Programación Orientada a Objetos

SISTEMAS UNI

Introducción a la Programación Orientada a Objetos (POO)

- La POO reúne un conjunto de técnicas para obtener la calidad interna como medio para obtener la calidad externa (reutilización y extensibilidad), se basa en dividir un programa en pequeñas unidades lógicas de código. A estas pequeñas unidades lógicas de código se les llama objetos.
- La POO tratan a las aplicaciones como conjuntos de objetos que se ayudan entre si para realizar acciones, permitiendo que los programas sean mas fáciles de escribir, mantener y reutilizar en el tiempo.
- El software se organiza como una colección de objetos que contienen tanto estructura como comportamiento, por lo tanto un programa orientado a objetos es una colección de clases, que luego será invocado para el uso de sus contenidos de diferentes formas mediante la creación de un objeto.
- La POO se fue convirtiendo en el estilo de programación dominante a mediados de los años ochenta, en gran parte debido a la influencia de C++, una extensión del lenguaje de programación C. Su dominación fue consolidada al auge de las interfaces graficas de usuario (GUI), para las cuales la POO esta particularmente bien adaptada.

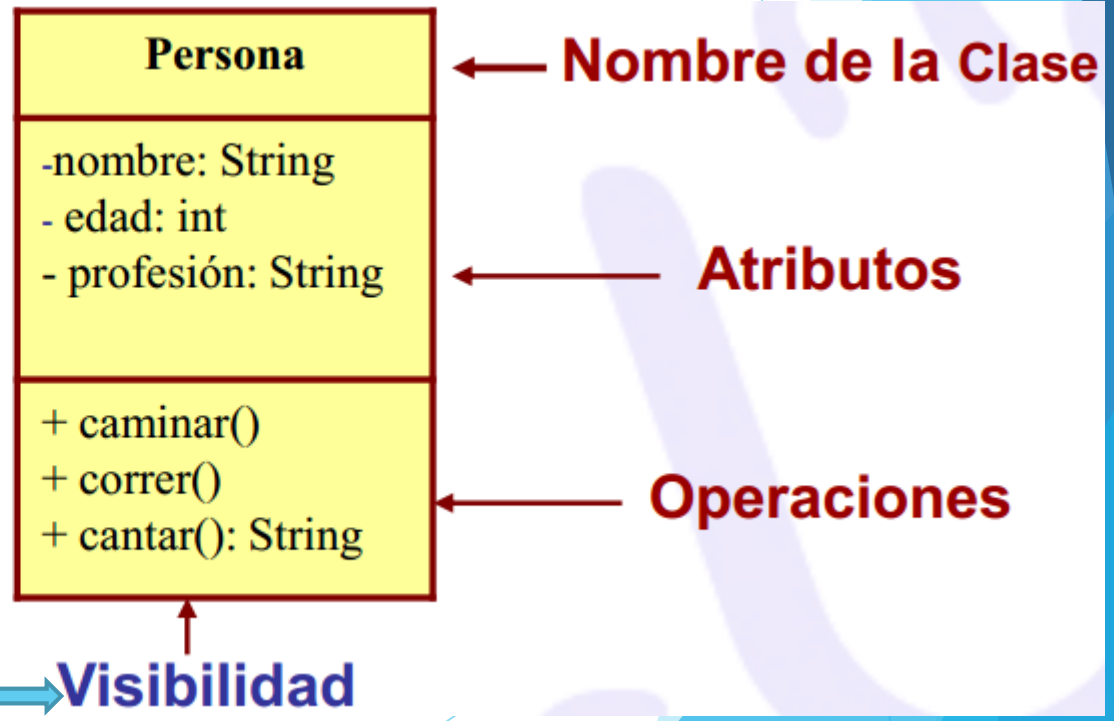
Características de la POO

- Clase :
  - Es la clasificación de las características y comportamientos comunes de objetos del mismo tipo.
- En la POO se dice que es la plantilla genérica para un conjunto de objetos con las misma características.

NOTACIÓN UML

CLASE

Calificación



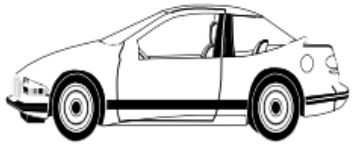
+ Publico: Acceso interior y exterior, Instancia (objeto) y herencia

