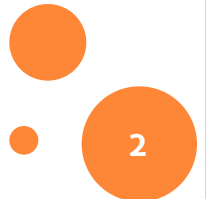


# INTRODUCCIÓN



# DESARROLLO DE APLICACIONES MÓVILES

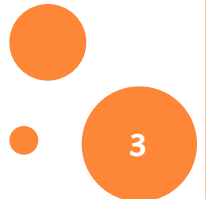
- Existen tres formas (básicas) de crear una aplicación:
  - Web Apps: sitios web diseñados para verse en pantallas de móviles (HTML5, JavaScript, CSS3)
  - Native Apps: se escriben en el propio lenguaje de la plataforma. Tienen acceso completo al HW
    - Aplicaciones nativas multiplataforma: Frameworks en los que se programa una vez, pero genera aplicaciones nativas para cada plataforma (Ej.: Xamarin)
  - Hybrid Apps: son Web apps empaquetadas para parecer una app nativa, con extensiones que proporcionan acceso a algunas funcionalidades del HW



# DESARROLLO DE APLICACIONES MÓVILES

## ○ Hybrid apps

- Ventajas:
  - Se reutiliza el código de la aplicación web
  - Adaptación a diferentes plataformas: sólo la funcionalidad extra
  - Costes muy bajos.
- Desventajas:
  - Necesidad de complementos para las funcionalidades extra
  - Las apps nativas siempre aprovechan mejor los recursos.



# DESARROLLO DE APLICACIONES MÓVILES

## ○ Web apps

- Ventajas:
  - Ofrecer también la funcionalidad en una web
- Desventajas:
  - Funcionalidad muy limitada

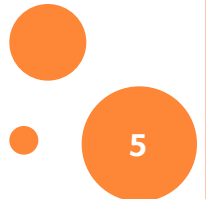
## ○ Native apps

- Ventajas:
  - Tienen mejor rendimiento.
  - Aprovechan mejor las posibilidades del HW.
  - Las más fiables
- Desventajas:
  - Hay que aprender el lenguaje de cada plataforma
  - Son las más caras de desarrollar



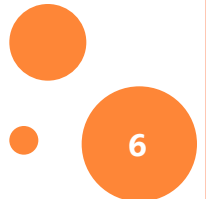
# ¿QUÉ ES ANDROID?

- Es un sistema operativo inicialmente pensado para dispositivos móviles.
- Desarrollado por Google
- Lanzado en 2008
- Actualmente se usa en otros dispositivos como televisores, coches, relojes, etc.



# VERSIONES DE ANDROID

- Hasta la versión 9, nombres de repostería
  - 1.0 → A: *Apple Pie*
  - 1.1 → B: *Banana Bread*
  - 1.5 → C: *Cupcake*
  - 1.6 → D: *Donut*
  - 2.0 → E: *Éclair*
  - 2.2 → F: *Froyo*
  - 2.3 → G: *Gingerbread*
  - 3.0 → H: *Honeycomb*
  - 4.0 → I: *Ice Cream Sandwich*
  - 4.1 → J: *Jelly Bean*
  - 4.4 → K: *KitKat*
  - 5.0 → L: *Lollipop*
  - 6.0 → M: *Marshmallow*
  - 7.0 → N: *Nougat*
  - 8.0 → O: *Oreo*
  - 9.0 → P: *Pie*
- La última versión es Android 15 (API 35 Vanilla Ice Cream)
  - Android 16 (versión Beta)



