

Análisis de tecnologías para aplicaciones en dispositivos móviles.

Caso práctico

El uso generalizado en los últimos tiempos de dispositivos móviles como **smartPhones** o **tablets** ha dado lugar a una gran demanda de software para este nuevo tipo de hardware. Es por esta razón que **Ada**, socia fundadora de la empresa **BK Programación** y con cierta experiencia en el desarrollo de aplicaciones para dispositivos móviles, ha decidido que su compañía entre también en este nuevo mercado. Se trata de una oportunidad interesante para abrir nuevos negocios tanto para sus clientes actuales como para la captación de otros nuevos.

Para ello necesita formar a sus trabajadores, Juan y María, que acaban de llegar a la oficina.

-¡Buenos días! -saluda Ada-, tengo un nuevo reto para vosotros. Os voy a formar en el desarrollo de aplicaciones para dispositivos móviles, debemos expandirnos en ese mercado.

-Suena interesante -responde María-, ¿tienes experiencia en ese campo?

-Sí, he desarrollado aplicaciones en Java para proyectos Java ME y Android Studio.

-Acepto el reto -dice Juan-, ¿cuándo comenzamos?



[Mikhail](#) ([Pexels](#))



[Ministerio de Educación y Formación Profesional](#). (Dominio público)

Materiales formativos de FP Online propiedad del Ministerio de Educación y Formación Profesional.

[Aviso Legal](#) 

1.- Primeros conceptos

Caso práctico

Esa misma mañana Ada comienza la formación de María y Juan.

-¿Os habéis dado cuenta del incremento que está habiendo en el uso de dispositivos móviles? -pregunta María-. Quiero que penséis en los diferentes dispositivos móviles que utilizáis en vuestro día a día y por qué los utilizáis, es decir, cuáles son sus ventajas e inconvenientes.

-Haremos la lista de dispositivos móviles -responde María- con características favorables y sus limitaciones.

-Y una vez hecho esa lista -prosigue Ada- debéis plantearos qué debéis tener en cuenta a la hora de desarrollar una aplicación para un dispositivo móvil. En concreto, la tecnología a utilizar.

Juan, viendo que necesitan informarse sobre las características (hardware, software, etc.) de los diferentes dispositivos móviles antes de que Ada continúe su formación, lo ve claro...

-María, nos toca investigar sobre las tecnologías disponibles para el desarrollo de aplicaciones en dispositivos móviles.



[Mikhail \(Pexels\)](#)

El uso generalizado en los últimos tiempos de dispositivos móviles como smartphones, tablets y PDAs ha dado lugar a una gran demanda de software para este nuevo tipo de hardware.

En los próximos apartados, cubriremos de forma general los aspectos más importantes de estas nuevas tecnologías que cada vez están más presentes en el día a día de millones de usuarios, que muchas veces se incorporan a la era digital apoyándose en estos dispositivos sin tan siquiera tener conocimientos previos de informática a nivel de usuario, o prácticamente sin haber utilizado apenas un ordenador. La inmediatez, hiperconectividad y movilidad son aspectos claves para que ya, en la actualidad, la tecnología smartphone y de dispositivo móvil tenga más usuarios en el mundo que las propias tecnologías de PC.



[Pixabay \(CC0\)](#)

1.1.- Introducción. ¿Qué es un dispositivo móvil?

La primera pregunta que podemos hacernos es ¿qué entendemos por móvil? Si se nos ocurre investigar sobre ese término, a través de algún buscador de Internet, podremos observar que no hay una respuesta única y que en algunas ocasiones las diferencias pueden ser sustanciales en función de qué es lo que consideremos "móvil". Obviamente esto da lugar a su vez a muchas otras preguntas como por ejemplo: Es móvil... ¿alguna parte del dispositivo? ¿El dispositivo completo? ¿La aplicación que usamos en el dispositivo? ¿Una aplicación cliente? ¿Una aplicación servidor? ¿El usuario del dispositivo? ¿Es "móvil" sinónimo de "portátil"? ¿Es "móvil" sinónimo de "limitado"? ¿Hasta qué punto "móvil" es sinónimo de "autónomo"? ¿Existen diversos grados de "movilidad"? ¿Se pueden clasificar los dispositivos móviles en distintos tipos? ¿Bajo qué criterios? Y así sucesivamente podríamos plantearnos más y más preguntas...

Para evitar este tipo de controversias, en nuestro caso vamos a intentar dar una definición con la que trabajaremos a lo largo del desarrollo del módulo.

¿Qué es un dispositivo móvil?

Se trata de un aparato de pequeño tamaño (normalmente que quepa en un bolsillo) y de poco peso, con pantalla y teclado, con pequeñas capacidades de procesamiento, memoria limitada y conexión (permanente o no) a una red. Este tipo de dispositivos están diseñados para cumplir algún tipo de función específica (realizar llamadas telefónicas, servir como agendas, jugar, navegación GPS, escuchar música, acceso al correo electrónico, navegar por Internet, proporcionar servicio de chat con otros dispositivos móviles, etc.) aunque normalmente pueden llevar a cabo también funciones más generales.

Estos aparatos son cada vez más populares especialmente para aquellos entornos en los que llevar consigo un ordenador convencional (incluso un portátil) no es práctico.

Por otro lado en la definición anterior, en relación a la capacidad de proceso hemos apuntado que tienen "pequeñas capacidades de proceso", lo cual es una verdad a medias, puesto que hoy en día, los dispositivos más avanzados pueden superar a equipos no móviles de gama media-baja de hace apenas 5 ó 6 años. Algo similar ocurre con la memoria, siendo ya frecuente encontrar en el mercado teléfonos móviles con 4 y 6 GB de RAM, e incluso valores más elevados.

Para saber más

Para obtener más información acerca de cuáles pueden ser las características de un dispositivo móvil puedes consultar el siguiente [artículo en la Wikipedia sobre dispositivos móviles](#). 

Clasificación de los dispositivos móviles.

¿Qué tipos de dispositivos móviles existen?

Entre los dispositivos móviles más habituales se encuentran:

- ✔ SmartPhones o "teléfonos inteligentes".
- ✔ PDA (Personal Digital Assistant - asistentes digitales personales) o PocketPC (PC de bolsillo).
- ✔ Handheld, o PCs de mano.
- ✔ Internet tables, que se encontrarían entre las PDA y los PC Ultramóviles (pequeños tablet PC).



