas funciones. En esta ocasión, los códigos sirven para reiniciar o formatear el móvil como venía de fábrica. Cabe advertir que el uso de estos códigos están bajo tu entera responsabilidad.

18.1.1. Lista de los diferentes códigos

```
*#*#8255#*#*
```

Este código sirve para lanzar el monitor de servicio de GTalk.

```
*#*#4636#*#*
```

La información que debe tener su batería es la siguiente:

- Información del teléfono
- Información de la batería
- Historial de la batería
- Estadísticas de uso

```
*#*#7780#*#*
```

Este código sirve para hacer un reset de fábrica por lo que borrará lo siguiente:

- Ajustes de su cuenta Google en el teléfono.
- Ajustes del sistema y de las aplicaciones.
- Aplicaciones descargadas en la SD (descargadas, no instaladas).

Por otro lado, no borrará lo siguiente:

- El software y las aplicaciones que esté instalada.
- Los archivos de la SD.

```
*2767*3855#
```

Cabe advertir que este código hará un reset de fábrica completo (wipe total). Borrará todos los archivos de la memoria interna y reinstalará el firmware.

```
*#*#34971539#*#*
```

Información de su cámara:

- Actualizar el firmware de la cámara (no seleccione está opción).
- Actualizar el firmware de la cámara en la SD.
- Sobre la versión del firmware de la cámara.

```
*#*#7594#*#*
```

Con este código, se apagará directamente su teléfono, sin pasar por el menú.

```
*#*#273283*255*663282*#*#*
```

Este código abre una pantalla que le permite copiar sus archivos multimedia.

```
*#*#197328640#*#*
```

Sirve para acceder al modo Servicio, que sirve para realizar diferentes test. Asimismo, permite utilizar los siguientes códigos:

```
*#*#232339#*#* O bien *#*#526#*#* OR *#*#528#*#* - WLAN test
*#*#232338#*#* - Enseña la dirección MAC actual
*#*#1472365#*#* - GPS test
*#*#1575#*#* - Otro GPS test
*#*#232331#*#* - Bluetooth test
*#*#232337#*# - Muestra la dirección del terminal Bluetooth
*#*#8255#*#* Abres el monitor de Gtalk.
```

18.1.2. Códigos para obtener información del firmware

```
*#*#4986*2650468#*#* - PDA, Phone, H/W, RFCallDate
*#*#1234#*#* - PDA and Phone
*#*#1111#*#* - FTA SW Version

*#*#2222#*#* - FTA HW Version
*#*#44336#*#* - PDA, Phone, CSC, Build Time, Changelist number
```

18.1.3. Códigos para ejecutar diferentes tests de fábrica

```
*#*#0283#*#* - Packet Loopback
*#*#0*#*#* - LCD test
*#*#0673#*#* OR *#*#0289#*#* - Melody test
*#*#0842#*#* - Device test (Vibration test and BackLight test)
*#*#2663#*#* - Touch screen version
*#*#2664#*#* - Touch screen test
*#*#0588#*#* - Proximity sensor test
*#*#3264#*#* - RAM versión
```

CAP.

19

Las mejores aplicaciones para Android

19.1. WHATSAPP



WhatsApp Messenger es un mensajero multiplataforma disponible en Android y en todos otros teléfonos inteligentes. La aplicación utiliza su conexión 3G o Wi-Fi (cuando está disponible) para recibir mensajes de amigos, colegas y familia. Olvídese de los SMS y envíe mensajes, fotos, notas de audio y videos sin coste alguno.

19.2. FACEBOOK



Manténgase en permanente contacto con sus amigos de forma más rápida.

- Vea qué están haciendo sus amigos.
- Comparta actualizaciones de estado, fotos y videos.
- Reciba notificaciones cuando sus amigos comenten publicaciones o marquen que les gusta.
- Envíe SMS, chatee y mantenga conversaciones en grupol
- Juegue y utilice sus aplicaciones favoritas

Para usar Facebook, debe tener 13 años como mínimo. Las condiciones del servicio las puede encontrar en:

http://m.facebook.com/terms.php

19.3. SPOTBROS



Esta aplicación permite enviar mensajes gratuitos a su grupo de contactos, comunicarse con otros usuarios a su alrededor. Asimismo, presenta las SBApps que consisten en aplicaciones que ofrecen servicios útiles y gratuitos a través de mensajería instantánea.

19.3.1. Características

- *Mensajería instantánea:* Envía y recibe mensajes gratuitos para estar siempre conectados con su grupo de contactos. Permite chatear con amigos y crear grupos privados, sin límite de miembros donde se podrá compartir todo tipo de archivos.