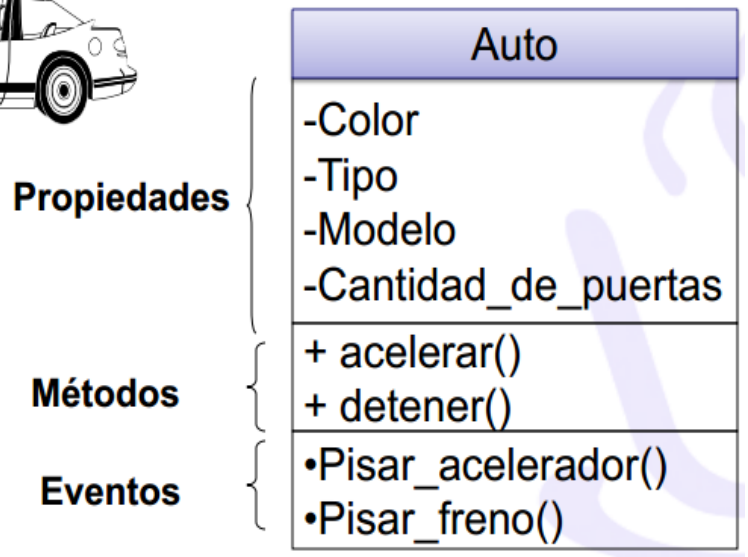
-Privado: Acceso interior, sólo dentro de la clase

-# Protegido: Acceso interior y exterior sólo por herencia

Clase

- Una clase es como una plantilla de la cual se pueden crear varios objetos con las mismas características y funciones.



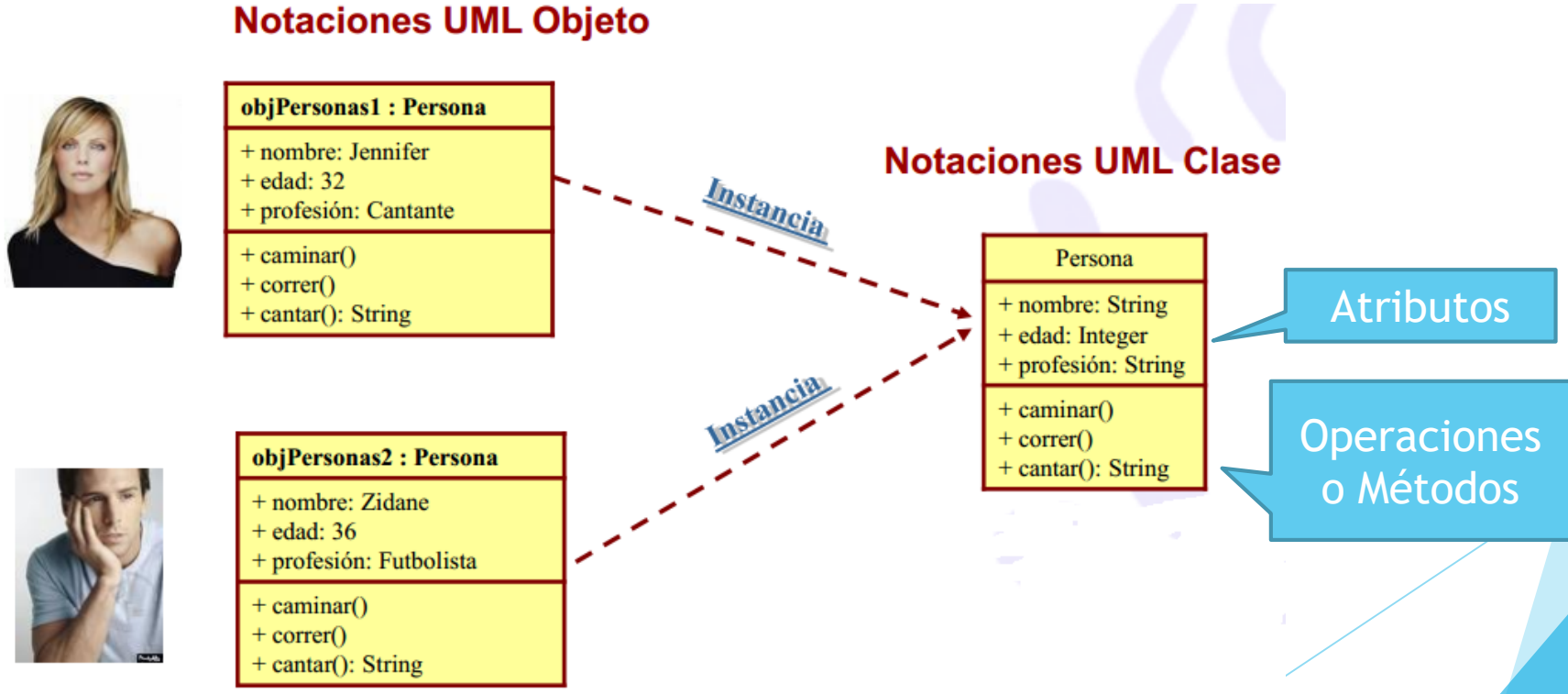



```
public class Auto
  {
      public string color;
      public int tipo;
      public string modelo;
      public int cantida_puertas;

      void acelerar()
      {
      }
      void detener()
      {
      }
  }
```

