ILERNA Online

Videotutoría 7: Diseño orientado a objetos

Módulo 05: Entornos de Desarrollo

RESUMEN

¿Qué habéis aprendido la VT anterior?

Objetivos VT 7

- Diseño orientado a objetos: UML

UML

- ***El lenguaje de modelado unificado (UML) es un lenguaje gráfico para visualizar, especificar y documentar el software.***
- ***Tipos de diagramas:***
  - **Diagramas de estructura (parte estática del modelo):** incluyen el diagrama de clases, diagrama de objetos, diagrama de componentes, diagrama de estructura compuesta, diagrama de paquetes y diagrama de implementación o despliegue.
  - **Diagramas dinámicos**: que explican la estructura dinámica del sistema y muestran las interacciones entre los objetos del sistema.
    - **Diagramas de comportamiento:** incluyen el diagrama de casos de uso, diagrama de comunicación y diagrama de estado.
    - **Diagramas de interacción:** Se centran en el flujo de control y de datos entre los elementos del sistema modelado

Diagramas de clase

- ***Entendemos por diagrama de clases una representación gráfica y estática de la estructura general de un sistema, mostrando cada una de las clases y sus interacciones representadas en forma de bloques.***

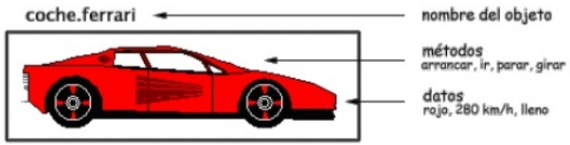
- Una clase es una plantilla para la creación de objetos de datos según un modelo predefinido. Las clases se utilizan para representar entidades o conceptos, como los sustantivos en el lenguaje.

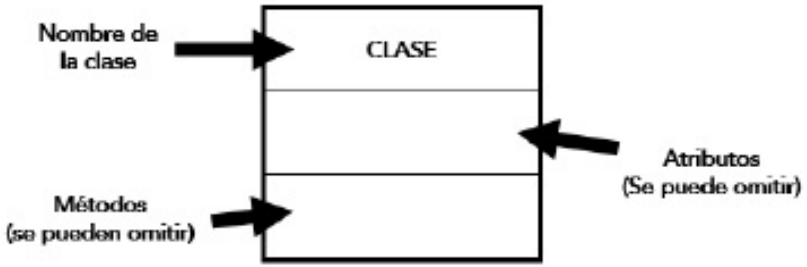
- Cada clase es un modelo que define un conjunto de variables -el estado, y métodos apropiados para operar con dichos datos -el comportamiento. Cada objeto creado a partir de la clase se denomina instancia de la clase

Clases y objetos

- **Clase**: *Coche*

- **Objeto**: *Ferrari*





ILERNA Online

Accesibilidad de clases

Estado	Accesible desde la propia clase	Accesible desde una clase derivada (herencia)	Accesible desde fuera de la clase
Public	SI	SI	SI
Protected	SI	SI	NO
Private	SI	NO	NO

Accesibilidad de clases: protected

```
class Metodo extends Suma
{
   public static void main(String args[])
   {
      Suma obj = new Suma();
      System.out.println(obj.SumaDeNumeros(5, 2));
}
}
```

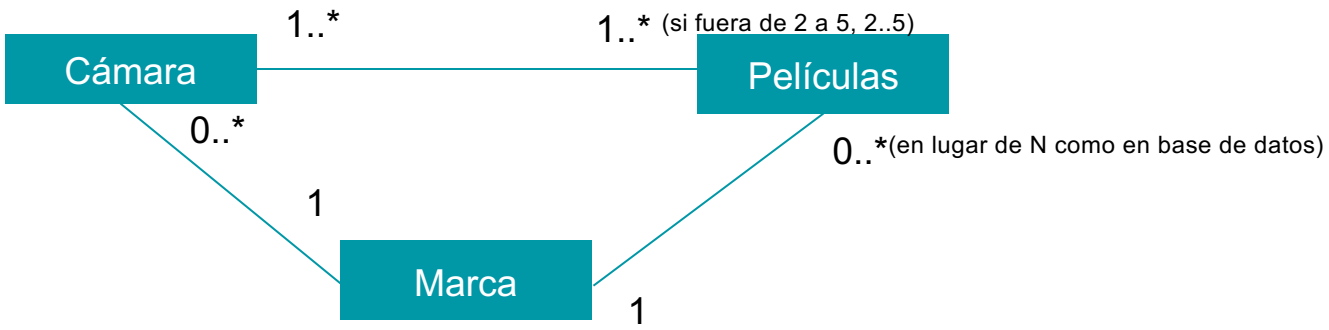
```
public class Suma {

protected int SumaDeNumeros(int a, int b)

{ return a+b; }

}
```

