JAVA

POO (Repaso)

LOS 3 PRINCIPIOS DE LA PROGRAMACIÓN ORIENTADA A OBJETOS (POO)

ENCAPSULAMIENTO

Encapsulamiento es el ocultamiento del estado o de la representación interna de un objeto. Donde el acceso de lectura-escritura se otorga a través de métodos de acceso público. Una forma de ver al encapsulamiento es como un envoltorio o capa protectora que previene que el código y datos sean accedidos arbitrariamente por otro código definido fuera de la capa protectora (o envoltorio).

En Java, la base del encapsulamiento es la clase. La clase define una estructura y comportamiento (datos y código) que serán compartidos por un conjunto de objetos. Cada objeto de una clase dada contiene la estructura y comportamiento definido por la clase, como si hubieran sido sacados de un molde con la forma de la clase. Es por esto, que los objetos suelen ser denominados instancias de una clase. Una clase es una representación lógica, un objeto en cambio podremos decir que es una representación física.

Una clase usualmente contiene atributos, métodos y constructores.

Siendo el propósito de una clase el de encapsular complejidad, existen mecanismos para esconder la complejidad de una implementación dentro de una clase. Cada método o variable en una clase puede ser marcado como público o privado. La interfaz pública de una clase representa todo lo que los usuarios externos de una clase necesitan conocer, o pueden conocer. Los atributos y métodos privados pueden ser solo accedidos por código dentro de la clase.

Animal.java > Animal

```java
public class Animal {

    private int peso;

    public int getPeso() {
        return peso;
    }

    public void setPeso(int peso) {
        this.peso = peso;
    }
}
```

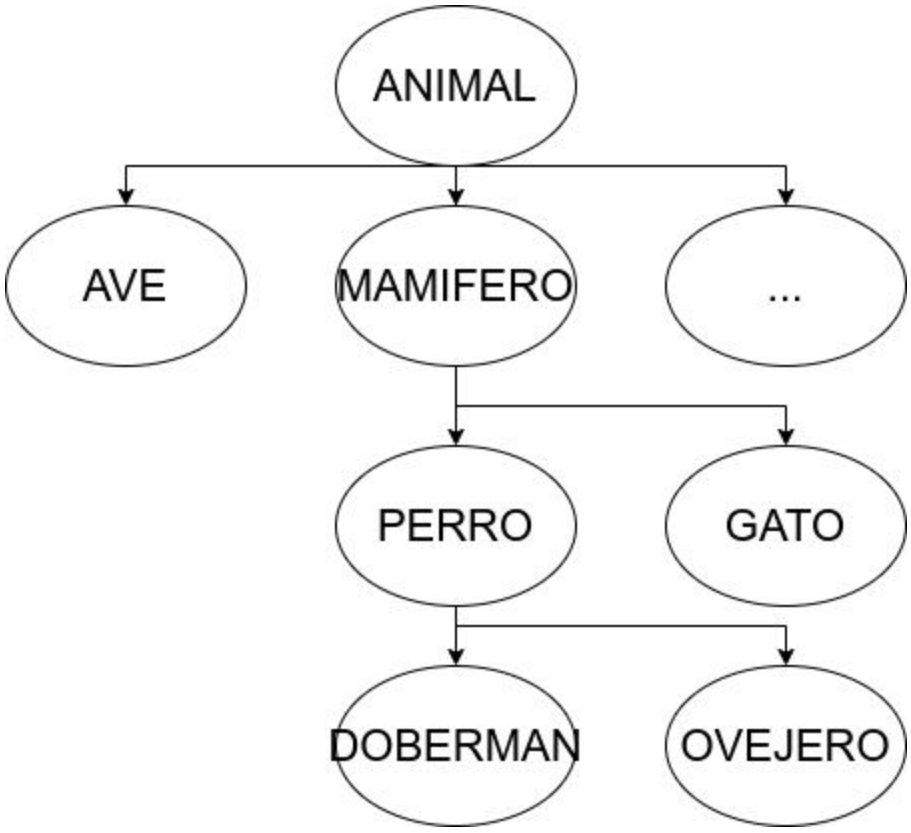
En la figura de arriba vemos que el atributo peso es encapsulado usando el modificador de acceso ***private***, y solo puede ser accedido y modificado por los métodos ***public getPeso()*** y ***setPeso()*** .

