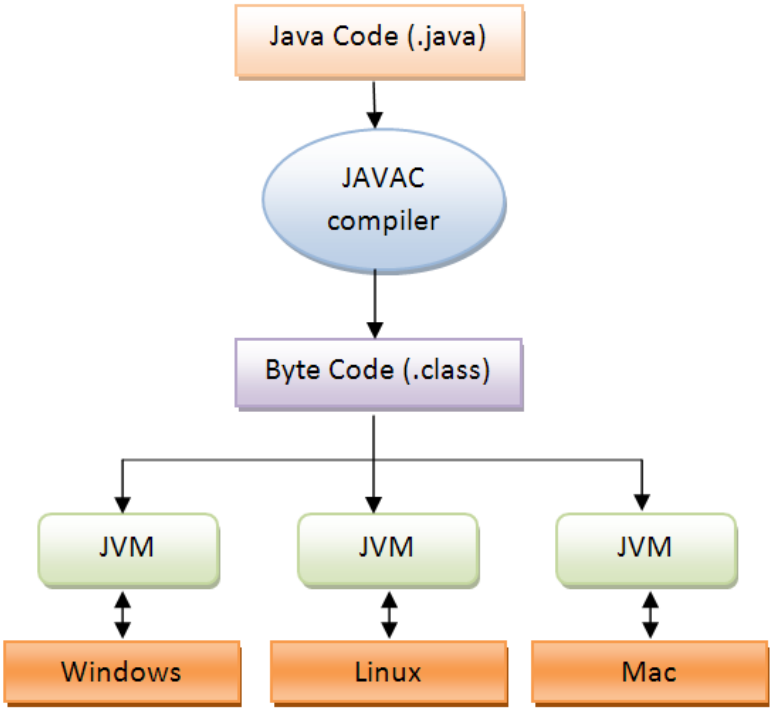
¿Qué es Java?

Java es un lenguaje de programación desarrollado por Sun Microsystems (Actualmente Oracle). Este lenguaje toma mucha de su sintaxis del lenguaje C y C++, pero eliminando muchas herramientas de bajo nivel que estos antiguos lenguajes poseen, permitiendo al programador, olvidarse de la administración de la memoria, ya que es el lenguaje quien la gestiona.

Los programas escritos en Java, son compilados a un "Bytecode" que luego es interpretado por una máquina virtual.

Máquina virtual : Programa que interpreta y ejecuta ciertas instrucciones pre definidas (bytecode) emulando un microprocesador. La máquina virtual de Java es llamada "JVM"

Debido a que existe una máquina virtual para cada plataforma (Windows, Linux, Mac, etc.) es posible independizar a un programa escrito en java (al bytecode) de la arquitectura donde se está ejecutando.



En la figura puede observarse el proceso de compilación y ejecución de un programa Java: El programa escrito en Java (archivos de extensión ".java") son compilados mediante el compilador java llamado "javac" el cual producirá comos salida archivos de extensión ".class" (este es el bytecode de java).

Es importante destacar cómo es posible ejecutar el mismo archivo ".class" en diferentes plataformas, ya que el archivo será interpretado por la JVM correspondiente, y ésta se encargará de "compatibilizar" la arquitectura de hardware para que el programa Java pueda correr sin problemas.

El conjunto de librerías y JVM, que permite la ejecución de programas java, es llamado **JRE**.

Cuando se quiere desarrollar programas en Java, no es suficiente con tener el paquete de librerías JRE, ya que éste solo nos permite ejecutar programas, pero no compilarlos. Para realizar esta tarea, se requiere de un paquete de herramientas llamado **JDK**, el cual posee el compilador "javac" junto con otras herramientas.

Entornos de desarrollo

Puede escribirse el código Java en cualquier editor de texto y realizar la compilación en forma manual con los programas proporcionados por la JDK. Sin embargo, Cuando el programa comienza a crecer es necesario tener un entorno de desarrollo que nos permita trabajar más cómodamente, los más populares son "NetBeans" y "Eclipse", las herramientas que nos permiten programar Java para el sistema operativo Android, están diseñadas para trabajar con el entorno de desarrollo "Android Studio" basado en IntelliJ.

