Primera Unidad: Conceptos Básicos

La programación orientada a objetos modela el mundo real, cualquier cosa del mundo puede ser modelada como un objeto. Así una circunferencia es un objeto, un automóvil es un objeto, una ventana es un objeto, un libro es un objeto e incluso un préstamo o una tarjeta de crédito son objetos. Un programa Java se denomina orientado a objetos debido a que la programación en Java se centra en la creación, manipulación y construcción de objetos.

Un objeto tiene ***propiedades*** (un estado) y un comportamiento. Las propiedades o el estado se detienen utilizando datos y el comportamiento se define utilizando métodos. Los objetos se definen utilizando clases en Java. Una clase es similar a una plantilla para construir objetos. Con la excepción de los tipos de datos primitivos, todo en Java es un objeto. En Java, al contrario de lo que sucede en C++, no hay funciones globales: todas las funciones se invocan a través de un objeto.

Por ejemplo, se puede definir un objeto cuadrado mediante una clase cuadrado (Fig. l.2), con un lado (propiedad) y calcularSuperficie como el método que encuentre o calcule la superficie del cuadrado

Un objeto es una realización concreta de una descripción de una clase. El proceso de creación de objetos se denomina instanciación (crear instancias) de una clase.

AI instanciar una clase, se crean objetos. Así, es posible crear un objeto cuadrado instanciando la clase con un lado determinado, por ejemplo se puede crear un cuadrado de lado 10 y otro cuadrado de lado 25. Se puede encontrar el área de los respectivos cuadrados usando el método calcularSuperficie.

Un programa consta de una o más clases que se disponen en una jerarquía en modo árbol, de modo que una clase hija puede heredar propiedades y comportamientos de su clase padre (ascendente). Java con un conjunto de clases predefinidas, agrupadas en paquetes que se pueden utilizar en los programas.

La programación orientada a objetos proporciona mayor flexibilidad, modularidad y reusabilidad. En la actualidad está ya muy implantado este tipo de programación y Java se es ahora uno de los lenguajes más usados de propósito general.

El paradigma orientado a objetos se enfoca a las características de comportamiento y estructura de las entidades como unidades completas, lo que nos permite diseñar software de manera modular y con un alto manejo de estructuras de datos complejas. El paradigma orientado a objetos se apoya en los siguientes conceptos:

- La **Abstracción** (de datos) involucra la formulación de un concepto (clase) poniendo atención en las similitudes y diferencias entre las entidades de un conjunto, para extraer las características esenciales que lo distingan y evitar las características no relevantes. Y así, se establece una sola representación del concepto que tenga esas características pertinentes.

  Es la capacidad de crear tipos de datos definidos por el usuario extrayendo las propiedades esenciales de un concepto, sin preocuparse de los detalles exactos de la

implementación. Algunos simplemente lo definen como la capacidad de enfocarse en lo esencial.

- La **Encapsulación** asegura que los usuarios de un objeto no alteren de manera inesperada el estado interno de un objeto; solo sus métodos internos están autorizados para acceder a ellos. Cada objeto expone una interfaz pública que especifica como otros objetos podrían interactuar con él.

  La abstracción y la encapsulación están representadas por la clase. La clase es una abstracción, porque define los atributos y métodos de un determinado conjunto de objetos con características comunes, y es una encapsulación por que constituye una caja negra que encierra tanto los datos que almacena como los métodos que permiten manipularlos.

- La **Herencia** Es la capacidad de definir una nueva clase a partir de otra clase. De esta forma, la reutilización del código está garantizada. Las clases están clasificadas en una jerarquía estricta, donde la clase padre es conocida como superclase y la clase hijo como clase derivada. Esto significa que la clase derivada dispone de todos los atributos y métodos de su superclase.

- El **Polimorfismo** es la propiedad que permite a una operación tener diferente comportamiento en diferentes objetos. Es decir, diferentes objetos reaccionan de manera diferente al mismo mensaje (de modo que un mismo método pueda tener múltiples implementaciones).

Estos conceptos son los pilares de la POO que permiten facilitar la comunicación, incrementar la productividad y la consistencia, facilitar la modificación de los programas y reducir la complejidad en esfuerzos de resolución de problemas.