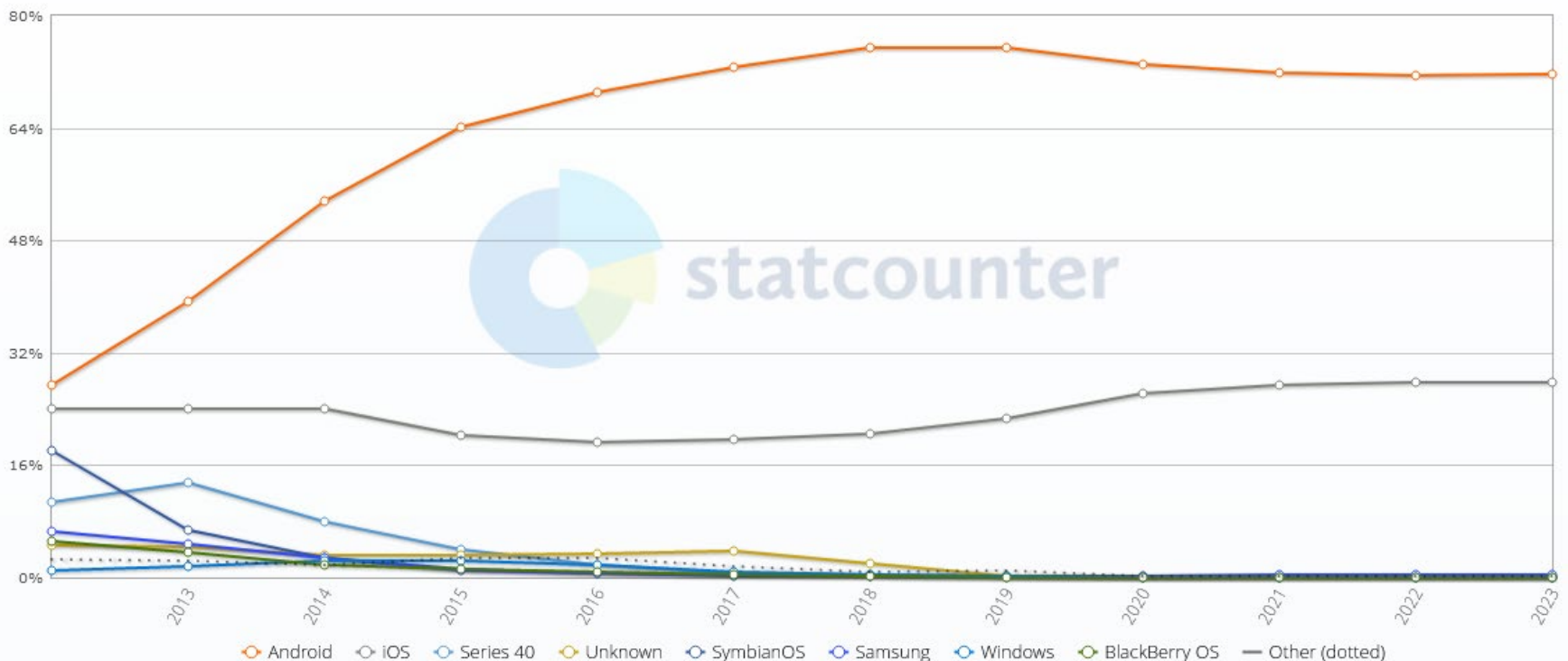
# ¿QUÉ ES ANDROID?

## ○ Mercado Diciembre 2023:

- 70.48% Android – 28.8% iOS – 0.72% Resto

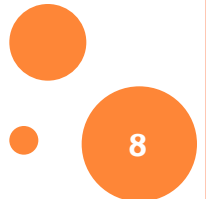
Mobile Operating System Market Share Worldwide  
2012 - 2023

Edit Chart Data



# EL GRAN PROBLEMA DE ANDROID: LA FRAGMENTACIÓN

- El desarrollo del S.O. es independiente de los dispositivos (excepto los propios de Google)
- El hardware se queda obsoleto muy pronto y no soporta las nuevas versiones del S.O. (incluso el hardware "propio" de Google)
- Las aplicaciones desarrolladas para una versión, no tienen por qué ser compatibles con la anterior, ni con la siguiente





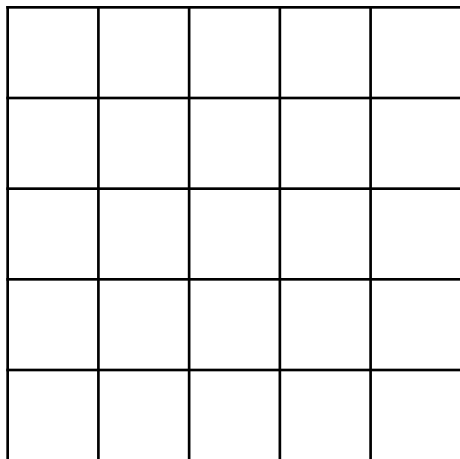
# EL GRAN PROBLEMA DE ANDROID: LA FRAGMENTACIÓN

ANDROID PLATFORM VERSION	API LEVEL	CUMULATIVE DISTRIBUTION
4.0 Ice Cream Sandwich	15	
4.1 Jelly Bean	16	99,8%
4.2 Jelly Bean	17	99,1%
4.3 Jelly Bean	18	98,2%
4.4 KitKat	19	97,9%
5.0 Lollipop	21	93,4%
5.1 Lollipop	22	91,4%
6.0 Marshmallow	23	83,3%
7.0 Nougat	24	71,0%
7.1 Nougat	25	62,8%
8.0 Oreo	26	56,9%
8.1 Oreo	27	49,1%
9.0 Pie	28	34,8%
10. Q	29	2,8%