Detalles sobre la JVM

Además de la máquina virtual de Oracle (Creadores del lenguaje Java) existen alternativas Open Source (ya que la JVM de Oracle no es de código abierto), como por ejemplo "Apache Harmony" y "OpenJDK".

Como se mencionó anteriormente, existe una JVM para cada plataforma, y esto logra independizar el código java y permitir que sea *portable*.

También es importante destacar que solo una JVM ejecutará todas las aplicaciones Java, es decir, no existirá una instancia de la JVM para cada aplicación Java que ejecutemos.

Manejo de memoria: Garbage Collector.

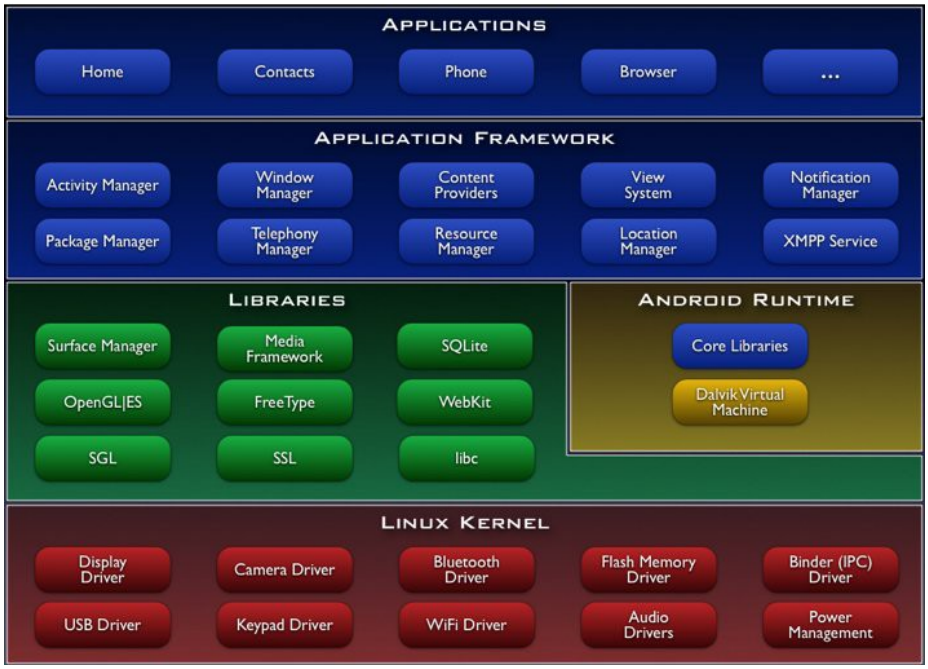
La JVM permite al programador olvidarse de la administración de la memoria dinámica, en otras palabras, el programador podrá crear objetos y nunca tendrá que preocuparse por liberar la memoria que estos ocupan cuando se dejen de utilizar. Esto es posible gracias a un módulo de la maquina virtual llamado "Garbage Collector" el cual busca en memoria los objetos a los cuales no existe ninguna referencia, y libera la memoria que éstos ocupan.

¿Qué es Android?

Android es un sistema operativo desarrollado por Google para Dispositivos Móviles el cual posee una gran comunidad de desarrolladores. Fue desarrollado inicialmente por Android Inc., una firma comprada por Google en 2005. Este sistema operativo esta basado en Linux y posee una Java Core Library basada en Apache Harmony (una de las JREs open source que mencionamos anteriormente).

A pesar de utilizar la librerías de Apache Harmony, la JVM es diferente, y es llamada "Dalvik Virtual Machine" (DVM). Debido a que esta maquina virtual es diferente, el bytecode estandard de Java no peude ser ejecutado directamente por la DVM, sino que debe ser convertido a archivos ".dex".

Arquitectura del sistema operativo



Como puede observarse en la figura, el sistema se compone en 4 capas:

- **Kernel de Linux :** El kernel es el núcleo del sistema operativo, posee todas las librerías de bajo nivel que permiten la interacción con el hardware, las librerías para el manejo del display, cámara, módem Wifi, GSM, Audio y memorias están en esta capa.

- **DVM y Libraries :** En esta capa se encuentra la máquina virtual de Dalvik, junto con las bibliotecas que utiliza (las que provienen de Apache Harmony) y las bibliotecas de nivel "medio" programadas en C y C++ que proveen un nivel de abstracción con respecto a las bibliotecas de bajo nivel que se encuentran en el kernel.