[Banco de imágenes de INTEF](#) (CC BY-NC-SA)



[Jeremy Keith](#) (CC BY)

Según las fuentes que consultes puedes encontrar diversas clasificaciones donde se incluyan unos u otros tipos de dispositivos, como por ejemplo:

- ✔ Pagers (o buscapersonas), hoy día ya en desuso.
- ✔ Navegadores GPS.
- ✔ E-Readers (lectores de libros digitales).
- ✔ Pequeñas videoconsolas de mano.
- ✔ Cámaras digitales.
- ✔ Calculadoras programables.

En nuestro caso, los principales aparatos a los que nos referiremos al hablar de dispositivos móviles serán los **smartPhones** y las **tablets**. Por otro lado, según la tecnología vaya avanzando te podrás ir encontrando con nuevos tipos de productos y servicios que irán ampliando las posibilidades de elección. Como siempre sucede en el mundo de la tecnología habrá que estar continuamente al día de los nuevos avances que van apareciendo para no quedarse atrás.

1.2.- Limitaciones de las tecnologías móviles.

Antes de comenzar a desarrollar software para alguno de estos dispositivos, es necesario ser conscientes de las limitaciones con las que nos podemos encontrar en estos aparatos. ¿Cuáles son las restricciones a las que nos vamos a tener que enfrentar?

Algunas de estas restricciones son:

- ✓ Suministro de energía limitado (normalmente dependiente de baterías).
- ✓ Procesadores con capacidad de cómputo reducida. Suelen tener una baja frecuencia de reloj por la necesidad de ahorrar energía. En algunos casos, por ejemplo, podrían no disponer de la capacidad de cálculos en punto flotante.
- ✓ Poca memoria principal (RAM).
- ✓ Almacenamiento de datos persistente reducido (pequeña memoria flash interna, tarjetas SD, etc.).
- ✓ Conexión a algún tipo de red intermitente y con ancho de banda limitado.
- ✓ Pantallas de reducidas dimensiones.
- ✓ Teclados con funcionalidad muy básica y muy pequeños.



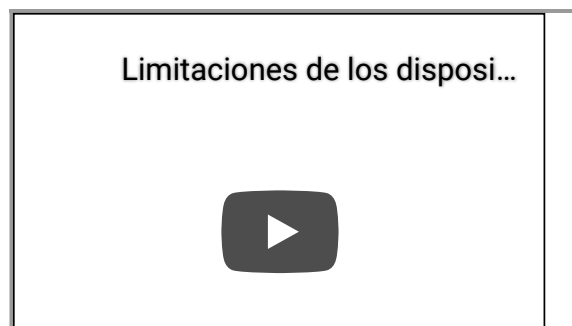
[Rubaitul Azad](#) ([Licencia de pexels](#))

Este tipo de restricciones, y algunas otras que dependerán de cada dispositivo en concreto, habrán de ser tenidas muy en cuenta a la hora del análisis y diseño de una aplicación "**móvil**", pues no podemos pretender, que esa aplicación pueda contener la misma funcionalidad, que la que podemos encontrar habitualmente en un programa que es ejecutado en un ordenador de sobremesa o un portátil.

Por otro lado, no todo va a ser restricciones. También habrá que tener en consideración que esta tecnología va a aportar una serie de ventajas muy importantes: movilidad, poco peso, pequeño tamaño, facilidad para el transporte, conectividad a diversos tipos de redes de comunicaciones (mensajería SMS y MMS; voz; Internet; Bluetooth; infrarrojos; radiofrecuencia, etc.). Ésas serán las ventajas que podrás explotar en tus aplicaciones.

Para saber más

A través del siguiente vídeo puedes aprender más sobre las características y limitaciones de los dispositivos móviles que han ido solucionándose en los nuevos dispositivos.



[Natalia Arroyo](#). ([Todos los derechos reservados](#))

[Descripción textual del vídeo](#) 

1.3.- Tecnologías disponibles.

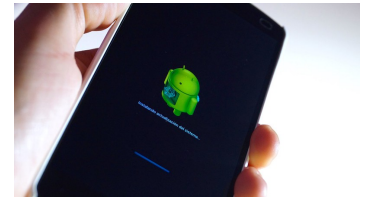
Programar para un dispositivo móvil no va a ser exactamente lo mismo que programar para un ordenador convencional, debido a las limitaciones y características especiales que aquellos aparatos pueden presentar. Hay que conocer qué tipos de tecnologías se pueden encontrar en este ámbito, tanto a nivel de hardware (dispositivos sobre los que se realizarán las aplicaciones) como a nivel de software (sistemas operativos que funcionan sobre esos dispositivos, plataformas de desarrollo disponibles, entornos, APIs, lenguajes de programación, etc.).

Cuando vas a desarrollar una aplicación para un dispositivo móvil, algunas de las primeras preguntas que te puedes hacer son:

- ✓ ¿Sobre qué tipos de dispositivos móviles se pueden hacer programas? ¿Sobre qué tipo de hardware se puede programar?
- ✓ ¿Qué sistema operativo puede llevar ese hardware?
- ✓ ¿Qué plataformas de desarrollo existen para desarrollar sobre ese hardware y ese sistema operativo? ¿con qué lenguajes puedo programar? ¿qué herramientas (compiladores, bibliotecas, entornos, etc.) hay disponibles?

Las respuestas a este tipo de preguntas pueden ser múltiples y muy variadas:

- ✓ Respecto al hardware, te puedes encontrar, como has visto ya, principalmente con teléfonos móviles (smartPhones) y tablets (las PDA están actualmente completamente en desuso). Entre los principales fabricantes de teléfonos móviles se encuentran **Samsung, Apple, Huawei, HTC, LG, Motorola, Sony Ericsson, Nokia, Alcatel-Lucent, Xiaomi**, etc. Además en los últimos años fabricantes chinos como **Oppo** y **Vivo** han conseguido hacerse un hueco importante en el mercado, desbancando a muchos de los fabricantes anteriormente mencionados.
- ✓ En cuanto a los sistemas operativos, dependiendo del hardware habrá sistemas diseñados para unos u otros dispositivos. Los hay basados en **Microsoft Windows**, en **Linux**, y en **MAC OS X**, así como otros totalmente originales y desarrollados específicamente para estos nuevos tipos de dispositivos. Entre los más populares se encuentran , **Android, iOS, Symbian OS, Blackberry OS** y **Windows Phone**.
- ✓ Si lo que deseas es conocer algo acerca de las plataformas de desarrollo disponibles para cada entorno (hardware y/o sistema operativo), podemos hablar de **Java ME, Windows Mobile SDK, Maemo SDK, Xamarin** o bien de IDEs como **Microsoft Visual Studio, CodeWarrior, Eclipse, Netbeans** y últimamente, con mucha presencia entre programadores profesionales, **Android Studio** .
- ✓ Si te refieres a lenguajes de programación, normalmente te encontrarás con lenguajes que son ya viejos conocidos para otras plataformas, como pueden ser las aplicaciones de escritorio para los PCs o las aplicaciones web (**Java, C#, C, Kotlin**, etc.).



