Arquitectura de Android

Juan David Higuita

Daniel Alexander Agualimpia Córdoba

Iván Taborda

Madai Valentina García Velásquez

Mariana Herrera

Jorge Henao

Trabajo practico realizado para sustentar los conocimientos en Arquitectura del sistema operativo Android

Andrea Carolina Bedoya Gómez

Instructor Sena

SERVIVIO NACIONAL DE APRENDIZAJE

Análisis y desarrollo de sistemas de información

Definir la arquitectura del sistema de información

Medellín

Septiembre

2020

# CONTENIDO

PAGINA

[Introducción](#INTRODUCCION)  4

[Objetivos](#OBJETVOS) 5

[Arquitectura de la plataforma](#_ARQUITECTURA_DE_LA) 6

[Kernel de Linux](#_KERNEL_DE_LINUX) 7

[Runtime de Android](#_RUNTIME_DE_ANDROID)  7

[Librerías Nativas](#_LIBRERIAS_NATIVAS) 8

[Framework de aplicaciones](#_FRAMEWORK_DE_APLICACIONES:) 10

[Aplicaciones](#_APLICACIONES) 11

[Conclusiones](#CONCLUSIONES) 13

[Web grafía](#BIBLIOGRAFIA)  14

# INTRODUCCION

[CONTENIDO](#_CONTENIDO)

Android es un sistema operativo creado para ser independiente de cualquier tipo de arquitectura de hardware en los dispositivos móviles. Esta característica hace que sea tan atractivo ante los fabricantes y desarrolladores.

Adicionalmente su portabilidad, flexibilidad y seguridad les da ese toque de simpatía a las personas interesadas en los sistemas de código abierto.

La arquitectura de Android debe ser estudiada antes de comenzar a programar. Por tal motivo, en este artículo veremos cómo está constituido el interior de Android.

# OBJETIVOS

[CONTENIDO](#_CONTENIDO)

**Objetivo General**

Realizar un documento en el que quede plasmada como está conformada la arquitectura del sistema operativo Android

**Objetivos específicos**

* Tener una guía en la que me pueda apoyar
* Conocer la estructura de Android

# ARQUITECTURA DE LA PLATAFORMA

[CONTENIDO](#_CONTENIDO)

Android es una pila de software de código abierto basado en Linux creada para una variedad amplia de dispositivos y factores de forma. En el siguiente diagrama, se muestran los componentes principales de la plataforma Android.



# KERNEL DE LINUX

[CONTENIDO](#_CONTENIDO)

El núcleo de Android está formado por el sistema operativo Linux versión 2.6. Esta capa proporciona servicios como la seguridad, el manejo de la memoria, el multiproceso, la pila de protocolos y el soporte de drivers para dispositivos.

Esta capa del modelo actúa como capa de abstracción entre el hardware y el resto de la pila. Por lo tanto, es la única que es dependiente del hardware.

Se ha modificado dramáticamente para adaptarse a dispositivos móviles. Esta elección está basada en la excelente potabilidad, flexibilidad y seguridad que Linux presenta. El Kernel de Linux está bajo la licencia GPL, así que en consecuencia Android también.

El uso del kernel de Linux permite que Android aproveche funciones de seguridad claves y, al mismo tiempo, permite a los fabricantes de dispositivos desarrollar controladores de hardware para un kernel conocido.

# RUNTIME DE ANDROID

[CONTENIDO](#_CONTENIDO)

Para los dispositivos con Android 5.0 **(nivel de API 21)** o versiones posteriores, cada app ejecuta sus propios procesos con sus propias instancias del tiempo de ejecución de Android **(ART).** El **ART** está escrito para ejecutar varias máquinas virtuales en dispositivos de memoria baja ejecutando archivos **DEX**, un formato de código de bytes diseñado especialmente para Android y optimizado para ocupar un espacio de memoria mínimo. Crea cadenas de herramientas, como Jack, y compila fuentes de Java en código de bytes **DEX** que se pueden ejecutar en la plataforma Android.

Estas son algunas de las funciones principales del **ART**:

Compilación ahead-of-time **(AOT)** y just-in-time **(JIT)**

Recolección optimizada de elementos no utilizados **(GC)**

En Android 9 **(nivel de API 28)** y versiones posteriores, se convierten los archivos de formato ejecutable **(DEX)** de un paquete de aplicaciones a un código de máquina más compacto

Esto mejora la compatibilidad con la depuración, el generador de perfiles de muestras dedicado, las excepciones de diagnóstico detalladas y los informes de fallos, y la capacidad de establecer puntos de control para supervisar campos específicos

Antes de Android 5.0 **(nivel de API 21),** **Dalvik** era el entorno de ejecución del sistema operativo. Si tu app se ejecuta bien en el **ART**, también debe funcionar en **Dalvik**, pero es posible que no suceda lo contrario.

En Android, también se incluye un conjunto de bibliotecas **(Core Libraries)** de entorno de ejecución centrales que proporcionan la mayor parte de la funcionalidad del lenguaje de programación Java; se incluyen algunas funciones del lenguaje Java 8, que usa el marco de trabajo de la API de Java.

# LIBRERIAS NATIVAS

[CONTENIDO](#_CONTENIDO)

Librerías: La siguiente capa se corresponde con las librerías utilizadas por Android. Éstas 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 el corazón de Android.