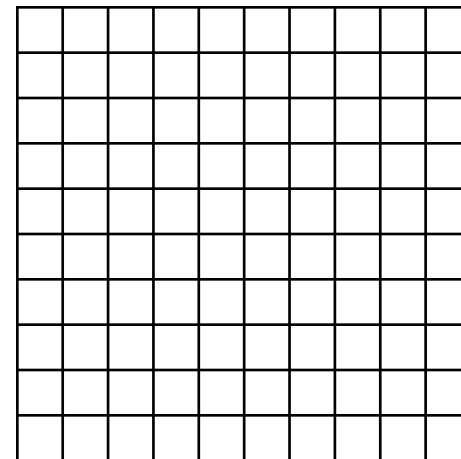


# EL GRAN PROBLEMA DE ANDROID: LA FRAGMENTACIÓN

- Múltiples fabricantes de hardware
- Cada uno decide tamaño y densidad de pantalla
  - Tamaño: se mide la diagonal de la pantalla en pulgadas (inches). 1 pulgada = 2,54 cm
  - Densidad: cantidad de puntos (píxeles) en una pulgada (dots per inch - dpi)



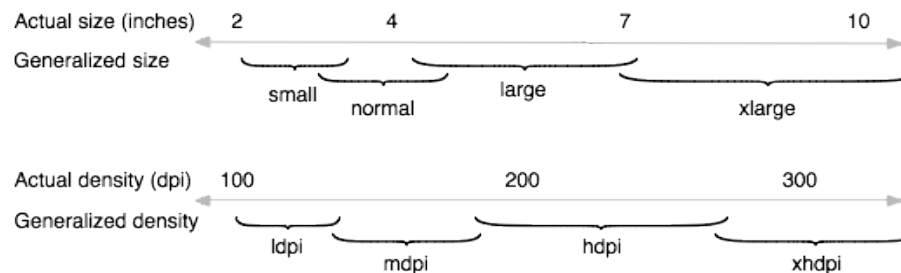
Mismo tamaño,  
distinta densidad



# EL GRAN PROBLEMA DE ANDROID: LA FRAGMENTACIÓN

- Categorías aproximadas (establecidas por Google)

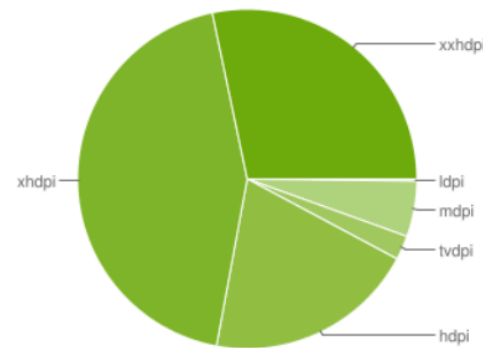
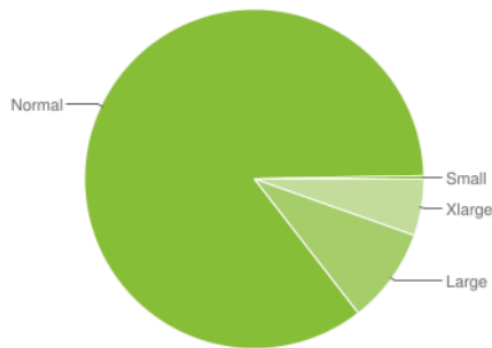
Categoría	Nombre	dpi (aprox.)
<i>ldpi</i>	Baja	120
<i>mdpi</i>	Media (densidad de referencia)	160
<i>hdpi</i>	Alta	240
<i>xhdpi</i>	Muy alta	320
<i>xxhdpi</i>	Muy, muy alta	480
<i>xxxhdpi</i>	Extremadamente alta	640



# EL GRAN PROBLEMA DE ANDROID: LA FRAGMENTACIÓN

- Múltiples combinaciones

	ldpi	mdpi	tvdpi	hdpi	xhdpi	xxhdpi	Total
Small	0.1%				0.1%		0.2%
Normal		0.4%	0.3%	17.0%	41.1%	25.9%	84.7%
Large		1.8%	2.0%	0.7%	2.6%	2.1%	9.2%
Xlarge		3.5%		1.9%	0.5%		5.9%
Total	0.1%	5.7%	2.3%	19.6%	44.3%	28.0%	












Datos recopilados durante un período de 7 días hasta el 30 de agosto de 2020.  
No se muestran configuraciones de pantalla con una distribución inferior al 0.1%.



