Algoritmos y Estructuras de Datos Cursada 2011

Taller de Java

Prof: Lic. Laura Fava (lfava@info.unlp.edu.ar)

Java
El Lenguaje y la Plataforma

Java abarca dos aspectos:

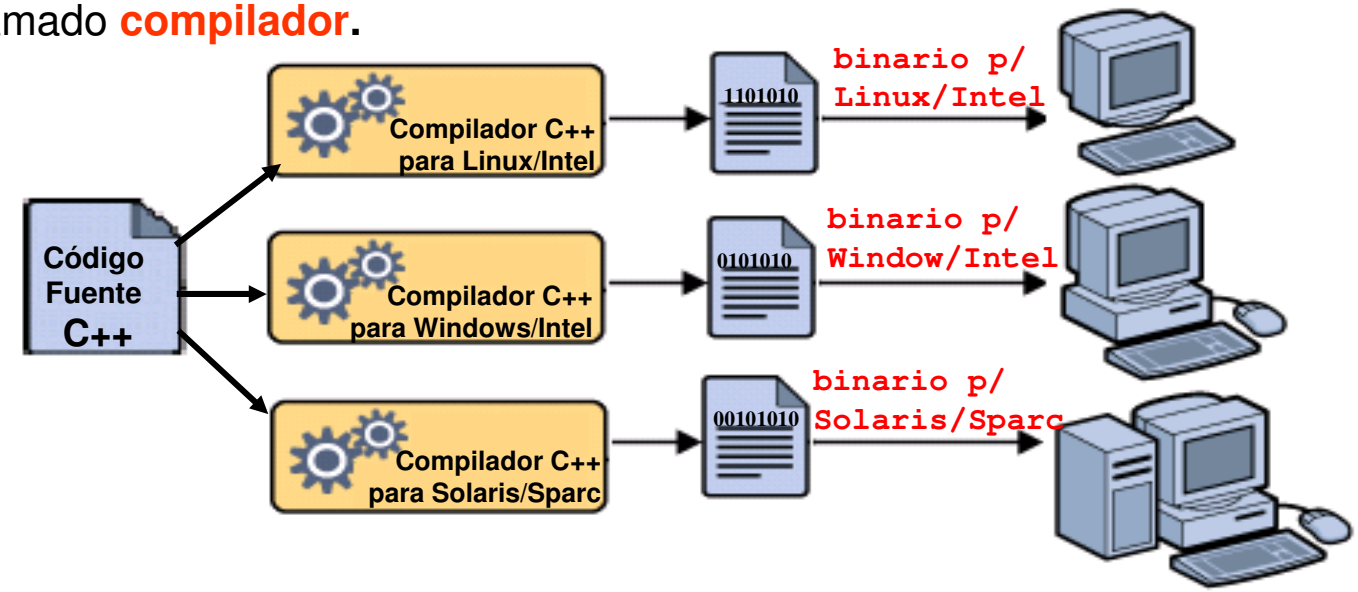
- Un **Lenguaje de Programación**
  - **Orientado a objetos**
  - **Independiente de la plataforma**
  - simple, seguro, distribuido, . . .
- Una **Plataforma**

La Plataforma Java
Introducción

Cada procesador requiere de un sistema operativo (SO), tal como **`Linux, Windows o Solaris`** para ejecutar programas, grabar archivos, leer de dispositivos, imprimir, etc. La combinación de procesador y sistema operativo se llama **plataforma de ejecución**.

- Los programas se escriben en **lenguajes de programación de alto nivel**, como Java, C++ o Pascal. Un programa escrito en un lenguaje de alto nivel, no puede ejecutarse directamente en la computadora, necesita ser traducido a lenguaje de máquina. Esta traducción puede realizarla un programa llamado **compilador.**

- El **lenguaje de máquina** o **código binario** consiste de instrucciones muy simples que la CPU de la computadora ejecuta directamente. Cada tipo de procesador tiene su propio lenguaje de máquina. Cada **código binario** es específico para cada plataforma.