Eduardo Woo (CC BY-SA)

En definitiva puedes observar que en este nuevo mundo del desarrollo para dispositivos móviles te encuentras con una problemática similar a la que te puedes enfrentar con los ordenadores convencionales: distintos tipos de hardware, distintas opciones de sistemas operativos dependiendo del hardware que los soporte, diferentes lenguajes de programación, plataformas, API y bibliotecas, entornos de desarrollo, etc.

Para saber más

A través del siguiente [enlace](#)  puedes aprender más sobre las PDAs.

Y mediante este otro puedes saber algo más sobre [Xamarin](#)  .



[Ryan McVay](#) (CC BY-NC-SA)

1.3.1.- Hardware

Como has visto en los apartados anteriores, dependiendo de los criterios que se utilicen para clasificar los dispositivos móviles se puede hablar de más o menos tipos. Este curso se va a centrar sobre todo en smartPhones y tablets.

Smartphones

Se puede definir un smartPhone o "teléfono inteligente" como un terminal de telefonía móvil que proporciona unas prestaciones y una funcionalidad mayor que la que podría ofrecer un teléfono móvil normal. Hoy día una buena parte de los teléfonos móviles que se pueden adquirir en el mercado son de este tipo.

Este tipo de terminales se caracteriza por tener instalado un sistema operativo y por tanto la posibilidad de ejecutar aplicaciones desarrolladas bien por el propio fabricante del terminal, bien por el operador de telefonía móvil, o bien por un tercero (empresa de desarrollo de software).

Algunas otras características que suelen tener este tipo de dispositivos son:

- ✓ Funcionamiento en multitarea (ejecución concurrente de varios procesos en el sistema operativo).
- ✓ Acceso a Internet.
- ✓ ConectividadWi-Fi, Bluetooth, etc.
- ✓ Posibilidad de ampliación de memoria mediante tarjetas externas de memoria (por ejemplo SD).
- ✓ Pequeñas pantallas pero de alta resolución y/o con millones de colores.
- ✓ Posibilidad de pantallas táctiles o incluso multitáctiles (multitouch).
- ✓ Receptor GPS.
- ✓ Cámaras digitales integradas. Capacidades fotográficas. Grabación de audio y vídeo.
- ✓ Posibilidad de conexión con un ordenador para cargar y descargar información. Normalmente con conexión USB o bien una conexión inalámbrica.
- ✓ Sensores (de orientación, de temperatura, de presión,acelerómetros,magnetómetros, etc.).
- ✓ Receptor de radio FM.
- ✓ Emisor de radio FM.
- ✓ Posibilidad de instalar y ejecutar aplicaciones sofisticadas:
 - Aplicaciones de asistente personal (gestión de contactos, calendarios, citas, agendas, alarmas, etc.).
 - Gestión del correo electrónico.
 - Gestión del sistema archivos del dispositivo.
 - Microaplicaciones de ofimática (procesador de textos, hoja de cálculo, etc.).
 - Aplicaciones multimedia (reproducción de audio y vídeo en diversos formatos).
 - Aplicaciones de cartografía y navegación.
 - Diccionarios.
 - Pequeñas aplicaciones científicas (matemáticas, física, medicina, etc.).
 - Juegos.
 - Aplicaciones de mensajería instantánea. Chats.



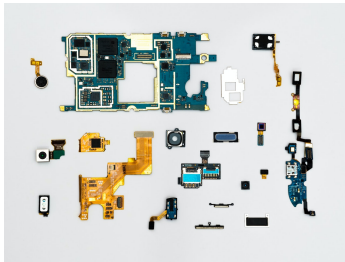
[Vojta Kovařík](#) (Licencia [pexels.com](#))

Puedes observar que aunque el dispositivo es un teléfono móvil muchas veces su uso principal no va a ser necesariamente el de un teléfono (hacer y recibir llamadas) sino que podrá estar dedicado a

muchos otros usos (hacer fotos, navegar por Internet, reproducir archivos de audio, jugar, gestionar la agenda personal, consultar un mapa, usar un diccionario, escuchar la radio, ver una película, trazar una ruta para el navegador por satélite, etc.).

Dentro de muy poco tiempo es probable que se deje de emplear el término smartPhone o "móvil inteligente" pues todos los terminales móviles serán de este tipo y se hablará simplemente de "teléfonos móviles". De hecho en la práctica ya ocurre hoy día, pues cuando nos acercamos a algún catálogo de productos la inmensa mayoría de terminales que se nos ofrecen son de este tipo.

El interior de un smartphone



[Dan Cristian Păduret](#) ([Licencia Pexels.com](#))

Si se te ocurriera abrir un teléfono móvil actual te encontraríamos una placa de circuito impreso con una serie de dispositivos electrónicos integrados o insertos en ella. En cierto modo podría recordar un poco a la placa base de un PC. Entre los dispositivos que podrías observar se encontrarían el procesador principal, procesadores digitales de señal (DSP), procesadores de imagen y de audio, módem, memorias caché, co-dec de audio y vídeo, etc. A este conjunto de dispositivos integrados en la placa se le suele llamar **chipset** (lo cual te debería recordar al mundo del hardware de los PCs y a los conceptos estudiados en el módulo **Sistemas Informáticos**).