# EL GRAN PROBLEMA DE ANDROID: LA FRAGMENTACIÓN

- Algunos ejemplos de dispositivos y densidades de pantalla:

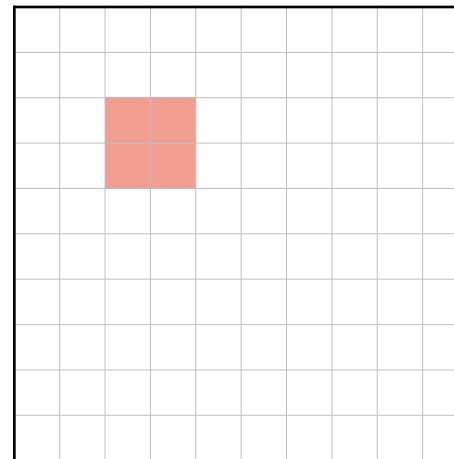
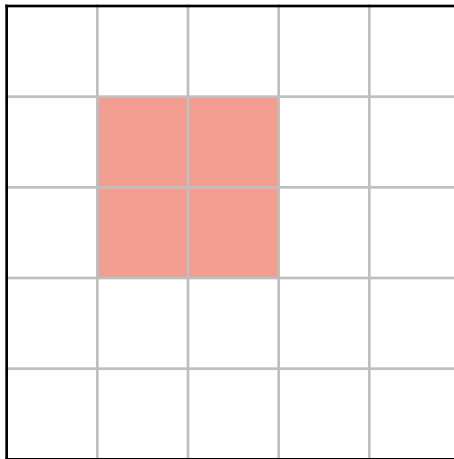
Type	Device	Platform	Screen dimensions in cm	Aspect Ratio	Width × Height dp	Width × Height px	Density
	Android One	Android	4.5 in 2.2 × 3.9 in	16 : 9	320 × 569 dp	480 × 854 px	1.5 hdpi
	Asus Zen Watch	Android	1.6 in 1.2 × 1.2 in	1 : 1	213 × 213 dp	320 × 320 px	1.5 hdpi
	Dell Venue 8	Android	8.4 in 4.5 × 7.1 in	16 : 10	800 × 1280 dp	1600 × 2560 px	2.0 xhdpi
	Google Pixel	Android	5.0 in 2.5 × 4.4 in	16 : 9	411 × 731 dp	1080 × 1920 px	2.6 xxhdpi
	Google Pixel XL	Android	5.5 in 2.7 × 4.8 in	16 : 9	411 × 731 dp	1440 × 2560 px	3.5 xxxhdpi
	HTC One M8	Android	5.0 in 2.5 × 4.4 in	16 : 9	360 × 640 dp	1080 × 1920 px	3.0 xxhdpi
	HTC One M9	Android	5.0 in 2.5 × 4.4 in	16 : 9	360 × 640 dp	1080 × 1920 px	3.0 xxhdpi
	Nexus 7 ('13)	Android	7.0 in 3.7 × 6.0 in	16 : 10	600 × 960 dp	1200 × 1920 px	2.0 xhdpi
	Nexus 9	Android	8.9 in 7.1 × 5.3 in	4 : 3	1024 × 768 dp	2048 × 1536 px	2.0 xhdpi