Prof: Lic. Laura A. Fava

La Plataforma Java
Introducción

Una alternativa al compilador es un **intérprete**. Un **intérprete** es un programa que traduce y ejecuta un programa escrito en un lenguaje de alto nivel, instrucción por instrucción en el momento que se ejecuta (a diferencia del compilador que traduce el programa como un todo y genera un ejecutable).

Los programas escritos en **Java** se **compilan** e **interpretan**:

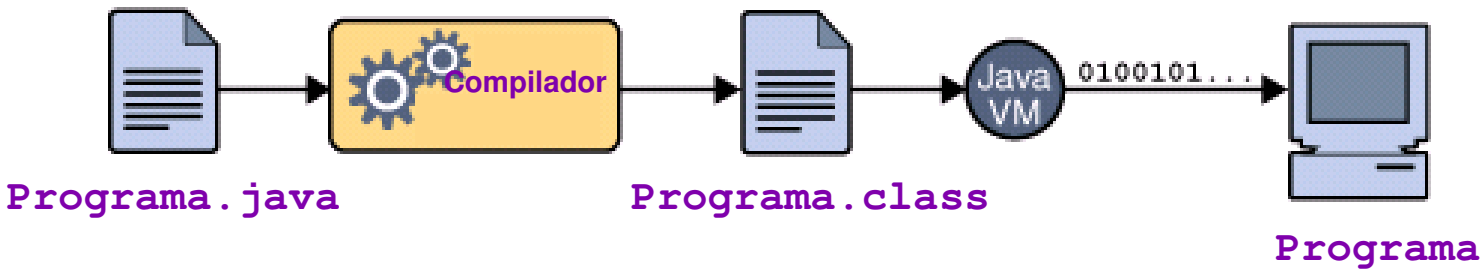
- un programa escrito en Java se compila a un lenguaje de máquina de una computadora virtual, llamada **Máquina Virtual Java (JVM)**. La JVM es **software**. El lenguaje de máquina de la **JVM** se llama **código de bytes Java** (en inglés *java bytecodes*).
- el **programa Java** compilado, luego es **interpretado**.

Un programa Java compilado, puede ejecutarse sobre cualquier plataforma que disponga de una JVM. El intérprete ejecuta el *código de bytes Java*.

La Plataforma Java
Introducción

En java, el código fuente es escrito en archivos con texto plano con extensión **.java**. Esos archivos son posteriormente compilados en archivos con extensión **.class** por el compilador java (**javac.exe**).

Un archivo .class no contiene código nativo/específico para un procesador determinado, sino que contiene **bytecodes** (el lenguaje de la máquina virtual de java –JavaVM-).



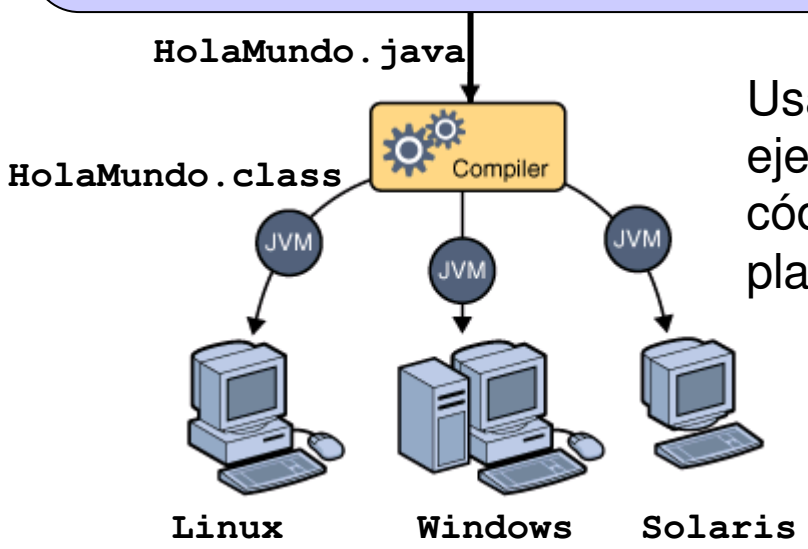
El **java.exe** es un programa que viene con la plataforma java, que permite ejecutar los **bytecodes**, es el intérprete java.