La principal arquitectura de microprocesadores utilizada para telefonía móvil es la diseñada por la empresa **ARM** ("Advanced RISC Machines", hoy día ARM Holdings), teniendo actualmente más del 90% de la cuota de mercado no ya sólo para teléfonos móviles, sino en general para todo tipo de pequeños dispositivos electrónicos que necesitan un microprocesador. Algunas de las familias de micros **ARM** más conocidas son las **ARM7**, **ARM9**, **ARM11** y **Cortex**. Hoy día muchas otras empresas diseñan microprocesadores basados en las patentes de la arquitectura de los procesadores **ARM** como por ejemplo **DEC**, **IBM**, **Texas Instrument**, **Samsung**, **Intel**, **Atmel**, **Alcatel-Lucent**, **Apple**, **Qualcomm**, **LG**, **Nintendo**, **Philips**, **Oki**, **Yamaha**, etc. Uno de los más conocidos es por ejemplo el **Intel XScale**.

Algunos ejemplos de productos que incorporan uno o varios micros ARM son: la mayoría de teléfonos móviles (Samsung, Apple, etc.), el iPod de **Apple**, las consolas **Gameboy** y **Nintendo DS**, routers, navegadores GPS, cámaras digitales, etc.

Respecto al chipset, existen diversas posibilidades de arquitectura, según sea el fabricante que las produce. Algunas de las arquitecturas más conocidas son **MSM** (de la empresa **Qualcomm**), **OMAP** (Open Multimedia Application Platform, de la empresa Texas Instrument), y las de la empresa **Marvell**.

Muchos teléfonos de **Nokia** utilizan la arquitectura **OMAP** de **Texas Instrument** (por ejemplo el **Nokia N810 WiMax** Edition que incorpora un chipset **OMAP 2420** con un procesador **ARM1136**), mientras que algunas **BlackBerries** hacen uso de la **MSM** de **Qualcomm** (por ejemplo la **BlackBerry Storm**, con una arquitectura **MSM7600**) o de arquitecturas **Marvell** (por ejemplo la **BlackBerry Bold**, con un chipset **Marvell Travor PXA930**). Algunos modelos de **Samsung** también se han decantado por **Marvell** (por ejemplo el **Samsung Omnia**, que incorpora un chipset **Marvell** a 624 Mhz).

2.- Sistemas operativos.

Caso práctico

María y Juan ya tienen claro qué es un dispositivo móvil y las diferentes tecnologías disponibles. Ada continúa la formación de sus empleados...

-Como habéis descubierto, cada hardware maneja un sistema operativo diferente.

-Sí -responde María-, hay que tener en cuenta el hardware del dispositivo móvil.

-Y el hardware depende del fabricante -añade Juan.


-Exacto -confirma Ada-, por lo ahora os toca conocer las principales características de los sistemas operativos más utilizados actualmente en el mercado.



[Mikhail \(Pexels\)](#)

Los sistemas operativos más habituales que te puedes encontrar en un dispositivo móvil son:

- ✓ **Android**. Desarrollado inicialmente por **Google** y basado en el núcleo de **Linux**. El primer fabricante de móviles que lo incorporó fue **HTC**.
- ✓ **iOS**. Desarrollado por **Apple** para el **iPhone** y usado más tarde también para el **iPod Touch** y el **iPad**.
- ✓ **Windows Phone** (anteriormente llamado **Windows Mobile** y posteriormente derivado en **Windows 10 Mobile**) (discontinuado). Desarrollado por **Microsoft** tanto para smartPhones como para otros dispositivos móviles (por ejemplo PDA). Algunos fabricantes de teléfonos móviles que incorporaban este sistema operativo son Samsung, LG o HTC.
- ✓ **Blackberry OS** (discontinuado). Desarrollado por **Research in Motion (RIM)** para sus dispositivos **Blackberry**.

Según la fuente [Global stat \(statcounter.com\)](https://www.statcounter.com)  en diciembre de 2021, la cuota de mercado de sistemas operativos para móviles se encuentra distribuida de la siguiente forma a nivel mundial:

Sistema Operativo	Android	iOS	Samsung	KaiOS	Unknown	Nokia Unknown	Windows	Series 40	Linux
Cuota de Mercado	70.01%	29.24%	0.43%	0.13%	0.12%	0.002%	0.01%	0.01%	0.01%

Según esa misma fuente y para la misma fecha, la cuota de mercado de sistemas operativos para tabletas se encuentra distribuida de la siguiente forma a nivel mundial:

Sistema Operativo	iOS	Android	Windows	Linux
Cuota de Mercado	55.66%	44.25%	0.04%	0.02%

Si además trasladamos estas estadísticas a nivel nacional, se puede decir que en España la cuota de mercado de Android en smartphones supera el 78% del mercado.

Vamos a hacer un rápido repaso de algunos de los sistemas operativos más conocidos: **Android**, **iOS** e **Windows Phone**.

2.1.- Android.



[Android, Google \(Wikimedia Commons\)](#)

Android fue inicialmente desarrollado por **Android Inc.**, hoy día parte de la compañía **Google**. Está basado en una versión modificada del kernel de **Linux**.

El primer fabricante que incorporó **Android** en sus dispositivos fue **HTC** con su terminal **HTC Dream** (comercializado también como **T-Mobile G1** y popularmente conocido con los nombres de **Google Phone** o **GPhone**) en 2008.

Es uno de los más "jóvenes" dentro del grupo de sistemas operativos para dispositivos móviles y se ha hecho un notable hueco alcanzando más de las dos terceras partes de la cuota de mercado actual. Existen miles de aplicaciones que funcionan sobre **Android**, con un crecimiento cada vez mayor.

Hoy día podemos encontrar multitud de fabricantes que incorporan **Android** en sus terminales: **Samsung** (p.e., series **Galaxy** y **Nexus**), **Xiaomi**, **Huawei** (en las últimas versiones no se incorpora por políticas de Google), **Oppo**, **Vivo**, **LG**, **Sony Ericsson** (p.e., series **Xperia**), **HTC** (p.e., **Series A**), **Acer** (p.e. series **beTouch**), **Motorola** (p.e., **Droid X**), **Garmin**, **Dell**, **Alcatel**, etc.

A diciembre de 2021, la última versión, de Android es 12.0. Hasta las versión Android tenía un simpático criterio de asignación de nombres para las diferentes versiones que se correspondían con postres y que avanzaban la versión según avanzaba alfabéticamente la letra del postre.