Entre las librerías más importantes ubicadas aquí, se pueden encontrar 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 (vídeo, 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:** Creación y gestión de bases de datos relacionales.

· **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.

# FRAMEWORK DE APLICACIONES:

[CONTENIDO](#_CONTENIDO)

Representa fundamentalmente el conjunto de herramientas de desarrollo de cualquier aplicación. Toda aplicación que se desarrolle para Android, ya sean las propias del dispositivo, las desarrolladas por Google o terceras compañías, o incluso las que el propio usuario cree, utilizan el mismo conjunto de API y el mismo "framework", representado por este nivel.

Esta es la capa que nos interesa a los desarrolladores, ya que en ella encontramos todas las librerías Java que necesitamos para programar nuestras aplicaciones. Los paquetes con más preponderancia son los Android. \*, en ellos se alojan todas las características necesarias para construir una aplicación Android.

No obstante, es posible acceder a clases como java.utils.\*, java.net.\* , etc. Aunque hay librerías Java excluidas como la java.awt.\* y java.swing.\*.

En esta capa también encontraremos manejadores, servicios y proveedores de contenido que soportaran la comunicación de nuestra aplicación con el ecosistema de Android.

Entre las API más importantes ubicadas aquí, se pueden encontrar las siguientes:

* **Activity Manager:** 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 Surface Manager.
* **Telephone** **Manager:** Incluye todas las API vinculadas a las funcionalidades propias del teléfono (llamadas, mensajes, etc.).
* **Content Provider:** Permite a cualquier aplicación compartir 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 teclado, etc. Incluye también algunas vistas estándar para las funcionalidades más frecuentes.
* **Location Manager:** Posibilita a las aplicaciones la obtención de información de localización y posicionamiento.
* **Notification Manager:** Mediante el cual las aplicaciones, usando un mismo formato, comunican al usuario eventos que ocurran durante su ejecución: una llamada entrante, un mensaje recibido, conexión Wi-Fi disponible, ubicación en un punto determinado, etc. Si llevan asociada alguna acción, en Android denominada Intent, (por ejemplo, atender una llamada recibida) ésta se activa mediante un simple clic.
* **XMPP Service:** Colección de API para utilizar este protocolo de intercambio de mensajes basado en XML.

# APLICACIONES

[CONTENIDO](#_CONTENIDO)

En Android se incluye un conjunto de apps centrales para correo electrónico, mensajería SMS, calendarios, navegación en Internet y contactos, entre otros elementos. Las apps incluidas en la plataforma no tienen un estado especial entre las apps que el usuario elije instalar; por ello, una app externa se puede convertir en el navegador web, el sistema de mensajería SMS o, incluso, el teclado predeterminado del usuario (existen algunas excepciones, como la app Settings del sistema).

Las apps del sistema funcionan como apps para los usuarios y brindan capacidades claves a las cuales los desarrolladores pueden acceder desde sus propias apps. Por ejemplo, si en tu app se intenta entregar un mensaje SMS, no es necesario que compiles esa funcionalidad tú mismo; como alternativa, puedes invocar la app de SMS que ya está instalada para entregar un mensaje al receptor que especifiques.

**¿Qué Tipo De Archivo Tienen Las Aplicaciones Para Android?**

El resultado del proceso de construcción es un archivo comprimido con formato. APK (Android Applicacion Package). Y dentro encontraremos los siguientes componentes:

Archivo Android Manifest: Este archivo es la definición de todas las características principales que tendrá nuestra aplicación al ejecutarse en un dispositivo móvil. Con características me refiero a los bloques que posee la aplicación, los permisos, su versión, las versiones previas soportadas, las dimensiones de la pantalla, etc.

Archivo classes.dex: Este será el fichero compilado preparado para ejecutarse en la Máquina Virtual Dalvik.

La carpeta Resources: Aquí encontramos todos los archivos externos que usamos para construir nuestro proyecto, como por ejemplo nuestros iconos, audio, archivos planos de texto, los archivos .XML de diseño, etc.

Librerías nativas: El archivo. APK también contiene aquellas librerías de las cuales depende la aplicación.

Carpeta META-INF: En ella se guardan archivos que corresponden a las Firmas Digitales de tu aplicación. Con esta especificación puedes indicar que tú eres el creador y dueño de la aplicación, además debes indicar tu ID de desarrollador para ser reconocido y autenticado en procesos de comercialización (Google es muy riguroso en este tema).

# CONCLUSIONES

[CONTENIDO](#_CONTENIDO)

En este artículo estudiamos la forma en que está construido Android como sistema operativo.

Vimos cómo se construyen las aplicaciones Android a través de un modelo de máquina virtual y como interviene el framework.

# WEBGRAFIA

[CONTENIDO](#_CONTENIDO)

<http://www.hermosaprogramacion.com/2014/08/aprendiendo-la-arquitectura-de-android/>

<https://developer.android.com/guide/platform?hl=es-419>

<http://www.androidcurso.com/index.php/recursos/31-unidad-1-vision-general-y-entorno-de-desarrollo/99-arquitectura-de-android>

<https://sites.google.com/site/swcuc3m/home/android/generalidades/2-2-arquitectura-de-android>