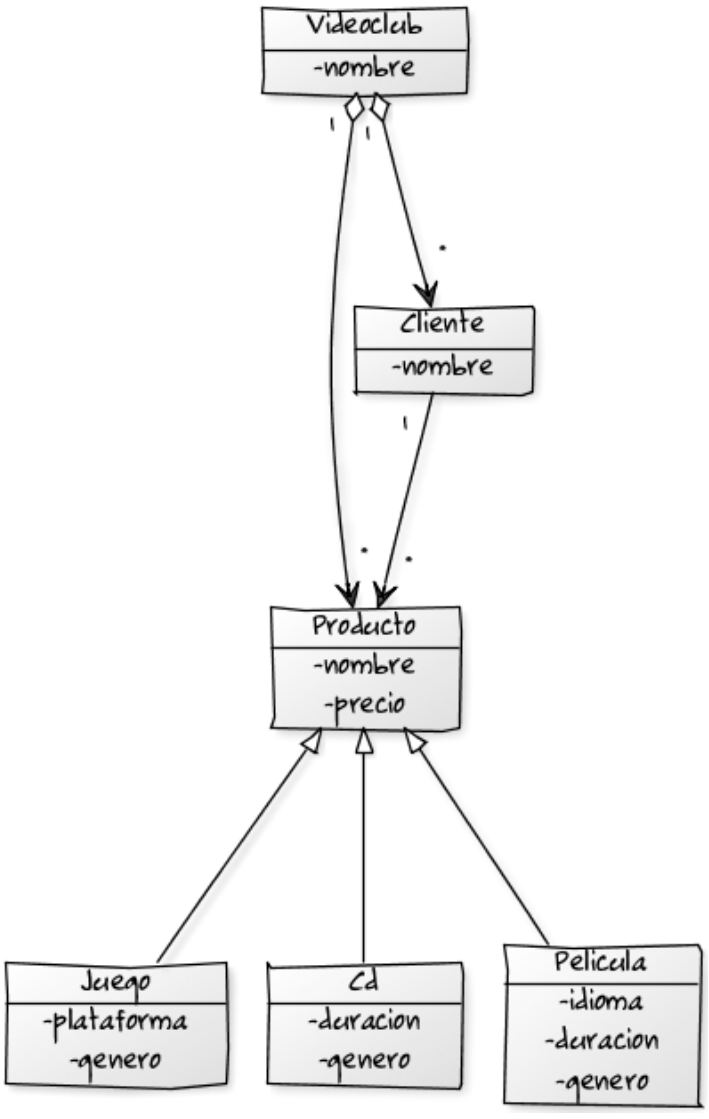
# Tarea UD 4. Objetos en PHP


En esta ocasión os propongo realizar la siguiente tarea para practicar varios de los conceptos que hemos visto sobre clases y objetos (no todos, claro). Os muestro un diagrama con la estructura UML que representa gráficamente las clases que utilizaréis en este ejercicio.

Un vez creada correctamente nuestra estructura de clases, lo ideal sería hacer una web en php que nos permita realizar todas las tareas, como dar de alta clientes, películas, alquileres..etc..Pero se nos alargaría mucho en el tiempo. (Puede ser un pero que muy buen trabajo de ampliación y os vendría muy bien a todos.). Lo que yo pediré, será un pequeño script en php donde hagáis uso de estas clases para probar que todo funciona correctamente.



Os lo detallo a continuación.

● La clase **Videoclub**: es la clase principal de nuestra aplicación. Representa a un videoclub en el dominio del problema y cuenta con una colección de clientes y productos registrados. Así como métodos para alquilar un producto a un cliente, y añadir nuevos productos y clientes en el videoclub.

● La clase **Cliente**: representa a un cliente. Cuenta con una variable que representa la colección de productos alquilados. Pensad en esta y en las demás clases qué métodos necesitamos: habrá que dar la opción de asignarle un producto en alquiler a un cliente, ¿verdad?, y saber qué tiene alquilado...

● La clase **Producto**: será una clase *abstracta* que representa un producto del videoclub. No podemos instanciar objetos *Producto*. Cuenta con una serie de atributos que heredarán las subclases: **nombre y precio**.