Características de la POO

- Objeto:
  - Es la representación concreta y detallada de algo en particular, tal representación determina su identidad (nombre único para distinguir un objeto de otro) su estado(conjunto de valores que caracterizan al objeto en un momento dado) y su comportamiento(conjunto de funciones que el objeto puede llevar a cabo)
- Los objetos son instancias de clases (Una instancia es un objeto)



Características de la POO

Clases y objetos

Veamos los siguientes ejemplos de clases y objetos

- Los objetos que existen en el mundo real pueden ser conceptuales o físicos. Este es un ejemplo de objetos físicos.

Clases en C#

En C# un programa se crea mediante la definición de nuevos tipos, de hecho C# todos los tipos son clases. Cada nuevo tipo esta formado por un conjunto de datos y funciones.

```
using System;
      // declaración de la clase
class Persona
{  // una variable que contendrá el nombre
       string nombre;
  // constructor: como inicializar un objeto de clase Persona
  public Persona(string nombre)
  {
    this.nombre = nombre; // this: una forma de acceder a este objeto
  }
  // propiedad: una forma de acceder al nombre de una persona
  public string Nombre
  {
    get { return nombre; }
    set { nombre = value; } // variable especial value
  }
  public static void Main()
  {
    Persona p = new Persona("Adán");
    Console.WriteLine("p.Nombre = {0}", p.Nombre);
    p.Nombre = "Eva";
    Console.WriteLine("p.Nombre = {0}", p.Nombre);
  }
}
```

Propiedades, métodos, constructores y destructores

- Propiedades
  - Los campos y propiedades representan información que contiene un objeto, los campos se aparecen a las variables ya que se pueden leer y establecer directamente.
    - Definiendo un campo
      - Class médicos {
        - public string sexo;
      - }
    - Las propiedades tienen procedimientos get y set, que proporcionan un mayor control sobre la forma en que se establecen o devuelven los valores.
    - Propiedad como procedimiento
      - private string código;
      - public string Codigo
      - {
        - get { return código;}
        - set { código = value}
      - }

Propiedad de implementación automática

public string Codigo {get; set;}

Propiedades, métodos, constructores y destructores

- Métodos :
  - Un método es una función o procedimiento definida en el interior de una clase. En C# todo el código se ejecuta como un método de alguna clase, pues todos los tipos que podemos usar son clases, e incluso, la funcion Main, es en realidad un método de alguna clase.
  - Class medicos {
  - public string buscarmedicoporcodigo(string código)

    {

    //método que retorna un valor => también llamado función.

    return valor;

    }

    public void registrarMedico(string código, string nombres, string apellidos,

    string especialidad, datetime fechaingreso)

    {

    //método que ejecuta una acción y que no retorna ningún valor => también

    // procedimiento

    }
  - }
  - Una clase puede tener varias implementaciones o sobrecargas del mismo método que se diferencian en el numero de parámetros o de tipos de parámetro.
  - public string buscarestadodelmedico(string código){ //código }
  - public string buscarestadodelmedico(string código, string especialidad){ //código }

Propiedades, métodos, constructores y destructores

- Constructores ;
  - Son un tipo especial de métodos que poseen las clases y cuya finalidad es inicializar los campos de un objeto. Los constructores tienen el mismo nombre que la clase y por lo general se usa para inicializar los miembros de datos del nuevo objeto.
  - public class **clsClientes**
  - {
    - private string codigo;
    - Private int teléfono;
    - Private string email;
    - public **clsClientes(string** xcódigo, int xteléfono,string xemail)
    - {
      - this.codigo =xcodigo;
      - This.telefono = xtelefono;
      - This.email= xemail;
    - }
  - }

Propiedades, métodos, constructores y destructores

- Destructores :
  - Los destructores son el equivalente de los constructores pero a la hora de deshacernos de la instancia de un objeto. Cada vez que el recolector de basura de C# decide eliminar un objeto de memoria, antes, ejecuta este método, no se puede sobrecargar porque es único, su nombre ha de estar formado por el símbolo ~ seguido del nombre de la clase.
  - class Persona
  - {
    - public Persona (parámetros …) { código del constructor ---}
    - ~ Persona () { …. Codigo del constructor … }
  - }
- La palabra clave this
  - Esta palabra nos permite referirnos a cualquier variable o método de la instancia de una clase en la que nos encontremos, también nos permite llamar a un constructor sobrecargado, un uso muy común de la misma es distinguir entre un campo de una instancia de un objeto y un parámetro.
  - public Persona(string nombre)
  - {
    - this.nombre=nombre;
  - }