Para desarrollar aplicaciones sobre **Android** se necesita el kit de desarrollo de software para Android (**Android SDK**), proporcionado gratuitamente por **Google**. Este paquete incluye todo lo necesario para construir aplicaciones sobre el entorno **Android** (depurador, bibliotecas, emulador, documentación, etc.). El lenguaje de programación que se utiliza es **Java** (y por tanto es necesario el **JDK** de **Oracle** para poder compilar programas **Java**).

El IDE oficial ha sido, durante varios años, **Eclipse** (a partir de la versión 3.2) junto con el plugin **ADT (Android Development Tools** - herramientas de desarrollo para **Android**). Sin embargo, las desavenencias entre **Oracle** (propietaria de **Java**) y **Google** (propietaria de **Android**) en los últimos años han provocado que **Android** abandonara la recomendación de uso de la versión ME de **Eclipse** para dispositivos móviles y haya desarrollado su propio IDE partiendo del IDE [IntelliJ IDEA de la compañía JetBrains](#) , convirtiéndose, a partir de finales del año 2013, en el nuevo IDE oficial para el desarrollo de aplicaciones para **Android**.

Nombre código	Número de versión	Fecha de lanzamiento
Apple Pie ¹⁵	1.0	23 de septiembre de 2008
Banana Cream ¹⁶	1.1	9 de febrero de 2009
Capote	1.5	26 de abril de 2009
Crumbl	1.6	19 de septiembre de 2009
Donat	2.0 - 2.1	26 de octubre de 2009
Froyo	2.2 - 2.2.3	20 de mayo de 2010
Gingerbread	2.3 - 2.3.7	6 de diciembre de 2010
Honeycomb ¹⁸	3.0 - 3.2.6	22 de febrero de 2011
Ice Cream Sandwich	4.0 - 4.0.5	19 de octubre de 2011
Jelly Bean	4.1 - 4.3.1	9 de julio de 2012
Kiklat	4.4 - 4.4.4	31 de octubre de 2013
Lollipop	5.0 - 5.1.1	12 de noviembre de 2014
Marshmallow	6.0 - 6.0.1	5 de octubre de 2015
Nougat	7.0 - 7.1.2	15 de junio de 2016
Oreo	8.0 - 8.1	21 de agosto de 2017
Pie	9.0	6 de agosto de 2018
10	10.0	3 de septiembre de 2019
11	11.0	8 de septiembre de 2020
12	12.0	4 de octubre de 2021

[Wikipedia](#). Captura de pantalla con las versiones de Android (Dominio público)

Para saber más

Para saber más sobre Android se pueden consultar los [artículos de la Wikipedia sobre este sistema operativo](#) , así como los documentos a los que hace referencia:

2.2.- iOS.

Se trata del sistema operativo desarrollado por **Apple** originalmente para su **iPhone**, aunque hoy día también es utilizado por otros dispositivos de la empresa.

Apple sólo permite que este sistema operativo funcione sobre hardware **Apple**. Es un sistema operativo derivado del **Mac OS X**, también de **Apple**.

En enero de 2022, según www.apple.com, la tienda de aplicaciones de **Apple** (servicio de descarga de aplicaciones), llamada **App Store**, dispone ya de 1,8 millones de aplicaciones para dispositivos con iOS. En la misma fecha, la cuota de mercado de Apple en los dispositivos móviles es del 29,21% según gs.statcounter.com .




[LoboStudioHamburg](https://www.pixabay.com/author/LoboStudioHamburg/) (Pixabay License)


La última versión de **iOS** en octubre de 2021 es la versión de 15.0.2. Como se ha comentado antes, los únicos dispositivos que incorporan este sistema operativo son los distintos modelos de dispositivos fabricados por la propia compañía **Apple**. Es decir, los **iPhone**, **iPad**, **iPod Touch**.

En 2015, **Apple** lanza su **Apple Watch**. Un reloj inteligente (**smartwatch**) que requiere un **iPhone** asociado para funcionar. Usa el sistema operativo **watchOS** basado en **iOS**.

Para desarrollar aplicaciones sobre **iOS**, las aplicaciones deben ser compiladas específicamente para este sistema operativo basado en la arquitectura **ARM**. Para ello puede utilizarse el kit de desarrollo **iPhone SDK**. El lenguaje de programación principal para este conjunto de herramientas es el **Objective-C**. Para poder utilizar este kit de desarrollo es necesario un ordenador **MAC** con un sistema operativo **MAC OS X Leopard** o superior. A mediados de 2011 ni el entorno **.NET** y ni **Adobe Flash** eran soportados por **iOS**, aunque **Adobe Flash** es utilizable a través de aplicaciones de terceros. Lo mismo sucede con **Java**, aunque se han producido algunos intentos de máquinas virtuales de **Java** (basadas en **JAVA ME**) para **iOS**, que rozan la legalidad a la que están sometidas las restrictivas licencias de **Apple**. Por supuesto, siempre habrá comunidades de usuarios y desarrolladores que intentarán saltarse estas restricciones (para más información al respecto puedes consultar el término **iOS Jailbreaking** en la web).

El **iOS SDK** se puede descargar gratuitamente, aunque es necesario registrarse en el [Programa del desarrollador de Apple](#)  para poder publicar el software creado, lo cual requiere la aprobación de la compañía **Apple** y un pago por ese servicio, que proporcionará al programador las claves firmadas que le permitirán publicar en la **App Store**.

Para saber más

Para saber más sobre **iOS** se pueden consultar los [artículos de la Wikipedia sobre este sistema operativo](#) , así como los documentos externos a los que hace referencia.

2.3.- Windows Phone.

Es el sistema operativo desarrollado por **Microsoft** para **smartPhones** y otros dispositivos móviles. Sustituye a su antecesor **Windows Mobile**. Las primeras versiones de sistemas operativos para dispositivos móviles desarrolladas por **Microsoft** eran conocidas como **Windows CE (Compact Embeded)**, y de hecho la familia de sistemas operativos de **Microsoft** para sistemas empujados en general (**smartPhones**, **PDA** y otros pequeños dispositivos) sigue llamándose **Windows CE**, así como el núcleo del sistema operativo.

Algunos de los primeros **Windows CE** podían encontrarse en dispositivos como las **PDA iPAQ** de **Compaq** (hoy día parte de HP), las cuales surgieron como competencia directa de las **PDA** de **Palm** (las **Palm Pilot**) durante la segunda mitad de la década de los noventa. Las **iPAQ** con **Windows CE** ofrecían un entorno más parecido al **Windows** de los ordenadores de escritorio frente al aspecto más sobrio de la interfaz de usuario de **Palm OS**.

