DISEÑO ORIENTADO A OBJETOS

Prof: Esp. Ing. Agustín Fernandez

Introducción a Java

- Java es un lenguaje de programación orientado a objetos, fuertemente tipado. Esto quiere decir que todas las variables y objetos van a tener un tipo dado por una clase o primitiva.
- Es intrínsecamente orientado a objetos.
- Funciona perfectamente en red.
- Tiene una gran funcionalidad gracias a una gran cantidad de librerías y framework.
- **NO** tiene punteros manejables por el programador, aunque los maneja interna y transparentemente.
- El manejo de la memoria no es un problema, la gestiona el propio lenguaje y no el programador.
- Incorpora Multi-Threading.
- Etc, etc, etc...

Introducción a Java

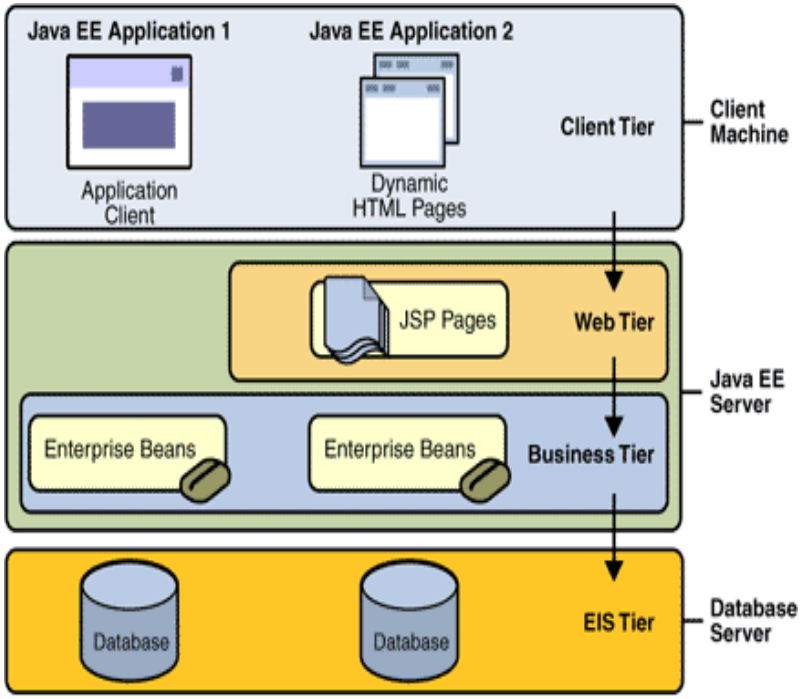
Existen varias APIs de Java:

- **J2SE**: Java 2 Standard Edition, se utiliza para aplicaciones de escritorio principalmente.
- **J2EE (actualmente Jakarta EE)**: ~~Java 2 Enterprise Edition, se utiliza para desarrollo de sistemas web, principalmente backend. Incluye a J2SE.~~ Es considerado el estándar de la industria para desarrollar aplicaciones empresariales Java portables, robustas, escalables y seguras en el lado del servidor (server-side monolitico). Proporciona APIs para servicios web, modelo de componetes, gestión y comunicación. Java EE proporciona una arquitectura multi-capa. La capa cliente puede estar constituida por aplicaciones Java de escritorio o navegadores HTML o de cualquier otro tipo. Las capas proporcionadas por Java EE propiamente dicha son las capas Web (mediante las tecnologías Servlets, JSP y JSF) y las capas de Negocio (mediante tecnologías como EJB, JMS o Web Services). Por último, estas capas se comunican con una capa de datos (base de datos o aplicaciones y sistemas legacy).