1. El entorno de desarrollo de Java

1.1.El compilador de Java

El significado de Java tal y como se le conoce en la actualidad es el de un lenguaje de programación y un entorno para ejecución de programas escritos en el lenguaje Java. AI contrario que los compiladores tradicionales, que convierten el código fuente en instrucciones a nivel de máquina, el compilador Java traduce el código fuente Java en instrucciones que son interpretadas por la Máquina Virtual Java (JVM, Java Virtual Machine). A diferencia de los lenguajes C y C++ en los que está inspirado, Java es un lenguaje interpretado.

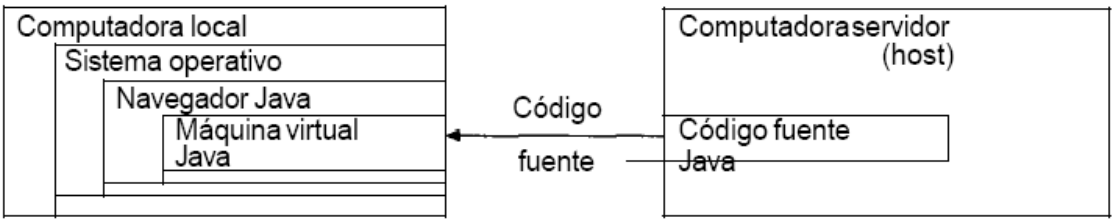
Aunque hoy en día Java también es un el lenguaje de programación para Internet, Java no comenzó como proyecto Internet y por esta circunstancia es idóneo para tareas de programación de propósito general y, de hecho, muchas de las herramientas Java están escritas en Java.

Java como lenguaje de Internet

Java es un lenguaje para programar en Internet que trata de resolver dos problemas claves con el contenido de Internet:

1. En la actualidad, el contenido de la WWW es pasivo y estático.
2. La entrega (Deliverry) del contenido WWW es dependiente de la configuración de cada navegador Web de usuario.

En el mundo de la Web, Java es una tecnología facilitadora que permite a los desarrolladores crear páginas Web que se entregarán de modo consistente a todos los usuarios con un navegador habilitado para Java y con independencia de la plataforma hardware y el sistema operativo que se esté utilizando. Dado que el código fuente se interpreta, si existe un intérprete Java para una plataforma específica hardware o sistema operativo, se pueden escribir programas con el conocimiento de que serán útiles en esa plataforma.



La Figura anterior muestra cómo el código fuente Java se transfiere en Internet. En la computadora servidor *(host)* se almacena el código fuente. Cuando un usuario de una computadora local se conecta con el servidor a través de Internet mediante un navegador habilitado para Java, el código fuente se transfiere de la computadora servidor a la computadora local.

Java como lenguaje de propósito general

A medida que Java se populariza en desarrollos de Internet, gana también como lenguaje de propósito general. Java es totalmente portable a gran variedad de plataformas hardware y sistemas operativos.

Java tiene muchos conceptos de sintaxis de C y C++, especialmente de C++, del que es un lenguaje derivado. Añade a C++ propiedades de gestión automática de memoria y soporte a nivel de lenguaje para aplicaciones multihilo. Por otra parte, Java, en principio a nivel medio, es más fácil de aprender y más fácil de utilizar que C++ ya que las características más complejas de C++ han sido eliminadas de Java: herencia múltiple, punteros (apuntadores) y sentencia goto entre otras.

Las implementaciones de la Máquina Virtual Java pueden ser muy eficaces y eso hace posible que los programas Java se ejecuten tan rápidamente como los programas C++. Esta característica clave de Java, unida a sus fortalezas como lenguaje de Internet, lo hacen muy adecuado para desarrollos en sistemas cliente/servidor, soporte masivo de los sistemas informáticos de la mayoría de las empresas y organizaciones.

Las propiedades que se verán más adelante hacen a Java doblemente idóneo para desarrollos cliente/servidor y para desarrollos de Internet.