● Las clases **Pelicula, Cd, Juego**: clases derivadas de la clase padre **Producto**. Representan los diferentes productos del que dispone el videoclub. Cada producto tiene asignado un precio diferente de alquiler. Para probar las clases abstractas, herencia y reemplazo de métodos, la clase **Producto** ha de contar con un método abstracto **getPrecio()** que, como ya sabes, deberán implementar las clases derivadas. Así, cada tipo de *Producto* contará con un método **getPrecio()** que devolverá el precio en función del tipo de producto que sea. Las películas tendrán un precio de 2 euros, los cds de música de 1 euro y los juegos de 3 euros. Ojo, prohibido usar un campo 'tipo' tanto en producto como en sus subclases.

Tras esta breve descripción de las clases del dominio de nuestra aplicación, debéis crear el código PHP que utilizaremos para instanciar objetos.

- Clase Cliente
- Clase Producto
- Clase Película
- Clase CD
- Clase Juego
- Clase VideoClub

Cada Clase se deberá implementar y entregar en un fichero diferente: NombreDeLaClase.class.php, como por ejemplo: cliente.class.php (esta forma de nombrar es simplemente para saber de un vistazo que ese script php es la definición de una clase)

Por último, deberéis crear un fichero ***aplicacion.php*** en el que tenéis que instanciar objetos y gestionar nuestro videoclub. Cread un pequeño script de ejemplo que muestre el manejo de estos objetos.

¿Qué cosas podríamos hacer en este script para probar las clases? Por ejemplo:

- *Instanciar la clase videoclub.*
- *Instanciar varios Clientes (**varios siempre es más de dos**).*
- *Instanciar varios Productos de cada tipo.*
- *Agregar los clientes y los productos al videoclub.*

- *Que algún cliente alquile al menos 2 películas.*
- *Que algún cliente alquile al menos 1 videojuego.*
- *Mostrar los clientes del videoclub.*
- *Mostrar pelis/cd/videojuegos del videoclub.*
- *Mostrar alquileres.*
- *Etc...*

Así pues, preparad el script ***aplicacion.php*** para comprobar que todo funciona correctamente.
**Al menos, tenéis que:**

● Crear un videoclub de prueba
● En el videoclub
  - Crear al menos 4 clientes.
  - Crearemos al menos 5 películas, 5 cd y 5 juegos.
  - OPCIONAL: La carga de clientes y de películas se puede hacer desde dos ficheros: clientes.txt y productos.txt
  - asignar varios alquileres.
  - Mostrar los datos por pantalla: Clientes del videoclub, alquileres, etc. Una vez hecho esto, haced que muestre por pantalla todos los clientes del Video Club, e información sobre los alquileres de cada uno de ellos, o bien si no tiene nada alquilado

● fijaos en el diagrama para ver, entre otras cosas, algunos atributos básicos que pueden no aparecer en la descripción.

● Tenéis hacer uso e implementar el método mágico **__toString** en la mayoría de las clases. Preparad estos métodos mágicos y haced uso de ellos para que nos muestre la información de manera correcta y completa.

● Todos las clases deberán implementar el método mágico: **__construct** que, como sabéis, se utiliza normalmente para inicializar los atributos del objeto que se está creando.

● Un ejemplo: cuando creamos un objeto de la clase Cd, al constructor le llegará el nombre, duración, genero y precio. Pero tened en cuenta que dos de estos atributos se heredan de la clase padre. ¿cómo inicializo esos valores? En este caso, quiero que llameis para ello al constructor de la clase padre.

● Como norma general, todos los atributos serán privados (o protegidos si lo necesitáis), y los métodos públicos. Trabajando de esta manera, habrá que crear siempre que trabajemos con objetos aunque no se pida expresamente, los métodos s*etAtributo* y get*Atributo*, para poder acceder a estos atributos desde fuera de la clase. Es decir: setNombre(), getNombre(),...etc.

Utilizad en este caso los foros para consultar como hacer una cosa u otra, o para llegar a un consenso de cómo hacer tal o cual cosa de manera apropiada...etc

**Debéis entregar un ZIP con todos lo archivos del ejercicio**.