# EL GRAN PROBLEMA DE ANDROID: LA FRAGMENTACIÓN

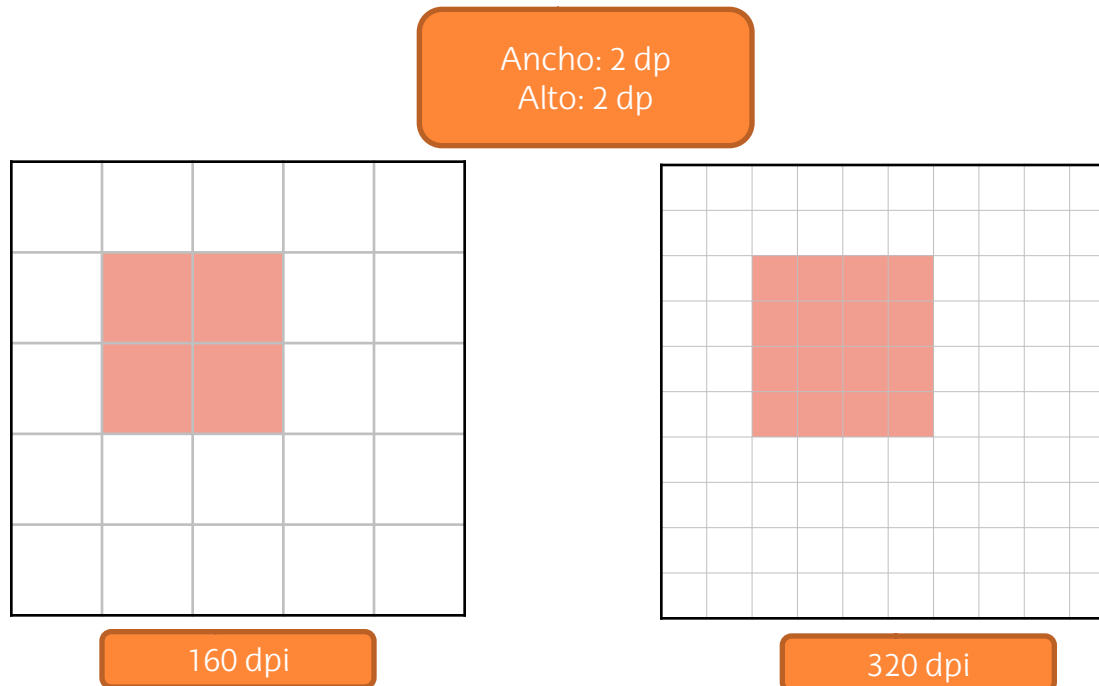
- Hay que evitar indicar tamaños usando píxeles como unidad

Ancho: 2px  
Alto: 2 px



# EL GRAN PROBLEMA DE ANDROID: LA FRAGMENTACIÓN

- Usaremos como unidad píxeles independientes de la densidad (dp o dip)
  - Equivale a un píxel en una pantalla de 160dpi
  - Android escalará automáticamente el tamaño a la densidad correspondiente



# EL GRAN PROBLEMA DE ANDROID: LA FRAGMENTACIÓN

- Al desarrollar una aplicación hay que tener en cuenta
  - Que funcione para la mayor cantidad de versiones del S.O. posibles
  - Que se pueda adaptar a distintos tamaños y resoluciones de pantalla
- [http://developer.android.com/guide/practices/screens\\_support.html](http://developer.android.com/guide/practices/screens_support.html)

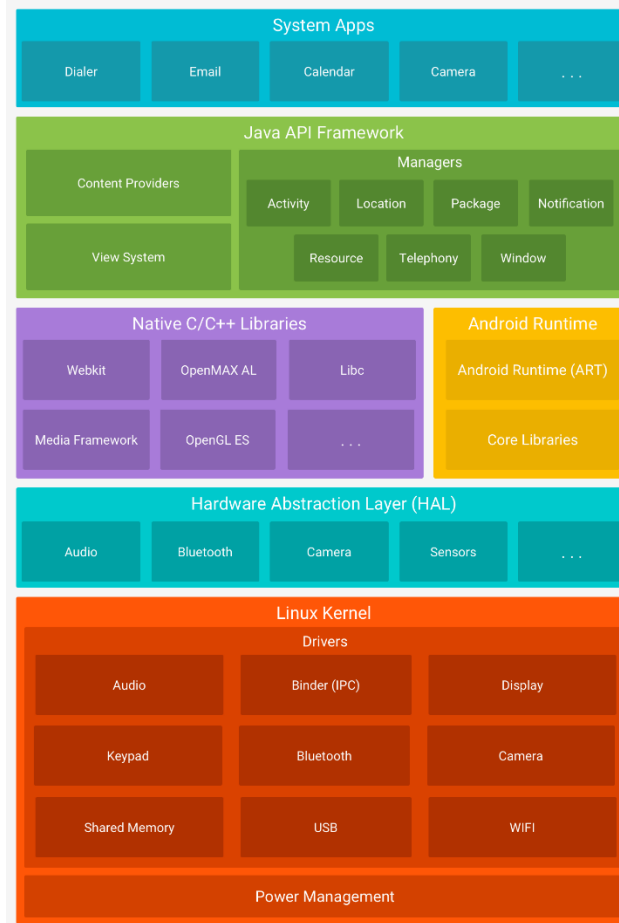
En Agosto de 2015 existían más de 24.000 tipos de dispositivos Android distintos producidos por 1300 marcas