Java ha conseguido una enorme popularidad. Su rápida difusión e implantación en el mundo de la programación en Internet y fuera de línea (offline) ha sido posible gracias a sus importantes características. Los creadores de Java escribieron un artículo, ya clásico, en el que definían al lenguaje como sencillo, orientado a objetos, distribuido, interpretado, robusto, seguro, arquitectura neutra, alto rendimiento, multihilo y dinámico

1.2.Arquitectura de la tecnología Java (JRE, JVM, GC)

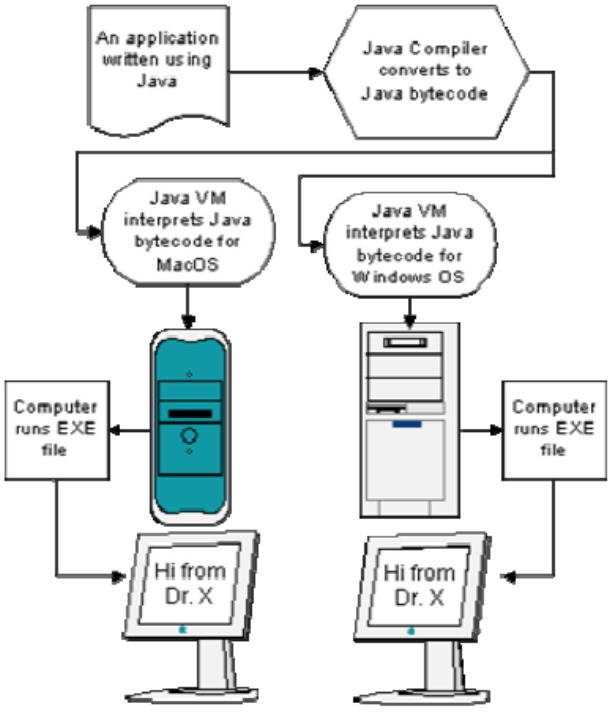
La tecnología de java se sustenta en 3 mecanismos principales que le permiten adquirir todos los beneficios de un sistema multiplataforma, estos son:

JRE

El ambiente en tiempo de ejecución, es el software responsable de permitir al usuario final ejecutar las aplicaciones java, posee una colección o librería estándar de clases que son indispensables para el funcionamiento de los programas java, además de incluir una maquina virtual para el lenguaje java.

JVM

La maquina virtual de java (Java VM) provee una especificación de plataforma la cual le permite ejecutar cualquier programa compilado en java (bytecode). De esta forma el compilador de java toma cualquier código hecho en java y genera un archivo precompilado llamado bytecode que es un conjunto de instrucciones que puede ser interpretado por cualquier maquina virtual de java.



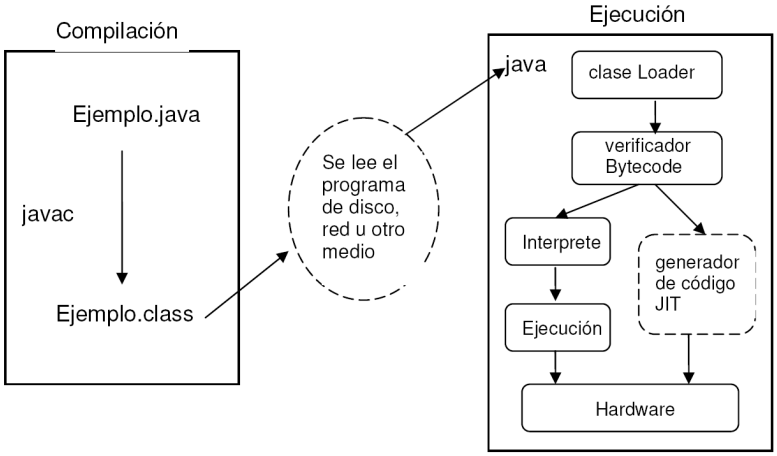
Garbage Collector

Muchos lenguajes de programación permiten alojar memoria de manera dinámica en tiempo de ejecución. El proceso para alojar memoria varía de acuerdo a las características del

lenguaje y en ciertas ocasiones los bloques de memoria reservados nunca son liberados, principalmente porque esta tarea se deja a cargo del programador (como en C/C++).

Java provee un mecanismo de recolección de bloques de memoria inútiles y los remueve de manera automática. Durante la ejecución de los programas en la maquina virtual, el recolector de basura comienza a verificar si existen bloques de memoria (objetos) que han dejado de ser usados o referenciados de tal manera que los marca para que en el próximo ciclo de verificación sean eliminados y se recupere espacio de memoria.