La Plataforma Java
Introducción

Existen diferentes Máquinas Virtuales Java para los diferentes sistemas operativos. **Los mismos archivos `.class` pueden ejecutar en los distintos sistemas operativos**, como Microsoft Windows, Solaris, Linux o Mac, sin ninguna compilación previa.

```
class HolaMundo {
public static void main(String[] args){
     System.out.println("Hola !!!");
  }
}
```

HolaMundo.java



Usando la **JVM**, la misma aplicación es capaz de ejecutar sobre las distintas plataformas => el código de bytes `(.class)` es independiente de la plataforma.

La Plataforma Java

Java provee una plataforma de software para **desarrollar** programas y otra para **ejecutarlos**.

- **Plataforma de Desarrollo**

  El **Java Development Kit (JDK)** es la plataforma básica para desarrollo de programas Java. Actualmente, el nombre oficial es **Java SE o JSE** (Java Standar Edition).
  Incluye herramientas tales como un compilador, un intérprete, un depurador, un documentador, un empaquetador de clases, etc. Estas herramientas se usan desde la línea de comando.

- **Plataforma de Ejecución**

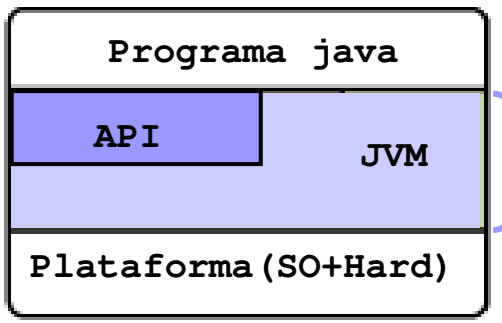
  La plataforma de ejecución de Java se llama **JRE (Java Runtime Environment)** y provee todas las componentes necesarias para ejecutar programas escritos para JSE (programas de escritorio o applets). La **JVM** es parte del **JRE**.
  Los programas Java se ejecutan sobre la máquina de software llamada **MVJ**.

Java SE 6 (update 24) es la últma versión de la plataforma Java SE, la url es la siguiente: http://www.oracle.com/technetwork/java/javase/downloads/index.html

Prof: Lic. Laura A. Fava

La Plataforma de Ejecución Java JRE

- **La JVM es una máquina de software que emula una máquina real**
- Es el corazón de la **plataforma Java (jre)**.
- Aisla al programa Java del Sistema Operativo (SO) y del hardware sobre el que se está ejecutando.
- Provee "independencia" de la plataforma.