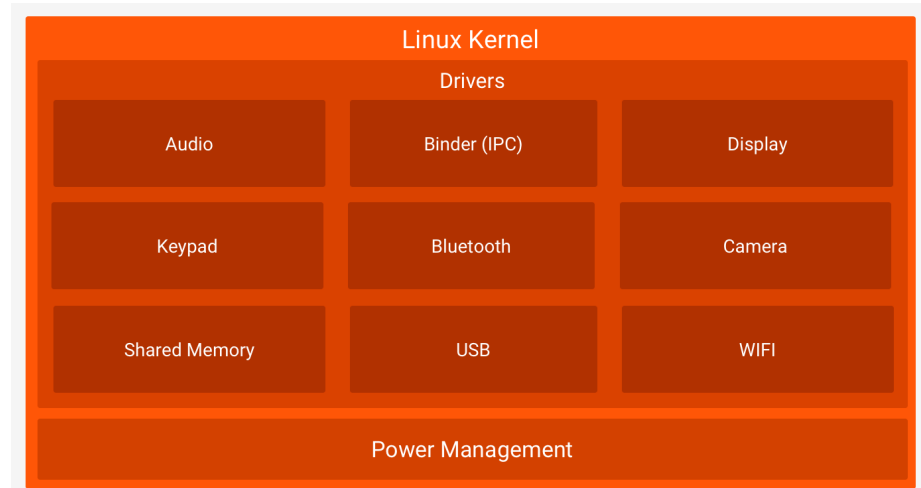
# ARQUITECTURA ANDROID

- Android tiene una arquitectura por capas



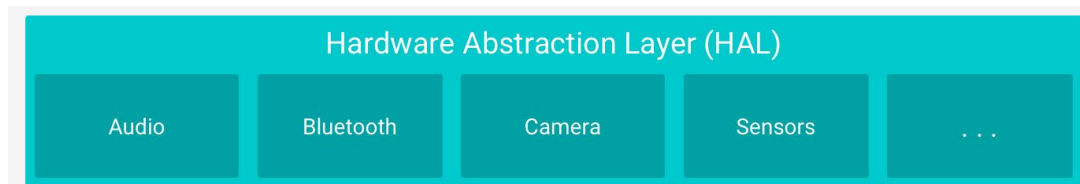
# ARQUITECTURA ANDROID

- Android usa un Kernel de Linux
  - Desde la versión 1.5 Cupcake
- Permite abstraer el software del hardware
- Gestiona:
  - La seguridad
  - La memoria
  - Los procesos
  - Red y modelo de drivers
- Cada fabricante lo adapta y compila para su hardware



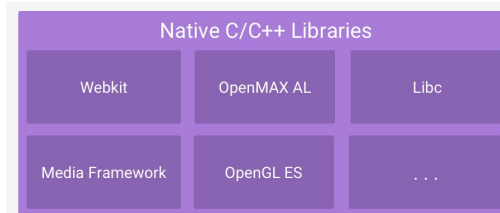
# ARQUITECTURA ANDROID

- Capa de abstracción de hardware
  - Ofrece interfaces para trabajar con el hardware del dispositivo
  - Evita tener que trabajar a bajo nivel
  - Cada módulo trabaja con un componente concreto del hardware
  - Cada fabricante adapta esos módulos a su hardware para que la interfaz funcione independientemente del hardware que se use