ILERNA Online

Relaciones en UML: Asociación y cardinalidad

Representación de una tienda de alquiler de cámaras fotográficas:

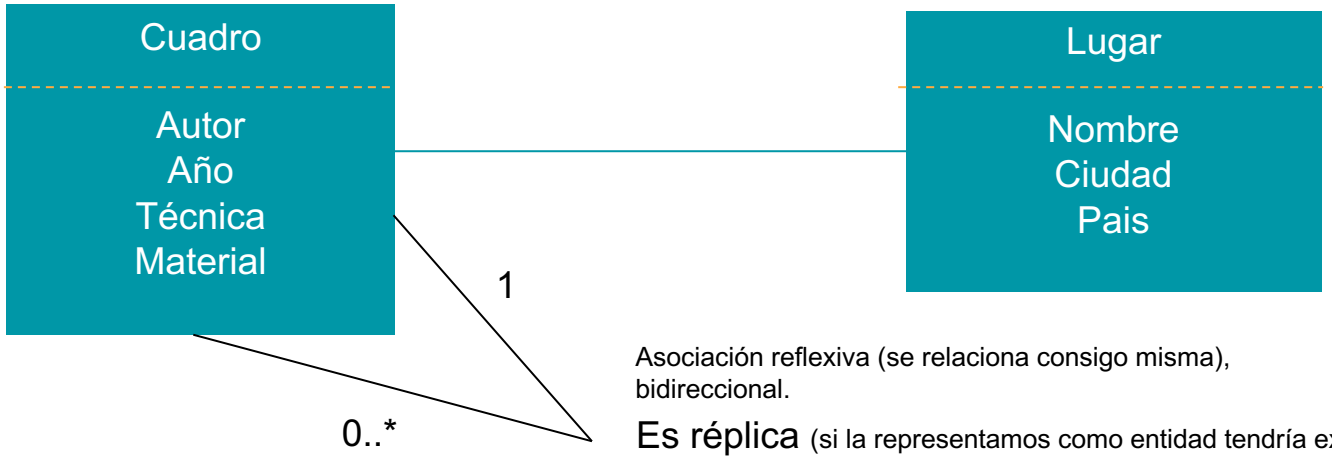
- La tienda alquila cámaras fotográficas analógicas.
- Las cámaras se caracterizan por su marca, modelo y soporte flash (si, no).
- Cada cámara es compatible con uno o más tipos de películas.
- Las películas se caracterizan por su marca, nombre, sensibilidad ISO (50, 100,200, 400, 800, 1600) y formato (35mm, 110mm, 120mm)
- Para cada marca con la que trabaja la tienda se conoce la dirección del servicio de reparación más cercano.



Relaciones en UML: Asociación reflexiva

Representación de un cuadro de un museo

- Autor: Anónimo
- Adscripción cronológica: 1503
- Técnica: óleo
- Material de soporte: madera de nogal
- Descripción: Existen muchas réplicas o copias de la obra la cual se encuentra en el Museo del Prado



¿Y la réplica?

Asociación reflexiva (se relaciona consigo misma), bidireccional.