HERENCIA

Es el proceso en el que un objeto adquiere las propiedades de otro objeto. Y donde podemos relacionarlo con un concepto de clasificación jerárquica.
En Java logramos esto al extender de una clase padre, de esta forma la clase hija obtiene todos los atributos del padre.
Cuando definimos una relación de herencia, formamos una r**elación de ES-UN**. Ejemplo, un Doberman ES-UN Perro, una Manzana ES-UNA Fruta, o viendo desde el punto de vista de un aula virtual un Alumno (o Profesor) ES-UN Usuario.

```java
Perro.java > Perro
1   public class Perro extends Animal {
2       //...
3   }
```

Ejemplo: Un Doberman es parte de la clasificación Perro, que a su vez es parte de la clase Mamífero, que está por debajo de la clase Animal. Sin la definición de jerarquías, cada objeto debería definir todas sus características explícitamente.



De la misma forma podemos definir una clase Automóvil, donde si evaluamos que tienen en común todos los automóviles podríamos decir que todos los automóviles poseen atributos en común como ruedas, motor y más: y comportamientos en común también como acelerar, frenar, etc. Si quisiéramos describir una clases más específicas de Automóvil podríamos definir una clase Automovil Electrico, Automovil a Base de Nafta, etc. que tendrán atributos específicos como lo es el motor. Esto es es conocido como subclases, donde Automóvil Eléctrico (o base de Nafta) es referenciado como Automóvil (superclase).

La herencia interactúa con la encapsulación. Si una clase encapsula algunos atributos, entonces cualquier subclase tendrá los mismos atributos más cualquiera que sean agregados como parte de la especialización.

POLIMORFISMO

Es la característica que permite que una interface sea usada para acciones de tipo general. Esto significa que es posible diseñar una interface genérica para un grupo de actividades relacionadas.

Siguiendo la analogía de la clase Perro, el sentido del olfato del perro es polimórfico. Si el perro huele un gato, ladrara y correrá detrás de él. Si el perro huele comida, se lamera los labios y correrá hasta su plato de comida. El sentido del olfato está en funcionamiento en ambas situaciones. La diferencia es que lo que está siendo olido, eso es, el tipo de dato que está siendo operado por la nariz del perro.

REPASO DE CONCEPTOS BASICOS DE POO

CLASE

Es una plantilla o plano para la creación de objetos, describe los tipos de estado y comportamientos que puede soportar objetos creados a partir de una clase.

OBJETO

Los objetos creados a partir de una clase son llamados **instancias de clase**. En Java cuando la Java Virtual Machine (JVM) encuentra la palabra ***new***, usará la clase apropiada para crear el objeto que es una instancia de esa clase. Ese objeto tendrá su propio estado, y acceso a todos los comportamientos definidos por su clase.

ESTADO (variables de instancia)

Cada objeto (instancia de clase) tendrá su propio y único conjunto de variables de instancia como se definen en la clase. En conjunto, los valores asignados a las variables de instancia del objeto definen el **estado del objeto**.

COMPORTAMIENTO (métodos)

Los métodos son donde la lógica de la clase se encuentra. Es donde los algoritmos se ejecutan, y los datos son manipulados.

ABSTRACCIÓN

Es el ocultamiento de detalles complejos de implementaciones de un programa. En Java, lograremos esto con el uso de interfaces y clases abstractas.

CONCLUSIÓN

En este apunte repasamos algunos conceptos del paradigma de programación orientada a objetos (POO) y como relacionarlos en el lenguaje de Java.
A medida que avancemos en la práctica de desarrollo de programas, aplicaciones o proyectos en Java deberemos enfocarnos en aplicar estos conceptos.