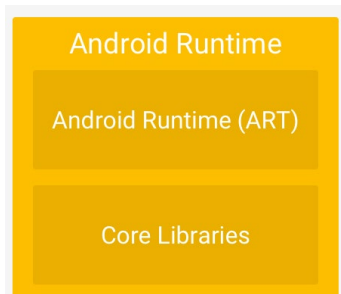
# ARQUITECTURA ANDROID

- Conjunto de librerías (C & C++) usadas por componentes del sistema
  - Surface Manager: Gestión del acceso a la pantalla.
  - Media Framework: Reproducción multimedia.
  - SQLite: Base de Datos.
  - WebKit: Navegador (Browser) optimizado.
  - SGL: Gráficos 2D.
  - Open GL | ES: Librerías 3D.
  - FreeType: Renderizado vectorial y fuentes bitmap.
  - SSL: Encriptación Secure Socket Layer
  - libc: todas las funciones de C



# ARQUITECTURA ANDROID

- Las Core Libraries son una implementación libre de Java (actualmente todo Java 7 y parte de Java 8)
- El Android RunTime (ART) es el entorno donde se ejecutan las aplicaciones
  - Cada aplicación en una instancia distinta
    - Seguridad
    - Resistencia a fallos
  - Al instalar la aplicación, la compila y almacena esa compilación (Ahead-of-Time, AOT) para su ejecución



# ARQUITECTURA ANDROID

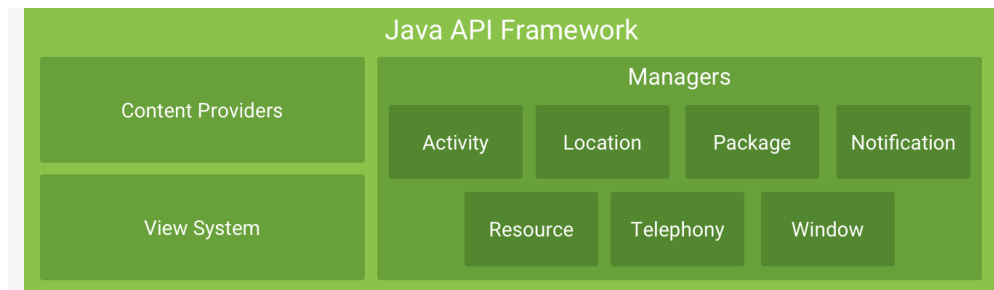
- Hasta Android 5, el entorno de ejecución era la Máquina Virtual Dalvik
  - Compilaba cada aplicación sobre la marcha, según se vaya usando la aplicación (Just-in-Time, JIT)
- ART vs Dalvik
  - La instalación es más lenta
  - El arranque de las aplicaciones es más rápido
  - Gasta menos energía
  - Ocupan un poco más de espacio

ART y DVM no son compatibles 100%



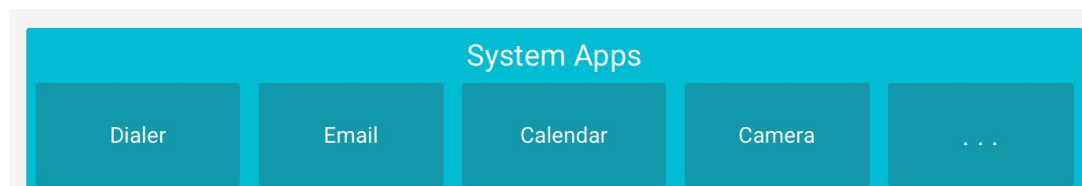
# ARQUITECTURA ANDROID

- El **JAVA API Framework** es el conjunto de herramientas para el desarrollo de aplicaciones
  - **Managers**: APIs que gestionan los componentes de las aplicaciones y los servicios que ofrece el sistema
  - **Content Providers**: APIs que permiten el acceso a datos como la agenda, el correo, etc.
  - **View System**: Contiene los elementos para construir interfaces gráficas: botones, etiquetas, etc.



# ARQUITECTURA ANDROID

- En esta capa se encuentran todas las aplicaciones del dispositivo
  - Las nativas
  - Las instaladas por el usuario
  - La principal del sistema: **launcher**
    - Permite lanzar otras aplicaciones
    - Muestra distintos escritorios
    - Permite poner **widgets** en los escritorios