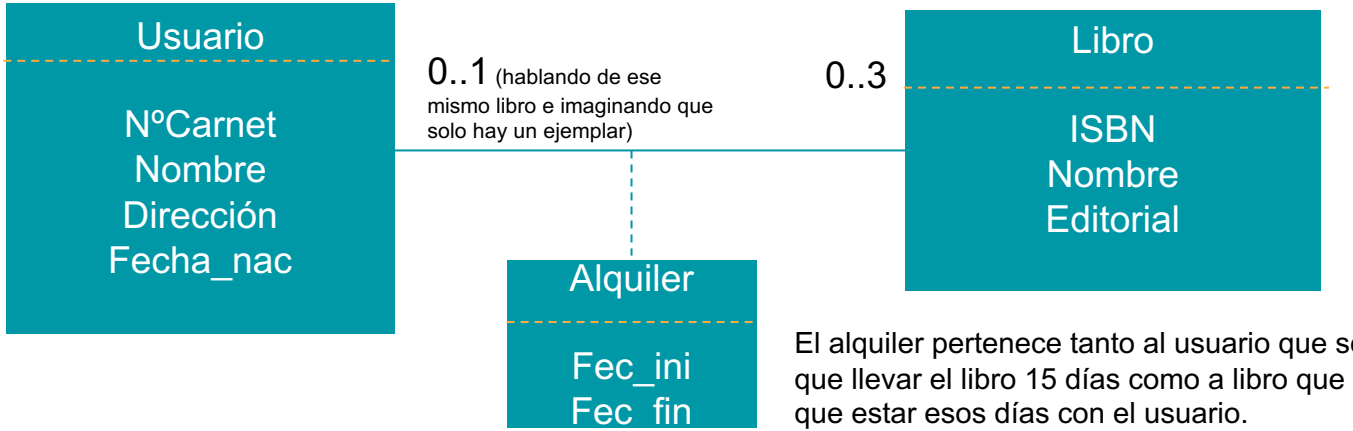
Es réplica (si la representamos como entidad tendría exactamente las mismas características que tiene Cuadro, sería redundante).

ILERNA Online

Relaciones en UML: Clase asociación (es una clase que aparece cuando tenemos que representar información adicional que no pertenece a ninguna de las otras dos). Línea discontinua.

Representación de un préstamo de un libro en una biblioteca

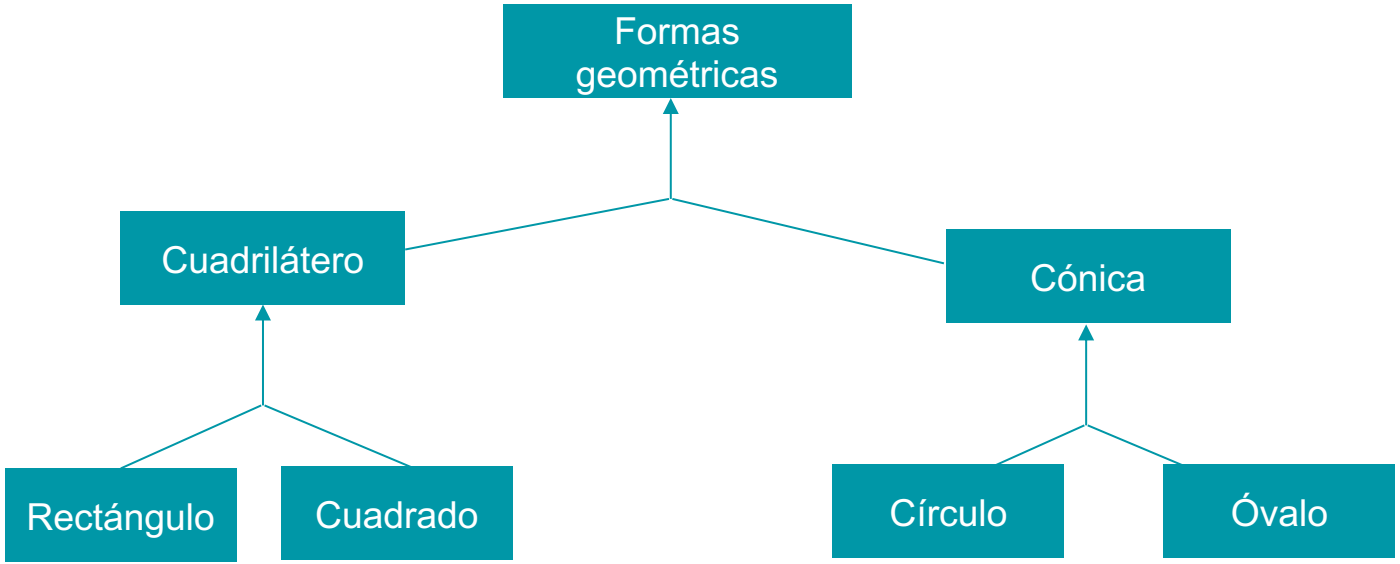
- Un usuario registrado de la biblioteca puede realizar el préstamos de como máximo 3 libros
- Cómo máximo el préstamo será de 15 días



El alquiler pertenece tanto al usuario que se tiene que llevar el libro 15 días como a libro que tiene que estar esos días con el usuario.