A continuación se muestra el esquema en tiempo de ejecución de la tecnología java:



El código fuente (Ejemplo.java) es compilado y se genera un archivo convertido a bytecode (Ejemplo.class), éste es el archivo que será cargado y ejecutado en alguna maquina objeto.

Al tiempo de ejecución, el código en bytecode es leído, verificado y puesto para su ejecución en el intérprete. El interprete tiene dos funciones ejecutar bytecodes y hacer las llamadas apropiadas al hardware. En algunos ambientes de ejecución una porción de la verificación de bytecode es compilada en código nativo de máquina y ejecutado directamente en la plataforma de hardware, por lo que la velocidad de interpretación-ejecución se acelera.

El lenguaje de programación java requiere de una infraestructura necesaria para su desarrollo y ejecución, inicialmente veremos como están constituidas sus plataformas de ejecución y desarrollo, posteriormente mencionaremos algunas aplicaciones que se pueden emplear como ambiente ambientes de desarrollo para facilitar la programación de aplicaciones.

JRE y JSDK

JRE

Es un paquete que contiene todo el software necesario para correr una aplicación desarrollada en Java, dentro del paquete esta contenida la librería de clases, que son programas generados en java (bytecode) y que están expresamente construidos para servir de apoyo para la construcción de aplicaciones por parte del usuario, además se cuenta con:

- Librerías de integración que permiten construir aplicaciones de comunicación externa.
- Librerías de interfase grafica de usuario AWT/Swing.
- Librerías para el manejo de multimedia.
- Una maquina virtual de java adecuada a la plataforma que permite ejecutar las aplicaciones

JSDK

Es el conjunto de herramientas para el desarrollo de aplicaciones en Java, esta compuesto por las siguientes herramientas de programación:

- **javac** – Compilador de java que se encarga de convertir código fuente a bytecode de java.
- **jar** – Programa que se encarga de manipular librerías de clases ya compiladas y empaquetadas.
- **javadoc** – Programa que se encarga de extraer documentación automáticamente partiendo de los comentarios en el código fuente.
- **jdb** – Depurador de programas.

Además de las herramientas adicionales para desarrolladores, se agrega el paquete JRE que contiene todas las librerías de clase y la maquina virtual en la que se producirán y ejecutaran las aplicaciones.

Las herramientas de desarrollo están disponibles en 3 posibles versiones:

- **Edición Micro**: Se emplea en ambientes con recursos limitados.
- **Edición Estándar**: Es la edición comúnmente empleada por la mayoría de los programadores.
- **Edición Empresarial:** Es una edición que posee una colección muy amplia de recursos adicionales para transacciones en Internet, seguridad y acceso a recursos remotos.

Tarea

- Investigar versiones de Java más recientes Java SE, Java EE y su evolución.

1.3.Ambientes de desarrollo

Los ambientes de desarrollo son aplicaciones que nos permiten escribir en un lenguaje de programación de manera sencilla, ya que se puede verificar gradualmente el producto que estemos desarrollando. De esta forma el ambiente de desarrollo se encuentra totalmente independiente de la plataforma de Java que se emplee, es decir que se puede emplear cualquier versión del kit de desarrollo de software (JSDK) de Sun.

Existen varios ambientes de desarrollo en el mercado, los hay desde los libres que no tienen algún costo, hasta las herramientas más sofisticadas y caras del mercado. El lenguaje Java (plataforma Java) y el ambiente de desarrollo integral no son una sola aplicación, en tal caso deberán ser instaladas de manera independiente.

Se recomienda inicialmente instalar la plataforma de Java (JSDK) y posteriormente el ambiente de desarrollo que más se ajuste a sus necesidades. En la red existen disponibles varios ambientes para el desarrollo de software como:

- **NetBeans:** una herramienta creada por una filial de Sun Microsystems, es una buena herramienta de diseño aunque algo lenta si se cuenta con poca memoria principal. Permite el diseño RAID (diseño rápido de interfase)
- **Gel:** una herramienta de distribución libre bastante flexible en uso de recursos y fácil de emplear debido a su poca complejidad, permite la opción de auto-completar texto.
- **JCreator:** es una herramienta amigable y fácil de emplear, cuenta con una versión libre bastante austera y una distribución comercial bastante competitiva.