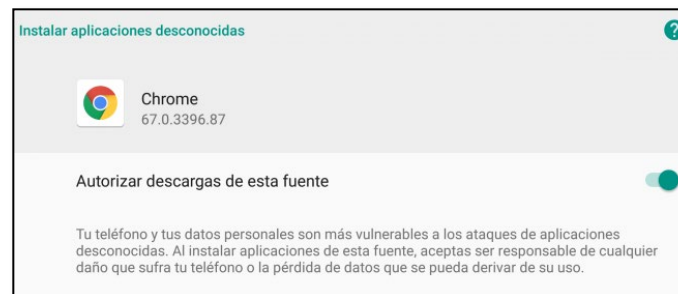
# APLICACIONES PARA ANDROID

- Se distribuyen en formato .apk (Application Package File)
  - Es una variante de .jar
  - Se pueden examinar usando un software de descompresión
- Se pueden instalar directamente desde la tienda de aplicaciones oficial
  - Play Store: <http://play.google.com>



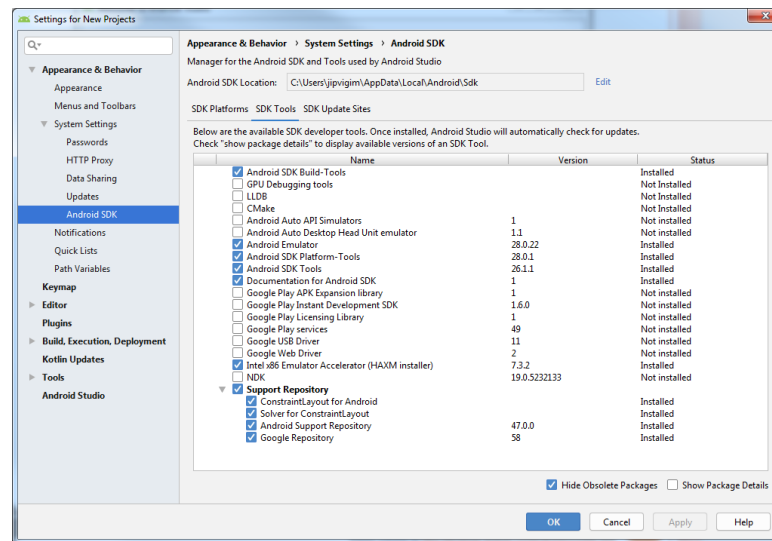
# APLICACIONES PARA ANDROID

- Se puede configurar el dispositivo para que permita instalar los ficheros .apk sin pasar por la tienda oficial
  - En versiones anteriores a Oreo (Android 8)
    - Ajustes → Seguridad → Origen desconocido (activar)
  - En Oreo y superiores
    - Hay que dar permiso para instalar aplicaciones a la aplicación que acceda al .apk
      - Por ejemplo
        - Descargamos un .apk desde el navegador
- En Ajustes activamos la opción de “Autorizar descargas de esta fuente”



# HERRAMIENTAS DE DESARROLLO

- JDK (Java Development Kit)
- Android SDK (Software Development Kit)
  - Conjunto de herramientas de desarrollo
  - Incluye
    - Las herramientas de depuración y pruebas (SDK Tools)
    - Imágenes para los emuladores
    - Las API de Google
    - Documentación
    - Ejemplos
    - .....



# HERRAMIENTAS DE DESARROLLO

- Android Studio (Google)
  - Es el entorno de desarrollo oficial
  - Usa una herramienta de automatización de tareas (Gradle) para cosas como
    - Compilación
    - Despliegue
  - Admite trabajar con Java, Kotlin y otros lenguajes



# KOTLIN

- Es un lenguaje de programación estáticamente tipado, diseñado para ser interoperable con Java
- Creado por JetBrains
- Corre sobre la máquina virtual de Java
- Historia con Google:
  - Desde octubre 2017 disponible en Android Studio
  - Desde mayo 2019 es considerado “de primer nivel”
- No lo veremos en esta asignatura



# KOTLIN

## ○ Ejemplo:

- Java:

```
public class MainActivity extends AppCompatActivity {  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
}
```

- Kotlin:

```
class MainActivity : AppCompatActivity() {  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
    }  
}
```

## ○ Los ejemplos de la documentación oficial de Android están en Java y Kotlin



# EN EL DISPOSITIVO

- Activar las opciones de desarrollo
  - Ajustes → Información del teléfono → Número de compilación (\*)
  - Hay que pulsarlo 7 veces seguidas
  - En el menú de Ajustes aparecerá una nueva opción "Opciones de desarrollo"
    - Activar la opción "Depuración de Android"
  - Más información:
    - <https://developer.android.com/studio/debug/dev-options#enable>

(\*) Puede variar según modelo