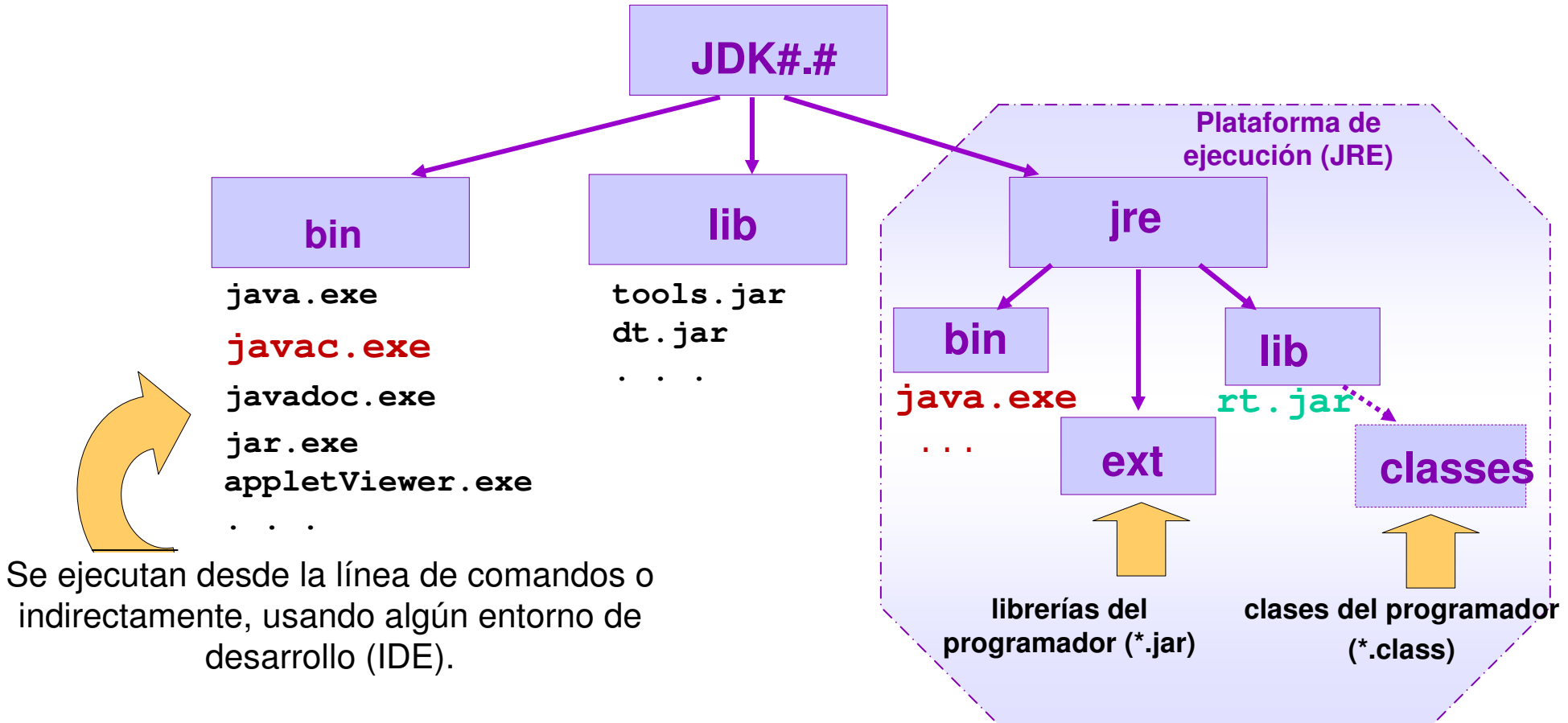
El código Java compilado tiene instrucciones específicas para la MVJ

La Plataforma Java

- La especificación de la MVJ es única. La especificación de la MVJ permite que el software Java sea "independiente de la plataforma" ya que se compila para una máquina genérica o MVJ.
- La especificación de la MVJ provee un estándar. Cada SO tiene su propia implementación de la MVJ.

