PROGRAMACION ORIENTADA A OBJETOS (POO)

1. Orientación a objetos
2. Observar los problemas en forma de objetos
3. CLASES PROPIEDADES METODOS Y OBJETOS
4. ENCAPSULAMIENTO ABSTRACCION HERENCIA Y POLIMORFISMO

**JAVA**

* Lenguaje orientado a objetos natiralmente
* Android
* Server side

.java

**PHP**

* Lenguaje interpretado
* Pensado para la web

.php

**PYTHON**

* Diseñado para ser fácil de usar
* Múltiples usos: web, server side, análisis de datos, machine learning, etc

.py

**JAVASCRIPT**

* Lenguaje interpretado
* Orientado a onjetos pero basado en prototipos
* Pensado para la web

.js

**OMT OBJECT MODELING TECHNIQUES 1991(descontinuado)**

Plasmar análisis en un grafico

**UNIFIED MODELING LANGUAGE UML 1997**

* Clases
* Casos de uso
* Objetos
* Actividades
* Iteración
* Estados
* Implementación

Como ya viste UML significa Unified Modeling Language el cual es un lenguaje estándar de modelado de sistemas orientados a objetos.

Esto significa que tendremos una manera gráfica de representar una situación, justo como hemos venido viendo. A continuación te voy a presentar los elementos que puedes utilizar para hacer estas representaciones.

Las clases se representan así:

En la parte superior se colocan los atributos o propiedades, y debajo las operaciones de la clase. Notarás que el primer caracter con el que empiezan es un símbolo. Este denotará la visibilidad del atributo o método, esto es un término que tiene que ver con Encapsulamiento y veremos más adelante a detalle.

Estos son los niveles de visibilidad que puedes tener:

- private

+ public

# protected

~ default

Una forma de representar las relaciones que tendrá un elemento con otro es a través de las flechas en UML, y aquí tenemos varios tipos, estos son los más comunes:

Asociación

associacion.png

Como su nombre lo dice, notarás que cada vez que esté referenciada este tipo de flecha significará que ese elemento contiene al otro en su definición. La flecha apuntará hacia la dependencia.

uml-relacion-asociacion.jpg

Con esto vemos que la ClaseA está asociada y depende de la ClaseB.

Herencia

herencia.png

Siempre que veamos este tipo de flecha se estará expresando la herencia.

La dirección de la flecha irá desde el hijo hasta el padre.

herencia-clases.png

Con esto vemos que la ClaseB hereda de la ClaseA

Agregación

agregacion.png

Este se parece a la asociación en que un elemento dependerá del otro, pero en este caso será: Un elemento dependerá de muchos otros. Aquí tomamos como referencia la multiplicidad del elemento. Lo que comúnmente conocerías en Bases de Datos como Relaciones uno a muchos.

Con esto decimos que la ClaseA contiene varios elementos de la ClaseB. Estos últimos son comúnmente representados con listas o colecciones de datos.

Composición

composicion.png

Este es similar al anterior solo que su relación es totalmente compenetrada de tal modo que conceptualmente una de estas clases no podría vivir si no existiera la otra.

uml-relacion-composicion.jpg

Con esto terminamos nuestro primer módulo. Vamos al siguiente para entender cómo podemos hacer un análisis y utilizar estos elementos para construir nuestro diagrama de clases de Uber.

Diagrama

Descripción generada automáticamente

**OBJETOS**

Cuando tengamos un problema lo primero que debemos hacer es **IDENTIFICAR OBJETOS.**

* Objetos son aquellos que tienen propiedades y comportamientos
* Los objetos pueden ser físicos o conceptuales

Ejemplos:

* Un usuario es un objeto físico
* Una sesión de usuario es intangible (Conceptual)
* Una cuenta de banco es un objeto conceptual
* Una tarjeta de banco es un objeto físico

**Atributos**

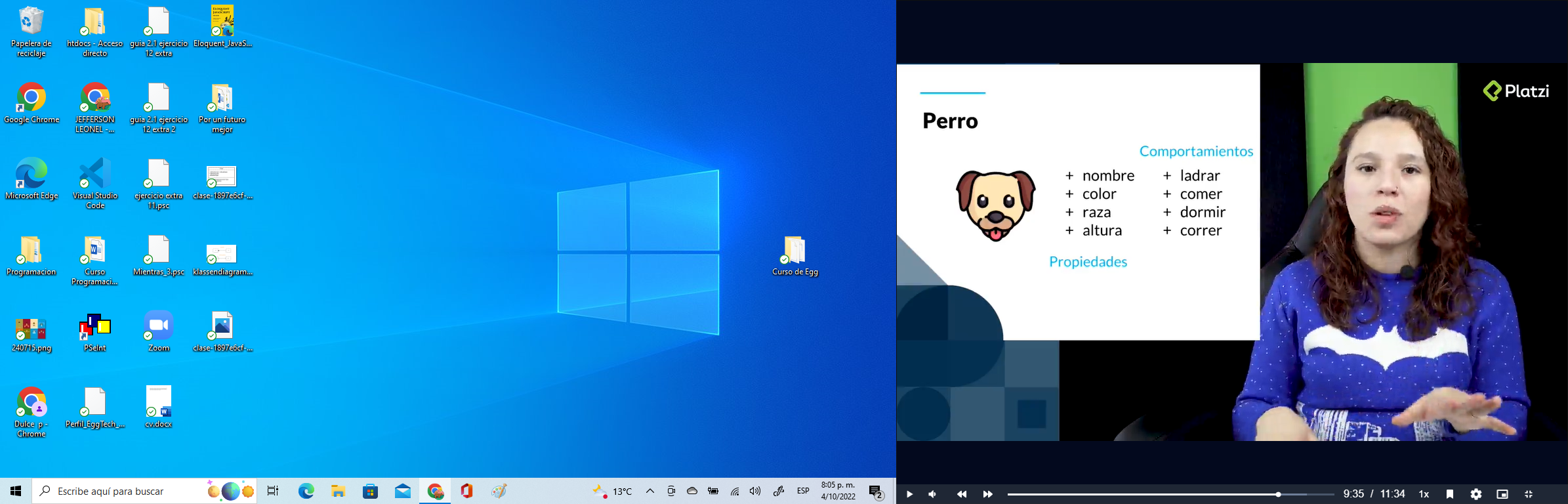
Propiedades también pueden llamarse **atributos** serán sustantivos

1. Nombre, tamaño, forma y estado.

**Comportamientos**

Serán todas las operaciones del objeto, suelen ser verbos o sustantivo y verbo

1. Login(), logout(), makeReport()



Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente

**CLASE**

Es el modelo sobre el cual se construira nuestro objeto.

Las clases me permitirán generar mas objetos

La clase es un modelo para generar mas objetos

* Analizar objetos para crear clases

*Abstracción*

Cuando nosotros abstraemos los datos de un objeto para generar un molde

**MODULARIDAD**

Diseño modular: Subdividir un sistema en partes más pequeñas

Un sistema organizado en módulos