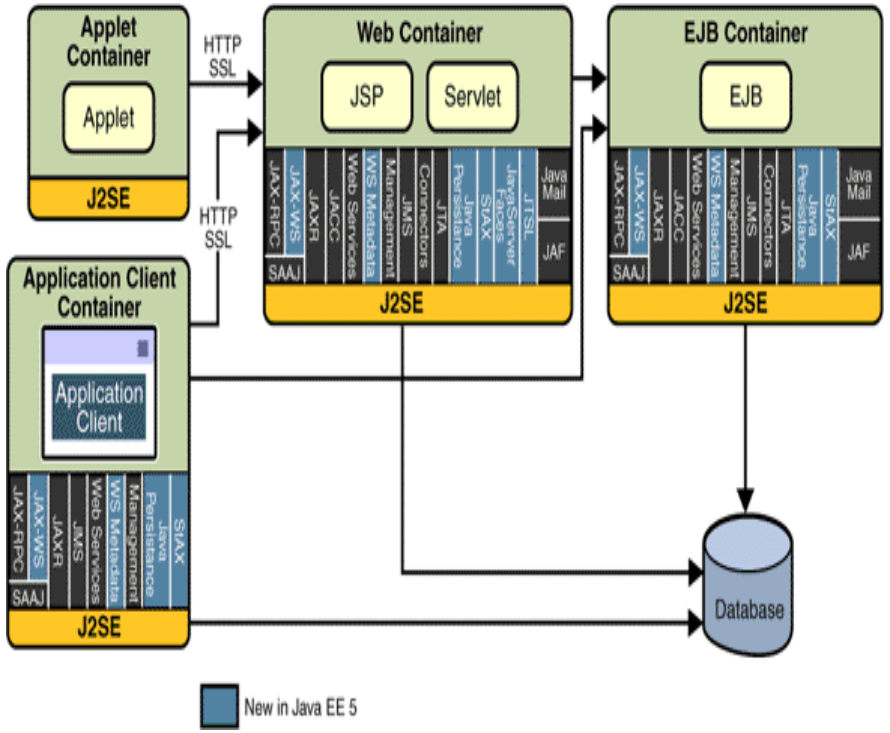
Introducción a Java

- **J2EE (actualmente Jakarta EE)**:

Multi tier

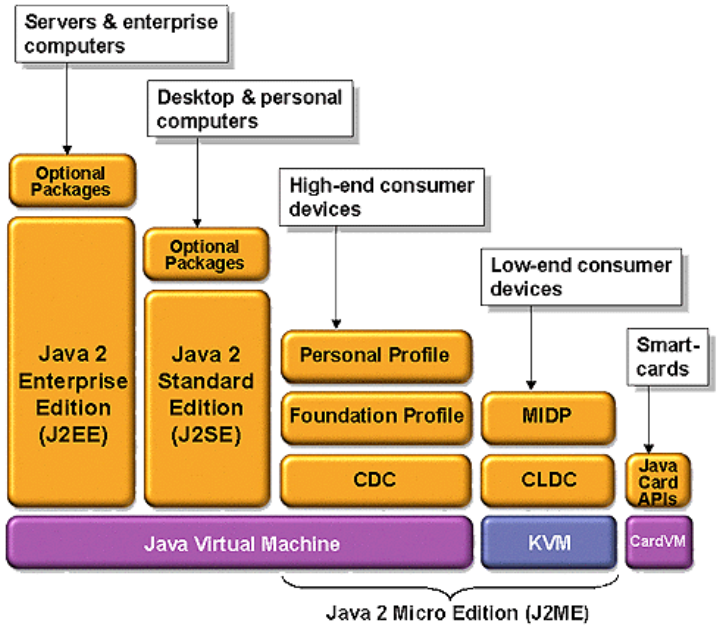


Arquitectura



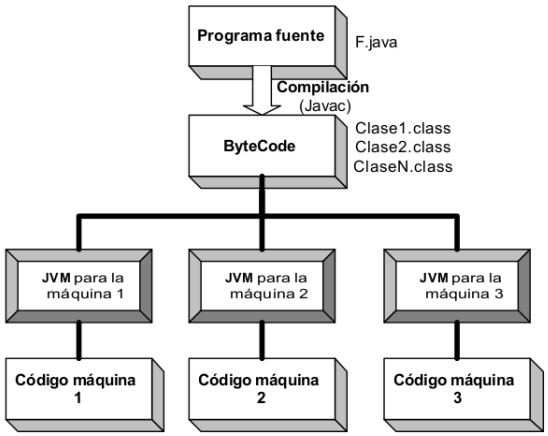
Introducción a Java

- **J2ME**: ~~Java 2 Micro Edition, se utiliza o utilizaba para desarrollo mobile antes de Android.~~ Tiene una porción reducida de J2SE. Actualmente tiene otro enfoque: Se la denomina Java Platform Micro Edition (Java ME) y proporciona un entorno robusto y flexible para aplicaciones que se ejecutan en dispositivos integrados y móviles en Internet de las cosas: microcontroladores, sensores, puertas de enlace, teléfonos móviles, asistentes digitales personales (PDA), set top box para, impresoras y más.