- *Seguridad*: En Spotbros, se preocupa por la privacidad de sus usuarios, por eso cifra sus conversaciones con AES256bits (el mismo sistema de cifrado que utiliza EE.UU. para sus documentos Top Secret) que luego las borra de los servidores en un plazo máximo de 30 días.
- *Spots*: Son grupos de chat públicos geolocalizados que se interesan por un tema en común. Además, permite que el usuario pueda crear sus propios Spots.

- *Shout*: Para una turista, esta aplicación le resulta útil porque puede ubicar lugares como un restaurante, por ejemplo, donde el usuario recibirá mensajes de los restaurantes más cercanos a su ubicación actual.
- *Crea grupos privados sin límite de miembros.*
- *SBoole*: Es una forma rápida y fácil de encontrar lo que le interesa al usuario, en cada momento. Encontrará Spots, SBApps y otros usuarios. Para ello, escriba una palabra o si prefiere busque Spots por ubicación o categoría.
- *SBapps:* Consiste en aplicaciones que ofrecen servicios vía mensajería instantánea. Existen SBapps sobre información de transporte público, resultados de fútbol, cambio de divisas, etc.
- *Newsfeed*: Permite compartir sus momentos y enterarse de las últimas noticias sobre sus amigos y contactos. Actualice su estado con fotos, mapas, videos y audios.

19.4. TWITTER



Presenta las siguientes características:

- *Esencial:* Un flujo organizado de tweets que ofrece el mejor contenido para el usuario.
- *Instantáneo:* Se encuentran todos los medios de comunicación, noticias, eventos e información que necesitas.
- *Personal:* Permite expresarse por medio de tweets y fotos.
- *Búsqueda en tiempo real:* Permite seguir a las personas que le gustaría conocer al usuario y connocer las tendencias que se encuentran en auge.

19.5. OTRAS APLICACIONES

Angry Birds para Móviles y Tablets



«[…] Angry Birds Android es uno de los juegos android más descargados y sobretodo más populares de estos tiempos, en el cual tu objetivo será derribar las casas de los cerditos, para esto deberás lanzar a las aves gruñonas con las que contarás, estas famosas Angry Birds que contarán con un gran sentido del humor, así´deberásT acabar con todos los cerditos de cada escenario para poder avanzar en cada uno de los niveles, pero cuidado, porque Angry Birds se irá poniendo cada vez más complicado […]».

Fuente: imovil.org

Camera ZOOM FX



- *Zoom óptico / digital (hasta 6 aumentos)*
- *Flash (si está disponible)*
- *Cámara frontal (si está disponible)*
- *Personalización de TODOS los botones físicos; p.ej: Botones de volumen para controlar el zoom, trackball para tomar la fotografía.*
- *Previsualización en vivo de algunos efectos*
- *Cámara silenciosa (si el dispositivo lo soporta)*
- *Modifica opciones del dispositivo: enfoque automático, balance de blancos, disparo nocturno, ...*
- *Envía y comparte tus tomas.*
- *Sube las imágenes con 1 click a Facebook, Twitter, Flickr, etc, ... ¡A la vez!*
- *Cuadrículas sobreimpresionadas personalizables*

Fuente: play.google.com

Avast! Mobile Security



DESCRIPCIÓN

Una aplicación antivirus y de seguridad para Android con todas las características y con la mejor valoración (4.7 de 5 estrellas).

Localizador/rastreador de teléfonos y protección contra malware (con bloqueador de USSD).

Altamente recomendado por autoridades en Android: AV-Test, PCAdvisor, Techworld, y ahora por 14 millones de usuarios en todo el mundo.

ANÁLISIS:

AndroidAuthority: "El mejor, aún mejor...nada se le compara"
Android and Me: "La mejor solución antirrobo del mercado."
AndroidPolice: "basicamente, lo que usted desearía tener si su dispositivo se pierde o es robado"
Droid-Life: "no se puede pedir mucho más que esto"
PCAdvisor: "El primer puerto en el que recalar cuando busque un producto de seguridad"