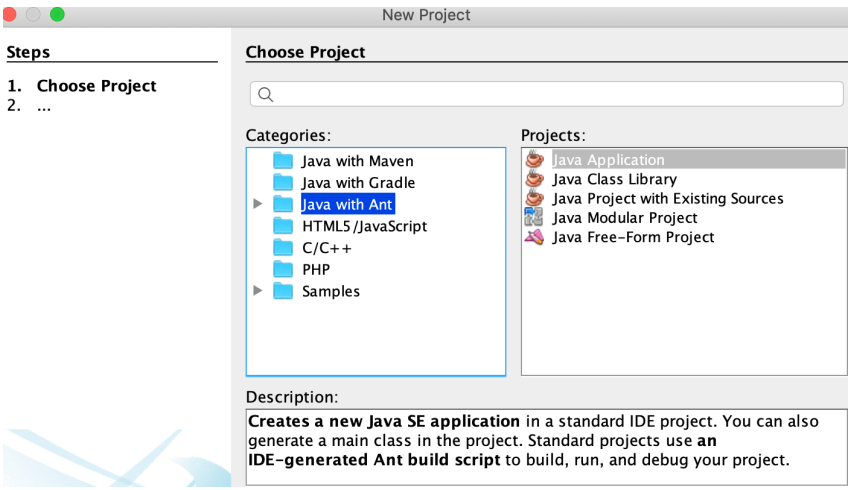
Es importante recordar que en el mercado se encuentran disponibles varias distribuciones libres para ambientes de desarrollo integral en Java, así que se deja al criterio del alumno el uso de cualquiera de ellas.

También se encuentra disponible una plataforma de diseño integral para Java, conocida como Eclipse, desarrollada por IBM. Eclipse es algo más que un simple ambiente de desarrollo integral (JDT), ya que cuenta con su propio kit de desarrollo para compilación (ECJ). Desde un principio se considero a Eclipse como una distribución libre para Java, por lo que su popularidad no se ha disminuido ante su mas claro competidor Sun Microsystems. Desafortunadamente por cuestiones de licencias Eclipse requiere la instalación de JRE para mantener la portabilidad.

https://netbeans.apache.org/download/nb126/nb126.html

Actividad: comente en el foro su experiencia en la instalación:

- Pasos realizados
- Problemas / Solución
- Captura de pantalla de Netbeans ejecutándose



2. Fundamentos del lenguaje

Los nombres de Java son sensibles a las letras mayúsculas y minúsculas. Por ejemplo, las variables mas*a*, Masa y MASA son consideradas variables diferentes. Las reglas del lenguaje respecto a los nombres de variables son muy amplias y permiten mucha libertad al programador.

2.1.Convenciones de escritura en Java

A continuación se muestran solo algunas convenciones de escritura (codificación) del lenguaje java:

Package.- Los paquetes son colecciones de clases distribuidas (organizadas) en directorios y por lo general se encuentran relacionadas entre sí. El nombre de los paquetes se escriben en minúsculas.

```
package proyecto.objetos
```

Clases.- Los nombres de las clases deben ser sustantivos (desde un punto de vista meramente didáctico se le define como el tipo de palabra que significa persona, animal o cosa concreta o abstracta), en algunos casos mezclados, con la primera letra de cada palabra en mayúscula.

```
class EjemploUno
```

Método.- Los nombres de los métodos deben ser verbos, con palabras mezcladas, con la primera letra en minúscula. En caso de emplear mas palabras se le recomienda hacer la primera letra mayúscula.

```
calculaPromedio()
```

Variables.- Todas las variables pueden ser mezclas de palabras procurando emplear la letra minúscula para la primera y empleando las letras mayúsculas para separar al resto de las palabras.

```
apellidoPaterno
```

Procure limitar el empleo de variables simples (a, b, …, x) para el manejo de bucles de control.

Constantes.- se recomienda el uso de las constantes con letras mayúsculas, separando las palabras con guión inferior.

```
TOPE_MAXIMO
```

https://www.oracle.com/technetwork/java/codeconventions-150003.pdf

Tarea: Realizar una síntesis del documento Java Code Conventions, en el que se especifican el estándar de codificación en Java.