- **Application Framework :** Son bibliotecas programadas en lenguaje Java las cuales proveen al desarrollador de aplicaciones Android un fácil acceso a el hardware del dispositivo, evitando al programador tener que utilizar las bibliotecas de nivel medio o bajo. Estas bibliotecas serán las que utilizaremos a lo largo del curso para desarrollar aplicaciones.

- **Aplicaciones :** La última capa del sistema operativo son las aplicaciones, aquí se encontrarán los programas que desarrollaremos, y como puede observarse, todo con lo que interactuamos en un dispositivo móvil, es una aplicación: Cuando marcamos un numero de teléfono, estamos utilizando una aplicación que se encarga de interactuar con el hardware del módem telefónico, cuando revisamos nuestra agenda de contactos, estamos utilizando una aplicación que lee la información de una memoria y nos la muestra en pantalla; inclusive la lista de iconos de las aplicaciones que tenemos instaladas en el "home" de nuestro teléfono, también es una aplicación.

Detalles sobre la DVM

La maquina virtual que utiliza Android esta optimizada para trabajar con sistemas embebidos, esto quiere decir es es rápida en procesadores chicos y posee pocos requerimientos de memoria.

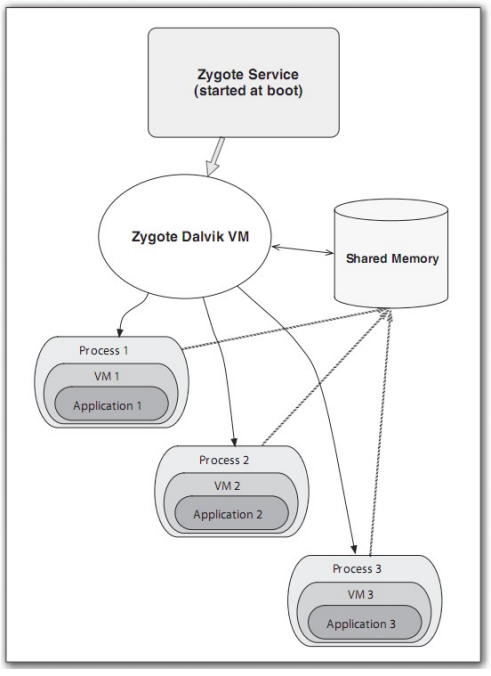
La DVM es energéticamente mas eficiente: El bytecode para DVM es más compacto, esto implica menos instrucciones, y eso se traduce en menos ciclos de CPU, que conlleva un menor consumo de batería.

Como se mencionó anteriormente, no puede ejecutarse código de java standard sobre esta maquina virtual directamente, el proceso que se lleva a cabo es el siguiente:

- Se escribe el programa en Java
- Se compila a bytecode de Java estandard para aprovechar todas las optimizaciones que éste posee.
- Se "traduce" el bytecode de Java (.class) al bytecode para la DVM (.dex).

Esto también trae como ventaja el poder reutilizar archivos .class ya compilados.

Inicialización de una aplicación



A diferencia con la JVM, existe una DMV por cada aplicación que se ejecuta.

Existe un servicio llamado Zygote que precarga las Core Libraries en una memoria compartida para que todos los programas tengan acceso. Se gana velocidad no copiando las Core Libraries por cada aplicación.

Nueva máquina virtual: ART

A partir de la versión de Android KitKat, se desarrolló una nueva metodología de ejecución de las aplicaciones para una mejor performance: se modificó la DVM y se transformó en **ART** (**A**ndroid **R**un**T**ime).

La DVM utiliza la tecnología **JIT** (**J**ust **I**n **T**ime Compilation) mediante la cual se compilan porciones de código a código nativo, en tiempo de ejecución, mejorando la performance a medida que la aplicación de ejecuta.
Las ventajas de esta nueva máquina virtual es la implementación de una compilación **AOT** (**A**head **O**f **T**ime), mediante la cual la aplicación se compila en el momento de la instalación, en lugar de compilarse en el momento de su ejecución, mejorando el rendimiento del CPU y el uso de batería.