Java SE o JDK

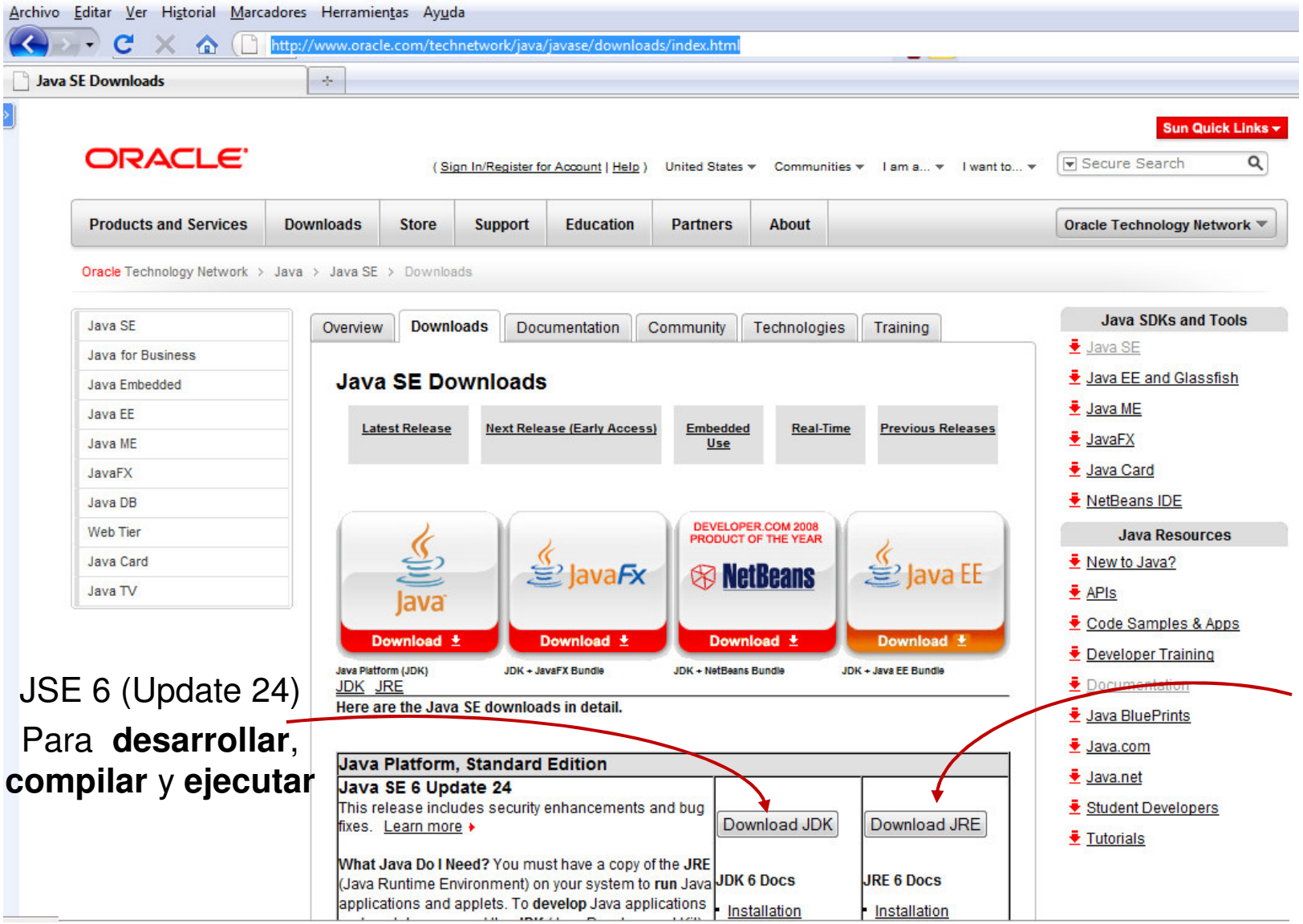
La plataforma provee un conjunto de herramientas para desarrollar y ejecutar programas Java. La **estructura de directorios del J2SE** es la siguiente:



Se ejecutan desde la línea de comandos o indirectamente, usando algún entorno de desarrollo (IDE).

¿Dónde encontrar las Ediciones de Java para descargar?

http://www.oracle.com/technetwork/java/javase/downloads/index.html



JSE 6 (Update 24)

Para **desarrollar**, **compilar** y **ejecutar**

JRE 6 (Update 24)

Para **ejecutar**

Ediciones de Java

- **JSE (Java Standard Edition):** está diseñada para programar y ejecutar applets y aplicaciones de escritorio JAVA. Típicamente son programas que se ejecutan en una PC. Es el fundamento de las 2 restantes ediciones. Está compuesta por el JRE y el JDK.
- **JEE (Java Entreprise Edition):** está diseñada para programar y ejecutar aplicaciones empresariales, caracterizadas por ser multiusuario y distribuidas. El procesamiento de estas aplicaciones se realiza en un servidor. Usualmente son aplicaciones web.
- **JME (Java Micro Edition):** está diseñada para programar y ejecutar aplicaciones para dispositivos con recursos de cómputo limitados, como pueden ser teléfonos celulares, palms, pdas, etc. Estos dispositivos cuentan con poca memoria RAM, pantallas muy chicas inclusive algunos carecen de ellas, la conexión de red puede ser intermitente, etc.