Microsoft anunció en enero de 2015 que daría de baja a **Windows Phone**, para enfocarse en un único sistema más versátil denominado **Windows 10 Mobile**, disponible para todo tipo de plataformas (teléfonos inteligentes, tabletas y computadoras). Así, **Microsoft** lanzó su sistema operativo para ordenadores **Windows 10** en julio de 2015, y finalmente, marzo de 2016, lanzó su equivalente para dispositivos móviles **Windows 10 Mobile**. Actualmente es un sistema operativo discontinuado.





[ArtificialQg \(Pixabay License\)](#)

La última versión de **Windows Phone** fue la 8.1 (lanzada en marzo de 2015), desarrollada propiamente para **smartPhones** y **PDA**. Algunas de las versiones anteriores han sido: **Windows Phone 8**, **Windows Phone 7.x**, **Windows Mobile 6.x**, **Windows Mobile 5.x**, **Windows Mobile 2003** o **Pocket PC 2002**. Todas ellas basadas en la plataforma **Windows CE**.

Entre los fabricantes que incluían **Windows Phone** en sus dispositivos se encuentran **HTC** (series S y T), **Samsung** (serie Omnia), **LG** (Optimus 7 y Pacific) y **Sony Ericsson** (Xperia X7) y **Nokia**.

Para desarrollar aplicaciones sobre **Windows Phone** pueden utilizarse las tecnologías **Silverlight** y **XNA** de **Microsoft**, así como el entorno de desarrollo **Visual Studio**.

Para saber más

Para saber más sobre [Windows Phone](#)  y [Windows 10 Mobile](#)  se pueden consultar los artículos de la Wikipedia sobre este sistema operativo, así como los documentos externos a los que hace referencia.

3.- Plataformas de desarrollo y lenguajes de programación.

Caso práctico

María y Juan ya conocen los sistemas operativos con los que van a trabajar y para los que van a desarrollar las aplicaciones de los diferentes dispositivos móviles. A María le surge una duda...

-Para los desarrollos de software que estamos acostumbrados a realizar, siempre nos ayudamos de algún editor que nos simplifica el trabajo -reflexiona María-, entiendo que tendremos también alguna ayuda en el desarrollo de software para dispositivos móviles.

-Así es -confirma Ada-, en este caso tenemos plataformas de desarrollo.

-¡Claro! -exclama Juan-, y cada plataforma trabajará con un lenguaje de programación.

-En efecto. Os los voy a mostrar...



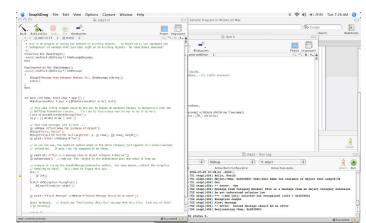
[Mikhail \(Pexels\)](#)

Para poder desarrollar aplicaciones para alguno de los anteriores sistemas operativos es necesario, disponer de alguna plataforma de desarrollo que permita generar código ejecutable sobre esos sistemas, o bien sobre alguna máquina virtual, o algún intérprete que esté instalado en el dispositivo y que sea soportado por el sistema operativo.

Dependiendo de la versatilidad de la plataforma de desarrollo, la aplicación podrá ser, más o menos portable a otros sistemas operativos y/o dispositivos. Por ejemplo, si realizamos programas en **Objective-C** utilizando el **SDK** de **Apple**, lo más probable es que nuestra aplicación pueda ser únicamente ejecutada en un dispositivo **Apple** (**iPhone** por ejemplo) sobre el que esté implantado **iOS**. Si, por el contrario, desarrollamos un programa en **Java** usando el **Java ME** de **Oracle**, y generamos una aplicación tipo **MIDlet**, ésta podrá ser ejecutada en cualquier dispositivo que disponga de una pequeña máquina virtual de **Java** (**KVM**). Se trata nuevamente de la misma problemática que podemos encontrar en el mundo de los ordenadores convencionales.

Entre las plataformas de desarrollo para móviles más populares se encuentran normalmente las que los propios autores de los sistemas operativos ofrecen para trabajar sobre su plataforma. De hecho ya hemos hablado algo sobre algunas de ellas al describir los sistemas operativos. Hacemos un repaso rápido de las más conocidas:

- ✓ **Android.** **Google** proporciona también de manera gratuita el **Android SDK** para programar aplicaciones en **Java** sobre su sistema operativo. El IDE más utilizado en el pasado fue **Eclipse** con el plugin **ADT** (Android Development Tools) y últimamente se impone **Android Studio**. También es posible programar en otros lenguajes como **C** o **C++** gracias al uso del **Android NDK** (Native Development Kit) que permite generar código nativo (native code), frente al código gestionado (managed code), que se ejecuta sobre una máquina virtual, como es el caso de **Java** o **C#**. Por otro lado, no se debe perder de vista el deliberado impulso que está haciendo **Google** para que **Android** sea programado en lenguaje **Kotlin**.



[Ron Bieber \(CC BY-NC-ND\)](#)

- ✓ **iOS.** El **SDK** también se puede descargar gratis, pero para poder comercializar el software a través de su tienda de aplicaciones hay que registrarse en el programa de desarrollo del **iPhone**, lo cual no es gratuito. El lenguaje de programación en este caso es **Objective-C**.






[LearnTek \(CC0\)](#)

✓ **Java ME:** Este caso sería una excepción respecto a los demás, pues no se trata de una serie de herramientas (bibliotecas, compiladores, IDEs, etc.) asociadas a un sistema operativo, sino de un subconjunto de la plataforma **Java** orientada para el desarrollo sobre dispositivos móviles. Se programaría en lenguaje **Java** y sería necesario que hubiera instalada una máquina virtual en el sistema operativo del dispositivo sobre el que se deseara ejecutar la aplicación desarrollada como sucede por ejemplo en **Symbian** (que se puede programar en modo nativo o bien con **Java ME**).