Relaciones en UML: Herencia (generalización, porque vamos desde abajo hacia arriba. Especialización si es al revés.)

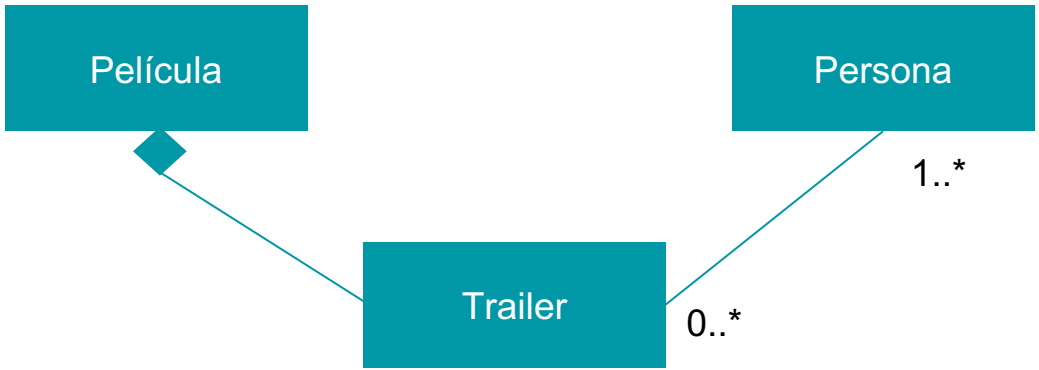
Representa mediante un diagrama las siguientes formas geométricas:



Relaciones en UML: Composición (asociación fuerte)

Representación de películas:

- De cada película se almacena el título, la sinopsis, el año, el género al que pertenece (drama, comedia, acción, terror, romance, aventura, SFI) y el país.
- Una película puede tener asociados varios trailers que son editados por una o más personas.
- El tráiler está compuesto de partes de película y está editado por persona.



Relaciones en UML: Agregación (asociación débil)

Representación de series:

- Las series se caracterizan por su título, año de inicio, sinopsis y género al que pertenece (acción, aventura, animación, comedia, documental, drama, horror, musical, romance, ciencia ficción)
- Las series se organizan en temporadas ordenadas que tienen una fecha de producción y una fecha de estreno de televisión a nivel mundial.
- Cada temporada está a su vez formada por capítulos ordenados que tienen un título, una duración y una sinopsis.
- Se puede ver un capítulo sin ver toda la temporada o una temporada sin ver toda la serie.

Relaciones en UML: Dependencia

Representación de una impresora:

- Una impresora quiere imprimir un documento determinado.

- Características:
  - La clase *IMPRESORA* usa a la clase *DOCUMENTO*.
  - La clase IMPRESORA depende de *DOCUMENTO*
  - Dada la dependencia, todo cambio en *DOCUMENTO* podrá afectar a IMPRESORA
  - La *IMPRESORA* conoce la existencia de *DOCUMENTO* pero al revés, no
  - Es una asociación unidireccional

¿Probamos un ejercicio?

- Queremos representar datos acerca de una persona.
- Cada persona tiene un nombre, unos apellidos y un titulo nobiliario que pueden cambiar a lo largo de su vida.
- Además, cada persona tiene una fecha y un lugar de nacimiento y de fallecimiento.
- Una persona puede tener diferentes ocupaciones, entrar en contacto con otras personas, y visitar lugares en diferentes periodos de su vida.
- También es interesante tomar nota de los eventos en los que participa.

¿Qué representar?