Programación Orientada a Objetos

Fundamentos

El proceso de abstracción

Las aplicaciones de software típicas, modelan el mundo real. El **mundo real es complejo** a simple vista y, cuando se lo observa con más detalle, el nivel de complejidad crece.

¿cómo modelamos este mundo tan complejo?

Los humanos entendemos al mundo, construyendo modelos mentales de partes del mismo. Un modelo mental es una **visión simplificada de cómo las cosas funcionan y cómo podemos interactuar con ellas**.
La abstracción es uno de los mecanismos que los humanos utilizamos para combatir la complejidad. La orientación a objetos, maneja la complejidad de los problemas del mundo real, abstrayendo su conocimiento y encapsulándolo en objetos.
=> es clave en el desarrollo de software.

¿cuál es el objetivo buscado por la programación orientada a objetos?

Organizar los **datos del programa y el procesamiento asociado a ellos**, en entidades coherentes, llamadas **objetos**. Cada objeto abstrae un dato del programa y lo que puede hacerse sobre él. La

Programación Orientada a Objetos
Fundamentos

Pensemos en un ejemplo de la vida real

Definición de un problema

Supongamos que Juan quiere hacer un pedido de pizzas para que se las entreguen en su domicilio.

¿qué hace?

Consigue el teléfono de la pizzería y llama. Lo atiende un telefonista, le hace el pedido deseado, indicándole la cantidad y tipos de pizzas que desea y el domicilio a donde debe enviarse el pedido **=> se resolvió el problema**.

Juan se comunicó con el **Telefonista**, y le pasó un **mensaje** con el requerimiento. El telefonista tiene la responsabilidad de satisfacer el requerimiento.

Juan

Juan no necesita saber como el telefonista resolverá el problema.

Mensaje
con el requerimiento

Telefonista
(agente apropiado)

El telefonista, usará algún **método** para satisfacer el requerimiento.

Prof: Lic. Laura A. Fava

Programación Orientada a Objetos
Fundamentos

agentes u objetos

La resolución del problema, requiere de la ayuda de otros individuos. Sin su colaboración, la resolución no sería fácil.

1º principio de programación orientado a objetos

Un programa **orientado a objetos**, está organizado como una comunidad de agentes interactuando, llamados **objetos**. Cada objeto cumple un rol. Cada objeto provee un servicio o ejecuta una acción, que es usada por otros miembros de la comunidad.

Prof: Lic. Laura A. Fava

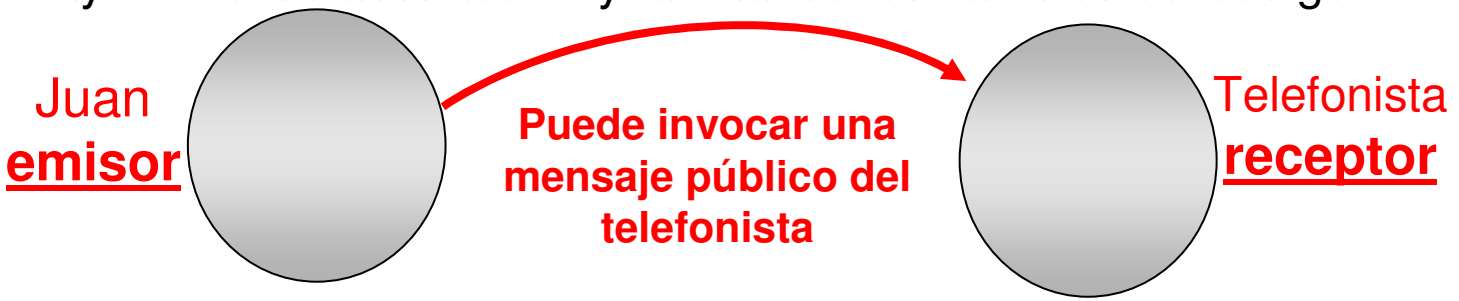
Programación Orientada a Objetos

Fundamentos

Ocultamiento de información

Un objeto (cliente, telefonista, repartidor, etc.) es una entidad que contiene información y operaciones relacionadas, que tiene sentido agrupar (empaquetar). Este concepto, en el contexto de POO es conocido como **encapsulamiento**.