✓ **Windows Phone** (discontinuado). Para desarrollar aplicaciones sobre este sistema operativo pueden utilizarse las tecnologías **Silverlight**, **XNA** y **.NET Compact Framework** de **Microsoft**. Las herramientas **Visual Studio 2010 Express** y **Expression Blend** para **Windows Phone** son ofrecidas gratuitamente por **Microsoft**. El lenguaje más habitual para el desarrollo ha sido **C#**, aunque la idea es que se pueda utilizar también otros lenguajes de la plataforma **.NET** como **Visual Basic .NET**.

Para saber más

Para saber más sobre el lenguaje Kotlin:

- ✓ [Artículo de xatakandroid.com sobre kotlin](#) 
- ✓ [Artículo de apiumhub.com sobre kotlin](#) 
- ✓ [Artículo de paradigmadigital.com sobre kotlin](#) 

3.1.- Elección de una alternativa.

Dado que durante este Ciclo Formativo, es probable que hayas programado ya en **Java** (módulo de Programación), las alternativas más propicias son **Java ME** y **Android Studio** sobre **Android**. En cualquier caso, hemos de ser siempre conscientes de que no se trata más que de una de las posibles alternativas disponibles en el mercado y que más adelante (en futuros proyectos dentro de la empresa) es probable que quizá tengas que trabajar con otros lenguajes (por ejemplo **C#**, **C++**, **Python**, **Objective-C**, **Kotlin**,...) y para otras plataformas.

Lo importante es adquirir unos conocimientos generales sobre este nuevo tipo de entornos, sobre algunas bibliotecas concretas para estos dispositivos (por ejemplo los paquetes **javax** en el caso de **Java ME**) y comenzar a desarrollar algunos ejemplos de aplicaciones para ir desarrollando una nueva filosofía de trabajo.

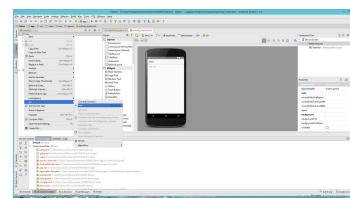
En nuestro caso Java puede ser una buena elección dado que es el lenguaje con el que aprendiste a programar en el módulo de Programación y por tanto no tendrás que aprender un nuevo lenguaje y podrías dedicarte de lleno al aprendizaje de las características específicas para el desarrollo de aplicaciones para dispositivos móviles. Por otro lado, un lenguaje multiplataforma como **Java** es también muy adecuado para un ciclo con un nombre como "Desarrollo de Aplicaciones Multiplataforma". Pero en cualquier caso no debes olvidar que se trata de una de las muchas opciones con las que te puedes encontrar en el mercado, y que estas alternativas irán apareciendo y desapareciendo constantemente en función del éxito que vayan obteniendo.

Respecto a la plataforma, debemos elegir entre aquellas que estén relacionadas con **Java** (**Java ME** y **Android Studio**). Concretamente, nos decantaremos por **Android Studio**, que es la que ha experimentado en los últimos años una progresión mayor.

Recapitulando, para facilitarte el aprendizaje vamos a trabajar con:

- ✓ **Plataforma Android.**
- ✓ **Lenguaje Java y XML.**
- ✓ **IDE Android Studio.**
- ✓ Sistema operativo que permita una máquina virtual **Java** para la emulación, aunque normalmente usaremos un emulador proporcionado por el propio **IDE**.
- ✓ Hardware (smartPhones, tablets, etc.) que soporte un sistema operativo del tipo anterior.

Como suele suceder en el mundo real, normalmente no podrás elegir una opción u otra en función de tus preferencias personales, sino más bien dependiendo cuáles sean las necesidades de tu cliente. Esa es una buena razón para intentar estar siempre al día.



[Jaumemonasterio \(CC BY-SA\)](#)

3.2.- La plataforma Android.

Los componentes principales del sistema operativo de **Android**:


- ✓ **Aplicaciones:** las aplicaciones base incluyen un cliente de correo electrónico, programa de SMS, calendario, mapas, navegador, contactos y otros. Todas las aplicaciones están escritas en lenguaje de programación Java.
- ✓ **Marco de trabajo de aplicaciones:** los desarrolladores tienen acceso completo a los mismos API del entorno de trabajo usados por las aplicaciones base. La arquitectura está diseñada para simplificar la reutilización de componentes; cualquier aplicación puede publicar sus capacidades y cualquier otra aplicación puede luego hacer uso de esas capacidades (sujeto a reglas de seguridad del framework). Este mismo mecanismo permite que los componentes sean reemplazados por el usuario.
- ✓ **Bibliotecas:** Android incluye un conjunto de bibliotecas de C/C++ usadas por varios componentes del sistema. Estas características se exponen a los desarrolladores a través del marco de trabajo de aplicaciones de Android. Algunas son: System C library (implementación biblioteca C estándar), bibliotecas de medios, bibliotecas de gráficos, 3D y SQLite, entre otras.
- ✓ **Runtime de Android:** Android incluye un set de bibliotecas base que proporcionan la mayor parte de las funciones disponibles en las bibliotecas base del lenguaje Java. Cada aplicación Android corre su propio proceso, con su propia instancia de la máquina virtual Dalvik. Dalvik ha sido escrito de forma que un dispositivo puede correr múltiples máquinas virtuales de forma eficiente. Dalvik ejecutaba hasta la versión 5.0 archivos en el formato de ejecutable Dalvik (.dex), el cual está optimizado para memoria mínima. La Máquina Virtual está basada en registros y corre clases compiladas por el compilador de Java que han sido transformadas al formato.dex por la herramienta incluida dx. Desde la versión 5.0 utiliza el ART, que compila totalmente al momento de instalación de la aplicación.
- ✓ **Núcleo Linux:** Android depende de Linux para los servicios base del sistema como seguridad, gestión de memoria, gestión de procesos, pila de red y modelo de controladores. El núcleo también actúa como una capa de abstracción entre el hardware y el resto de la pila de software.



[Torsten Dettlaff \(pexels.com\)](https://pexels.com)









3.3.- El entorno de ejecución.

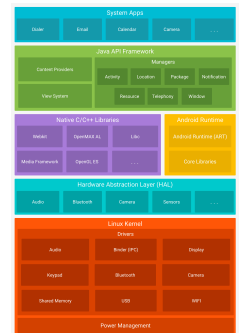
Debes conocer

A partir de ahora haremos muchas referencias durante todo el curso a la documentación oficial proporcionada por Google sobre el desarrollo de Android (developer.android.com ). De hecho se puede decir que es la principal guía para elaborar los contenidos relativos a Android de este curso.