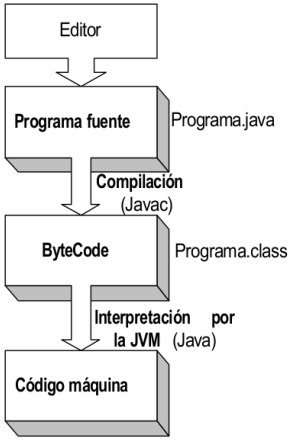


Introducción a Java

- **JVM:** Para compilar y correr aplicaciones Java necesitamos una Java Virtual Machine. El resultado de la compilación es normalmente un .jar, que la JVM puede ejecutar.



- Pero…. ¿El lenguaje es Compilado o Interpretado?



Introducción a Java

Pero… ¿Qué es un JAR?

- Es un tipo de archivo que permite ejecutar aplicaciones y herramientas escritas en el lenguaje Java.
- Es una abreviatura de Java Archive.
- Las siglas están deliberadamente escogidas para que coincidan con la palabra inglesa "jar" (tarro).
- Los archivos JAR están comprimidos con el formato ZIP y cambiada su extensión a .jar.

Introducción a Java

Y…. ¿Qué es el JDK?

- La herramienta básica para empezar a desarrollar aplicaciones en Java. El JDK (Java Developer's Kit) o Kit de Desarrollo Java, que consiste, básicamente, en un compilador y un intérprete (JVM) para la línea de comandos.
- No dispone de un entorno de desarrollo integrado (IDE), pero es suficiente para aprender el lenguaje y desarrollar pequeñas aplicaciones.
- **A partir de la versión de Java 9 se agrego la herramienta Jshell, la cual permite experimentar rápida y fácilmente con código Java.**
  - https://javadesdecero.es/fundamentos/jshell-herramienta-repl-java-9/

Introducción a Java

Instalación del JDK: (OpenJDK en realidad)

- Para Windows: https://docs.azul.com/core/zulu-openjdk/install/windows
- Para Linux: https://docs.azul.com/core/zulu-openjdk/install/debian
- Para macOS: https://docs.azul.com/core/zulu-openjdk/install/macos
- La fundacion Eclipse recomienda el JDK Eclipse Temurin https://adoptium.net/

Introducción a Java

Y…. ¿Qué es el JRE?

- Es el entorno de tiempo de ejecución de Java (JRE) que no es más que el conjunto de elementos que permite ejecutar una aplicación de Java, y forma parte del kit de desarrollo de Java (JDK).
- Puede ser instalado de manera independiente si es que no voy a desarrollar con el lenguaje.

Introducción a Java

- **Archivos .java y .class:** Los archivos fuente de Java tiene la extensión .java. Cuando se los compila, se generan archivos .class, estos últimos son los que se agrupan en el .jar.
- **Paquetes e Importaciones:** Java utiliza una organización de los archivos por paquetes, que son subdirectorios a nivel de sistema operativo, pero que a nivel lógico pueden ser identificado e importados dentro de cada .java.
- **Clases, Atributos y Métodos:** Se pueden definir Clases con atributos, métodos, constantes. Tanto las clases como los atributos, métodos y constantes puede tener modificador de visibilidad: **private, protected y public.**