Comúnmente los objetos son como cápsulas opacas, con una interfaz pública y una representación privada. Este concepto se conoce como **ocultamiento de información** (***information hiding***). Permite eliminar de la vista cierta información propia del objeto, logrando mayor nivel de abstracción y facilitando los cambios del código.



2º principio de programación orientado a objetos

El **encapsulamiento y el ocultamiento de información** se complementan, para aislar las diferentes partes de un sistema, permitiendo que el código sea modificado, extendido y que se puedan corregir errores, sin el riesgo de producir efectos colaterales no intencionados.
Los objetos ponen en práctica estos dos conceptos:

1. Se abstrae la funcionalidad y la información relacionada y se encapsulan en un objeto.
2. Se decide que funcionalidad e información, podrá ser requerida por otros objetos y el resto se oculta.

Programación Orientada a Objetos

Fundamentos

mensajes y métodos

Juan hace un primer requerimiento al telefonista, quien hace otro requerimiento que conduce a más y más requerimientos, hasta que se resuelve el problema: la pizza le llega a Juan!.

3º principio de programación orientado a objetos

Una acción es iniciada cuando un objeto, el **emisor**, envía un mensaje a un agente responsable de la acción, el **receptor**. El **mensaje** representa el requerimiento y es acompañado por información adicional **(argumentos)** necesaria para cumplir el requerimiento. El receptor es el objeto a quien se le envía el mensaje. El receptor en respuesta al mensaje ejecutará un conjunto de acciones o método para satisfacer el requerimiento.

Prof: Lic. Laura A. Fava

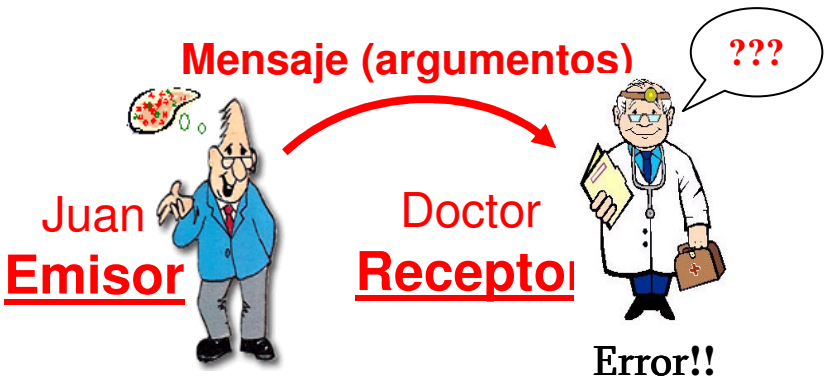
Programación Orientada a Objetos
Fundamentos

mensajes y métodos vs. llamadas a procedimiento

Existen 2 distinciones importantes:

(1) En un mensaje, siempre hay designado un receptor para aquel mensaje; el receptor es algún objeto, al cual se le envía un mensaje. Cuando se llama a un procedimiento, NO hay receptor.

(2) La interpretación del mensaje (el método usado para responder al mensaje) es determinado por el receptor y podría variar para diferentes receptores.



Prof: Lic. Laura A. Fava

Programación Orientada a Objetos
Fundamentos

Clases e instancias

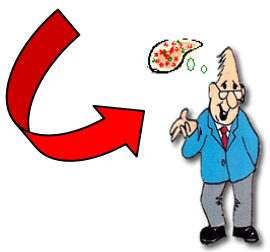
¿qué sabe el telefonista acerca de Juan?