PRINCIPALES CARACTERÍSTICAS

- *Disponible en los siguientes idiomas: Catalán, Checo, Holandés, Inglés, Francés, Alemán, Húngaro, Chino (Simplificado), Chino (Tradicional), Italiano, Japonés, Koreano, Polaco, Portugués, Ruso, Español.*
- *Network Meter: controla los flujos de datos de entrada y salida.*
- *Widget: Le permite controlar la aplicación desde el escritorio de Android.*
- *Interfaz Web: Proporciona control remoto de las opciones de anti-theft (bloqueo de teléfono, seguimiento mediante GPS, eliminación remota de datos, etc.).*
- *SiteCorrect: Corrige automáticamente las URLs mal tecleadas.*
- *Control de escudos: Opciones avanzadas para el análisis de aplicaciones, web y mensajes.*
- *Diseño UI: Excelente legibilidad en teléfonos y tablets Android.*
- *Antivirus: analiza, bajo demanda, las aplicaciones instaladas y el contenido de la tarjeta de memoria y cada nueva aplicación antes del primer uso.*
- *Asesor de Privacidad: Analiza y muestra los derechos de acceso de las aplicaciones instaladas.*
- *Filtro SMS y de llamadas: Filtra los contactos que usted seleccione.*
- *Anti-Theft (componente oculto): le proporciona acceso remoto mediante SMS o Web (bloqueo, localización, sirena, borrado de datos, etc.).*
- *Gestor de Aplicaciones: Listado de aplicaciones y su tamaño (MB), carga de CPU, memoria usada, etc.*
- *Escudo Web: analiza cada URL cargada y le avisa si está infectada con malware. Incluye bloqueo de USSD.*
- *Ahorra de batería: Funciona sólo cuando está ejecutando alguna tarea.*
- *Cortafuegos (sólo funciona en teléfonos rooteados): Sirve como filtro para el tráfico de red.*

Para más información, visite: http://www.avast.com/free-mobile-security

Controle de forma remota su teléfono en: https://my.avast.com

Fuente: play.google.com

Battery Widget? Reborn!



Battery Widget Reborn es una excelente aplicación para controlar el uso de la batería que cuenta una interfaz estilo Holo muy cuidada y muy fácil de usar. Gracias a su widget y a su notificación configurable en la barra de estado podemos estar siempre informados sobre la batería restante.

Fuente: www.xatakandroid.com

Cerberus



«Descripción
Cerberus es un programa completo para el uso contra los robos de sus dispositivos. Es la mejor protección para que puede recuperar sus dispositivos de Android extraviados, perdidos o robados. Esta es una versión de prueba gratuita durante una semana. Luego usted puede comprar una licencia de por vida a un pequeño precio (2.99€) desde la aplicación: no hay cuotas mensuales o anuales, sólo un pago por única vez. La licencia está asociada a su cuenta de Cerberus. Si tiene varios dispositivos (hasta 5), lo puede utilizar todos con la misma cuenta.

Tiene tres maneras de proteger su dispositivo:

- *Via control remoto a través de la página web www.cerberusapp.com*
- *Via control remoto a través de de mensajes de textos SMS*
- *SIM Checker (para los dispositivos que tienen una tarjeta SIM): recibirá automáticamente alertas si alguien utiliza su teléfono con una tarjeta SIM no autorizada*

El control remoto le permite realizar muchas operaciones en el dispositivo, como:

- *Localizar y rastrear su dispositivo*
- *Iniciar una fuerte alarma, incluso si el dispositivo está en modo silencioso*

- *Borra la memoria interna y de la tarjeta SD*
- *Ocultar Cerberus del menu de las aplicaciónes*
- *Bloquear el dispositivo con un código*
- *Grabar audio desde el micrófono*
- *Obtener una lista de los últimas llamadas enviadas y recibidas*
- *Obtener información sobre la red y el operador a cual el dispositivo está conectado*
- *Y mucho mas!*

Cuando intenta localizar su dispositivo, Cerberus habilita automáticamente GPS si está apagado, (sólo en Android <2.3.3) y lo puede proteger de la desinstalación no autorizada - más información en la aplicación

La aplicación funciona incluso si el dispositivo no tiene una conexión a Internet gracias a la función de control remoto por medio de mensajes SMS. Asimismo, la función SIM Checker le permite conocer el nuevo número si se cambia la tarjeta SIM, para que puede enviar textos a este número. Los textos recibidos y enviados para el control remoto de Cerberus, no iniciaran ninguna notificación ni aparecerán en la aplicación de mensajes.

Cerberus no agota la batería, porque no hay servicio en segundo plano en ejecución: se activará sólo si hay una solicitud por control remoto y se detendrá después [...]».

Fuente: play.google.com

Kingosft Office



«Kingsoft Office Reader for Android Free es un excelente visor de documentos de Microsoft Office para teléfonos moviles y tablets con sistema Android.

El programa se compone fundamentalmente de tres partes: el Word reader, el Excel reader y el PowerPoint reader. Gracias cada uno de estos tres componentes podremos abrir en nuestro smartphone o tablet cualquier archivo creado con Office y que tenga diferentes extensiones, tales como doc, .docx, .txt, .ppt, .xls y .xlsx.

Kingsoft Office Reader for Android Free es completamente funcional, permite la visualización de todos los elementos incrustados en los documentos, desde simples graficos hasta las mas sofisticadas imagenes. Además nos permite la visión en pantalla completa y un potente gestor de archivos».

Fuente: es.kioskea.net

Impreso en los Talleres Gráficos de



MACRO®
EMPRESA EDITORA

Surquillo
☎ 719-9700