I. Introducción a la programación orientada a objetos y al lenguaje JAVA

Programación Orientada a Objetos

La Programación Orientada a Objetos es un paradigma de programación diferente a la Programación Estructurada o a la misma Programación Procedural al que la mayoría de los programadores están acostumbrados a utilizar.

En la Programación Estructurada y la Programación Procedural lo mas importante es el procedimiento que se esta desarrollando a través de un lenguaje de programación.

En Programación Orientada a Objetos, se hace un gran énfasis a los datos, y todo gira alrededor de ellos.

Cada vez que alguien quiere hacer una aplicación, debe de pensar en los elementos (datos) que va a utilizar para programar, y la relación que existe entre estos datos, en su forma de interactuar entre si.

Cuando queremos hacer una aplicación de alumnos que estan inscritos en algunas materias con ciertos profesores, entonces debemos de pensar en los diferentes datos a manejar: Alumnos, Materias, Profesores, etc.

Pensar en estos elementos y pensar en todo lo que esta alrededor de ellos para poder tipificarlos en los elementos que definen la Programación Orientada a Objetos.

El mundo esta lleno de objetos y estos objetos tienen ciertas características que los hacen únicos y esas características se derivan de atributos que agrupadas representan a una clase que compone a los objetos definidos.

Java es un lenguaje que nos ayuda a entender mucho mejor el paradigma Orientado a Objetos de una manera mas sencilla y natural.

Antecedentes del Lenguaje Java

Java se creó como parte de un proyecto de investigación para el desarrollo de software avanzado para una amplia variedad de dispositivos de red y sistemas embebidos. La meta era diseñar una plataforma operativa sencilla, segura, portable, distribuida y de tiempo real.

Cuando se inició el proyecto, C++ era el lenguaje del momento. Pero a lo largo del tiempo, las dificultades encontradas con C++ crecieron hasta el punto en que se pensó que los problemas podrían resolverse mejor creando una plataforma de lenguaje completamente nueva.

Se hizo uso de la arquitectura y diseño de una amplia variedad de lenguajes como Eiffel, SmallTalk, Objetive C y Cedar/Mesa. El resultado es un lenguaje que se ha mostrado ideal para desarrollar aplicaciones de usuario final seguras, distribuidas y basadas en red en un

amplio rango de entornos desde los dispositivos de red embebidos hasta su uso para soluciones en Internet.

Características en el Diseño de Java

- **Sencillo, orientado a objetos y familiar**: Sencillo, para que no requiera grandes esfuerzos de entrenamiento para los desarrolladores. Orientado a objetos, porque la tecnología de objetos se considera madura y es el enfoque más adecuado para las necesidades de los sistemas distribuidos y/o cliente/servidor. Familiar, porque aunque se rechazó C++, se mantuvo Java lo más parecido posible a C++, eliminando sus complejidades innecesarias, para facilitar la migración al nuevo lenguaje.

- **Robusto y seguro**: Robusto, simplificando la administración de memoria y eliminando las complejidades del uso de apuntadores y aritmética de apuntadores del C. Seguro para que pueda operar en un entorno de red.

- **Independiente de la arquitertura y portable**: Java está diseñado para soportar aplicaciones que serán instaladas en un entorno de red heterogéneo, con hardware y sistemas operativos diversos. Para hacer esto posible el compilador Java genera un código llamado 'bytecodes' o comúnmente conocido como código byte, un formato de código independiente de la plataforma diseñado para transportar código eficientemente a través de múltiples plataformas de hardware y software. Es además portable en el sentido de que es rigurosamente el mismo lenguaje en todas las plataformas. El 'bytecode' es traducido a código máquina y ejecutado por la Java Virtual Machine, que es la implementación Java para cada plataforma hardware-software concreta.

- **Alto rendimiento**: A pesar de ser interpretado, Java tiene en cuenta el rendimiento, y particularmente en las últimas versiones dispone de diversas herramientas para su optimización. Cuando se necesitan capacidades de proceso intensivas, pueden usarse llamadas a código nativo.

- **Interpretado, multi-hilo y dinámico**: El intérprete Java puede ejecutar codigo byte en cualquier máquina que disponga de una Máquina Virtual Java (JVM). Además Java incorpora capacidades avanzadas de ejecución multi-hilo (ejecución simultánea de más de un flujo de programa) y proporciona mecanismos de carga dinámica de clases en tiempo de ejecución.

Características del Lenguaje

- ✓ Lenguaje de propósito general.
- ✓ Lenguaje Orientado a Objetos.
- ✓ Sintaxis inspirada en la de C/C++.
- ✓ Lenguaje multiplataforma: Los programas Java se ejecutan sin variación (sin recompilar) en cualquier plataforma soportada (Windows, UNIX, Mac, etc.)
- ✓ Lenguaje interpretado: El intérprete a código máquina (dependiente de la plataforma) se llama Java Virtual Machine (JVM). El compilador produce un código intermedio independiente del sistema denominado *bytecode* ó código byte.
- ✓ Lenguaje gratuito: Creado por SUN Microsystems, que distribuye gratuitamente el producto base, denominado JDK (Java Development Toolkit) o actualmente J2SE (Java 2 Standard Edition).
- ✓ API distribuida con el J2SE muy amplia. Código fuente de la API disponible.

Facilidades del J2SE (Java 2 Estándar Edition)

- ✓ Herramientas para generar programas Java. Compilador, depurador, herramienta para documentación, etc.
- ✓ La JVM, necesaria para ejecutar programas Java.
- ✓ La API de Java (jerarquía de clases).
- ✓ Código fuente de la API (Opcional).
- ✓ Documentación.

Java Runtime Environment (JRE)

JRE es el entorno mínimo para ejecutar programas Java 2. Incluye la JVM y la API. Está incluida en el J2SE aunque puede descargarse e instalarse separadamente. En aquellos sistemas donde se vayan a ejecutar programas Java, pero no compilarlos, el JRE es suficiente.

El JRE incluye el Java Plug-in, que es el 'añadido' que necesitan lo navegadores (Explorer o Netscape) para poder ejecutar programas Java 2. Es decir que instalando el JRE se tiene soporte completo Java 2, tanto para aplicaciones normales (denominadas 'standalone') como para Applets (programas Java que se ejecutan en una página Web, cuando esta es accedida desde un navegador).