El telefonista sabe que quien está llamando, **es un cliente** y puede asumir, por ello, ciertas cosas. Cree por ejemplo, que Juan, su cliente, le abonará las pizzas cuando las reciba en su domicilio => se comporta como un cliente. Esto es porque Juan pertenece a una **categoría** o **clase** que podríamos llamar `Cliente` y el telefonista espera que Juan siendo una **instancia** de esta categoría, se ajuste a un patrón. Todos los clientes se deberían ajustar a este comportamiento.

4º principio de programación orientado a objetos

Todos los objetos son instancias de una clase. El método invocado por un objeto en respuesta a un mensaje es determinado por la clase del objeto receptor.

Una clase es un molde a partir de la cual se crean instancias con las mismas características y comportamiento.



Juan, es una instancia de la clase `Cliente`.

Prof: Lic. Laura A. Fava

Programación Orientada a Objetos
Fundamentos

Clases e instancias

Una **instancia u objeto** es una entidad de software que combina un **estado/datos** y **comportamiento/métodos**.

Juan, es una instancia de la clase `Cliente`, es un objeto de tipo `Cliente`

Todos los objeto de tipo `Cliente`

estado: #cliente, domicilio de entrega, deuda.

comportamiento: dar#Cliente, abonar pedido, ...

- El **estado** de un objeto es todo lo que el objeto **conoce de si mismo** y, el **comportamiento** es todo lo que el objeto **puede hacer**.
- Un **objeto** mantiene su **estado** en **variables** y su **comportamiento** está implementado en los **métodos** de la clase a la que pertenece.

Programación Orientada a Objetos
Fundamentos

Herencia

¿qué más sabe el telefonista acerca de Juan?

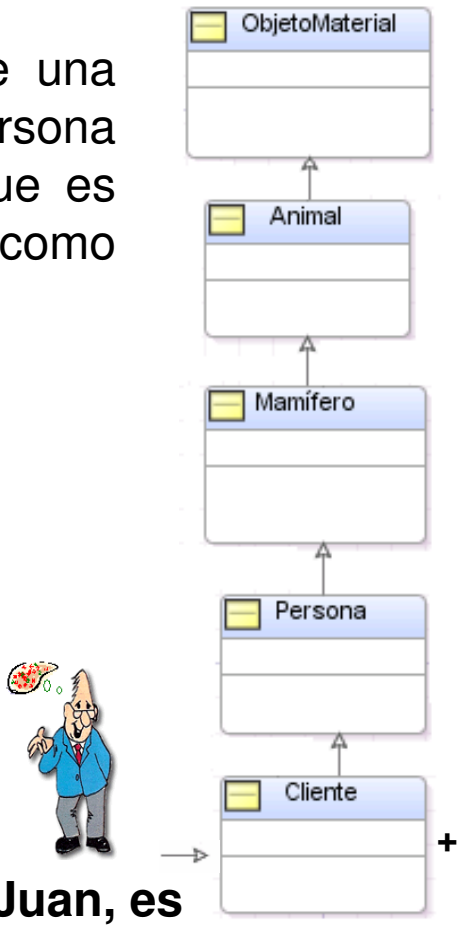
El podría pensar u organizar el conocimiento en términos de una **jerarquía de categorías**. Juan, es un cliente, es una persona especial. Al conocer que es una persona o humano, sabe que es bípedo y como también es un mamífero sabe que tiene pelo y como es animal, sabe que respira oxígeno.

5º principio de programación orientado a objetos

El conocimiento de una categoría más general, es también aplicable a una categoría mas específica y se denomina **herencia.**

Las clases pueden ser organizadas en jerarquías de herencia donde, las clases hijas o **subclases**, heredarán **estado y comportamiento** de las clases que se encuentran más arriba en la jerarquía, llamadas **superclases**.

Las subclases pueden agregar nuevas variables y métodos y pueden cambiar el comportamiento de los métodos heredados.



Juan, es un Cliente

Prof: Lic. Laura A. Fava