Si consultamos la página oficial de desarrollo de Android (developer.android.com) podemos observar la siguiente estructura de capas o niveles.


Analicemos las diferentes capas empezando por abajo:

- A. **Kernel de Linux:** La base de la plataforma Android es el kernel de Linux. Por ejemplo, [el tiempo de ejecución de Android \(ART, acrónimo de Android Run Time\)](#)  se basa en el kernel de Linux para funcionalidades subyacentes, como la generación de subprocesos y la administración de memoria de bajo nivel. 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.
- B. **Capa de abstracción de hardware (HAL).** La [capa de abstracción de hardware \(HAL\)](#)  brinda interfaces estándares que exponen las capacidades de hardware del dispositivo al [framework de la Java API](#)  de nivel más alto. La HAL consiste en varios módulos de biblioteca y cada uno de estos implementa una interfaz para un tipo específico de componente de hardware, como el módulo de la [cámara](#)  o de [bluetooth](#) . Cuando el framework de una API realiza una llamada para acceder a hardware del dispositivo, el sistema Android carga el módulo de biblioteca para el componente de hardware en cuestión.
- C. **Tiempo de ejecución de Android (ART):** 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 de elementos no usados (GC) optimizada
 - ✓ Mejor compatibilidad con la depuración, como un generador de perfiles de muestras dedicado, excepciones de diagnóstico detalladas e informes de fallos, y la capacidad de establecer puntos de control para controlar campos específicos.



[Google Developers](#) ([Apache 2.0](#))

Antes de Android 5.0 (nivel de API 21), Dalvik era el tiempo de ejecución del sistema operativo. Por tanto ART es una evolución de Dalvik mejorando su rendimiento, en cuanto a velocidad de ejecución y tiempo que tarda en abrir una aplicación. 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 de tiempo 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 el framework de la Java API usa.

D. **Bibliotecas C/C++ nativas:** Muchos componentes y servicios centrales del sistema Android, como el ART y la HAL, se basan en código nativo que requiere bibliotecas nativas escritas en C y C++. La plataforma Android proporciona la API del framework de Java para exponer la funcionalidad de algunas de estas bibliotecas nativas a las apps. Por ejemplo, puedes acceder a la [guía](#) de [OpenGL ES](#) a través de la [Java OpenGL API](#) del framework de Android para agregar a tu app compatibilidad con los dibujos y la manipulación de gráficos 2D y 3D.

Si desarrollas una app que requiere C o C++, puedes usar el [NDK de Android](#) para acceder a algunas de estas [bibliotecas de plataformas nativas](#) directamente desde tu código nativo.

E. **Framework de la API de Java:** Todo el conjunto de funciones del SO Android está disponible mediante API escritas en el lenguaje Java. Estas API son los cimientos que necesitas para crear apps de Android simplificando la reutilización de componentes del sistema y servicios centrales y modulares, como los siguientes:

- ✓ Un [sistema de vista](#) enriquecido y extensible que puedes usar para compilar la IU de una app; se incluyen listas, cuadrículas, cuadros de texto, botones e incluso un navegador web integrable.
- ✓ Un [administrador de recursos](#) que te brinda acceso a recursos sin código, como strings localizadas, gráficos y archivos de diseño.
- ✓ Un [administrador de notificaciones](#) que permite que todas las apps muestren alertas personalizadas en la barra de estado.
- ✓ Un [administrador de actividad](#) que administra el ciclo de vida de las apps y proporciona una [pila de retroceso de navegación](#) común.
- ✓ [Proveedores de contenido](#) que permiten que las apps accedan a datos desde otras apps, como la app de Contactos, o compartan sus propios datos.

Los desarrolladores tienen acceso total a las mismas [API del framework](#) que usan las apps del sistema Android.

F. **Apps del sistema:** 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 elige 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.

3.4.- Maquinas virtuales.

Una máquina virtual es una aplicación software que emula el comportamiento de otra máquina (por ejemplo la emulación de un ordenador "virtual" dentro de un ordenador "real").




En el caso de una máquina virtual de **Java** (JVM – Java Virtual Machine), si recordamos lo visto en el módulo de **Programación**, se trata de un programa encargado de interpretar código intermedio (bytecode en el caso de Java) de los programas precompilados (en lenguaje Java) a código máquina ejecutable por el hardware (instrucciones máquina), realizar las llamadas al sistema operativo necesarias, velar por la seguridad e integridad del código ejecutado, etc.

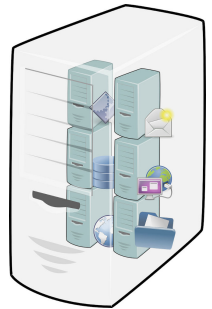
De esta manera, la **JVM** proporciona al programa compilado en **Java** independencia de la plataforma hardware y del sistema operativo que corra sobre él.

Las **JVM** clásicas para la plataforma **Java SE** son demasiado pesadas (necesidades de memoria, requerimientos de cómputo, pantalla, etc.) para dispositivos pequeños. Imagina por ejemplo la limitada pantalla de un teléfono móvil. Es fácil suponer que ese tipo de pantallas serían incapaces de soportar toda la funcionalidad proporcionada por la **AWT**, que es la principal interfaz gráfica de usuario que tiene **Java**. Es por esta razón que la plataforma **Java ME** ha propuesto otras máquinas virtuales bastante más ligeras.

Por otro lado, una única plataforma de Java podría no encajar adecuadamente con todos los modelos de dispositivos posibles. Es por ello por lo que **Java ME** introdujo los conceptos de configuración y de perfil.

Las dos máquinas virtuales disponibles en **Java ME** son la **KVM (K Virtual Machine)**, se le ha dado ese nombre porque sólo ocuparía unos pocos Kilobytes de memoria) y la **CVM (Compact Virtual Machine)**, algo más pesada.

De manera similar, para Android, [Dalvik](#) , que es la [máquina virtual](#)  utilizada originalmente por Android, y lleva a cabo la transformación de la aplicación en instrucciones de máquina, que luego son ejecutadas por el [entorno de ejecución](#)  nativo del dispositivo. En versiones más modernas de **Android**, **ART** ha reemplazado a **Dalvik**.



[OpenClipart-Vectors \(Pixabay License\)](#)