Introducción a Java

- **Clases Abstractas e Interfaces:** Existen las clases abstractas, que permiten definir Clases que no se pueden instanciar por sí mismas, dado que a nivel conceptual no tienen un sentido como tales en un dominio determinado. Para que una clase sea abstracta debe tener al menos un método abstracto, puede tener atributos y otros métodos no abstractos. También están las **interfaces**, que permiten definir un contrato de comportamiento y constantes, posibilitando lo que en otros lenguajes se hace con herencia múltiple. Las interfaces no se pueden instanciar, deben ser implementadas por clases concretas.
- **Clases Anónimas:** Si bien las clases abstractas y las interfaces no se pueden instanciar, existen las clases anónimas, que es una forma de generar una instancia de una abstracta o interfaz sin identificarla, pero respetando los contratos.
- **La Clase Object:** En Java todas las clases heredan de Object, que es la clase base del lenguaje. Contiene algunas definiciones necesarias a todos los objetos del lenguaje.

Introducción a Java

- **Método main:** Para ejecutar una porción de código de Java hay que definir el método estático main en el .java dentro de la Clase, con el prototipo adecuado. Esto habilita a ese archivo a ser ejecutado. Puede haber múltiples archivos ejecutables dentro de un proyecto o .jar.
- **Gráficos: AWT, Swing, JavaFX:** J2SE trae AWT y Swing como bibliotecas base para el manejo de UI, JavaFX aparece en versiones más recientes como una capa por encima de las otras dos con algunas facilidades y características extra.
- **Anotaciones:** Sirven para extender el lenguaje con funcionalidad en tiempo de compilación y ejecución. Hay varias nativas, como @Override que indica antes de un método que éste es una redefinición de un método de la superclase o interface.

Introducción a Java

- **Modificadores:** Además de los modificadores de visibilidad que nombramos anteriormente, existen otros como:
  - **Static**: indica que lo que sigue es estático o de clase y no de instancia.
  - **Final**: indica que lo que sigue sólo podrá ser alterado en el proceso de inicialización y nada más.
  - **Abstract**: inicia que lo que sigue es abstracto, ya se una clase o método.
- **Entornos de Desarrollo (IDE):** Existen opciones en materia de entornos de desarrollo para Java, uno elige el que le quede cómodo. Algunos de ellos: Eclipse, NetBeans, IntelliJ IDEA, entre otros.