- Diagrama de clases
- Relaciones entre ellas
- Atributos de las clase
- Cardinalidad

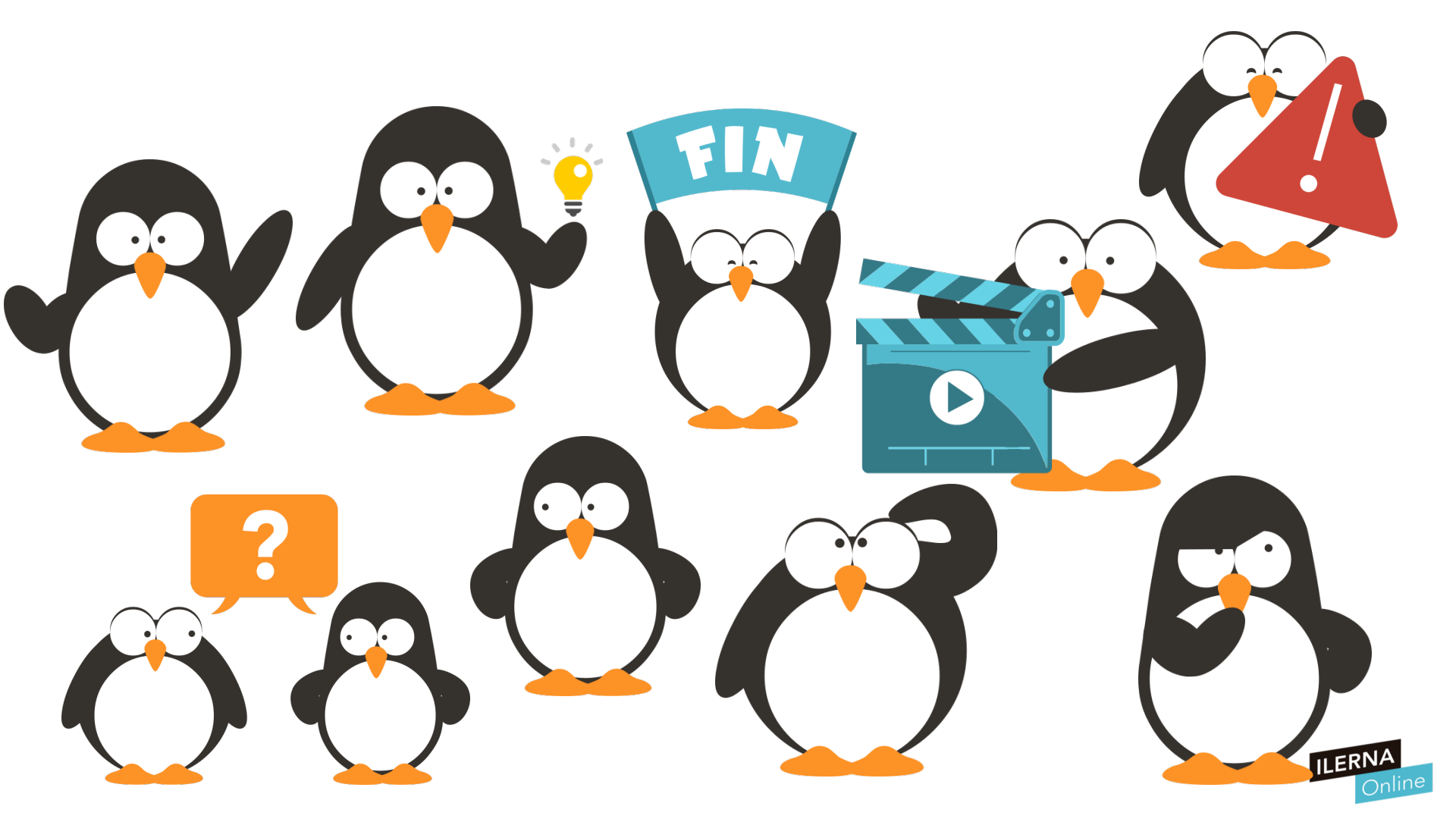
Programas de diseño de OOP

- **Magic Draw**: https://www.nomagic.com/products/magicdraw

- **Papyrus UML**: El entorno de modelado de Eclipse. Gratuito y open source
  https://www.eclipse.org/papyrus/

- **Modelio**: De open source al que se le pueden añadir funcionalidades mediante un sistema de extensión modular.
  https://www.modelio.org/

- **ArgoUML**: La más antigua. http://argouml.tigris.org/
- **StarUML**: Herramienta referenciada por Grady Booch: http://staruml.io/


FIN
?
ILERNA
Online