Herramientas de trabajo para Java y UML

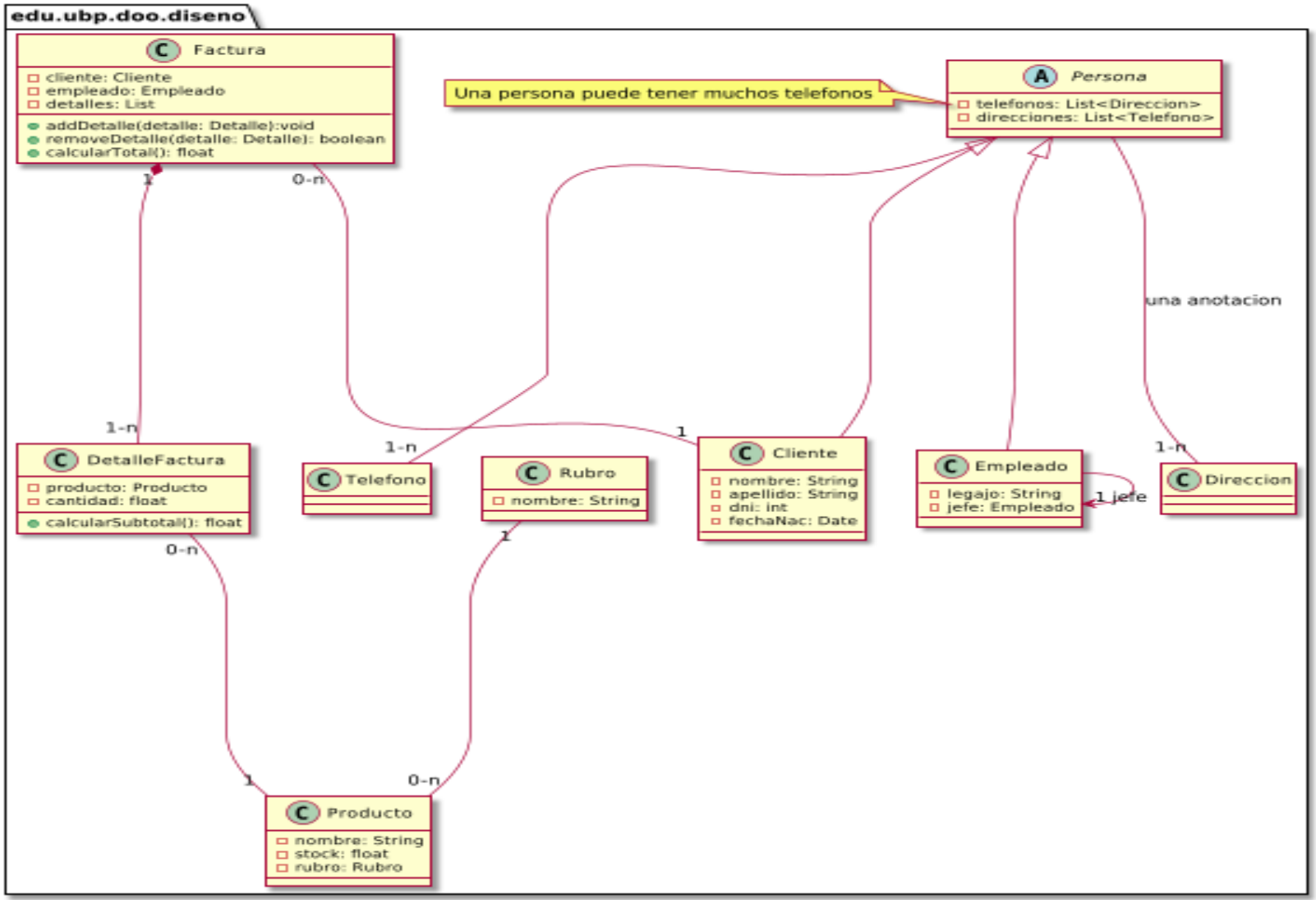
- Se recomienda NetBeans:
  - https://netbeans.apache.org/download/index.html
  - Otras IDEs:
    - Eclipse: https://www.eclipse.org/downloads/
    - IntelliJ: https://www.jetbrains.com/es-es/idea/download/
- Para UML: se recomienda PlantUML:
  - https://plantuml.com/es/
  - Plugin netbeans: https://plugins.netbeans.apache.org/catalogue/?id=58
  - Guia instalación: https://plantuml.com/es/starting
  - Editores online:
    - https://plantuml-editor.kkeisuke.com/
    - https://www.planttext.com/
    - http://sujoyu.github.io/plantuml-previewer/

1er Ejercicio practico

Enunciado:

- Se nos ha pedido que diseñemos el modelo de clases para poder construir un pequeño sistema de ventas.
- El equipo de análisis funcional ha podido recabar lo siguiente:
  - El modelo de clases debe ser capaz de representar los siguientes requisitos:
    1. Cuando un cliente se acerca al local puede comprar diferentes productos de distintos rubros a saber: Cosmética, higiene, farmacia, etc.
    2. De cada cliente es importante recabar los datos personales: nombre completo, dni, fecha nacimiento y también su dirección y teléfono. Respecto a estos dos últimos puntos se nos ha hecho hincapié en que cada cliente puede poseer varios teléfonos y direcciones a lo largo del tiempo.
    3. Cada producto pertenece a un único rubro de los ya mencionados.
    4. La aplicación debe tener también la capacidad de poder manejar los datos personales de los empleados teniendo en cuenta las mismas cuestiones mencionadas anteriormente para un cliente.
    5. Se aclara además que cada empleado, al atender a un cliente, debe siempre realizar una factura de compra que detalle: la fecha de compra, el monto total de dicha compra, qué cliente fue el que realizo la compra, qué empleado lo atendió y por supuesto el detalle de los productos adquiridos con sus cantidades y subtotales.
    6. Por último se nos a pedido modelar las relaciones jerárquicas de los empleados, es decir, poder saber quien es el jefe de cada empleado.

Resolución: diagrama de clases



Resolución extra: diagrama de casos de uso