



Diplomado Instalación de Programas Informáticos

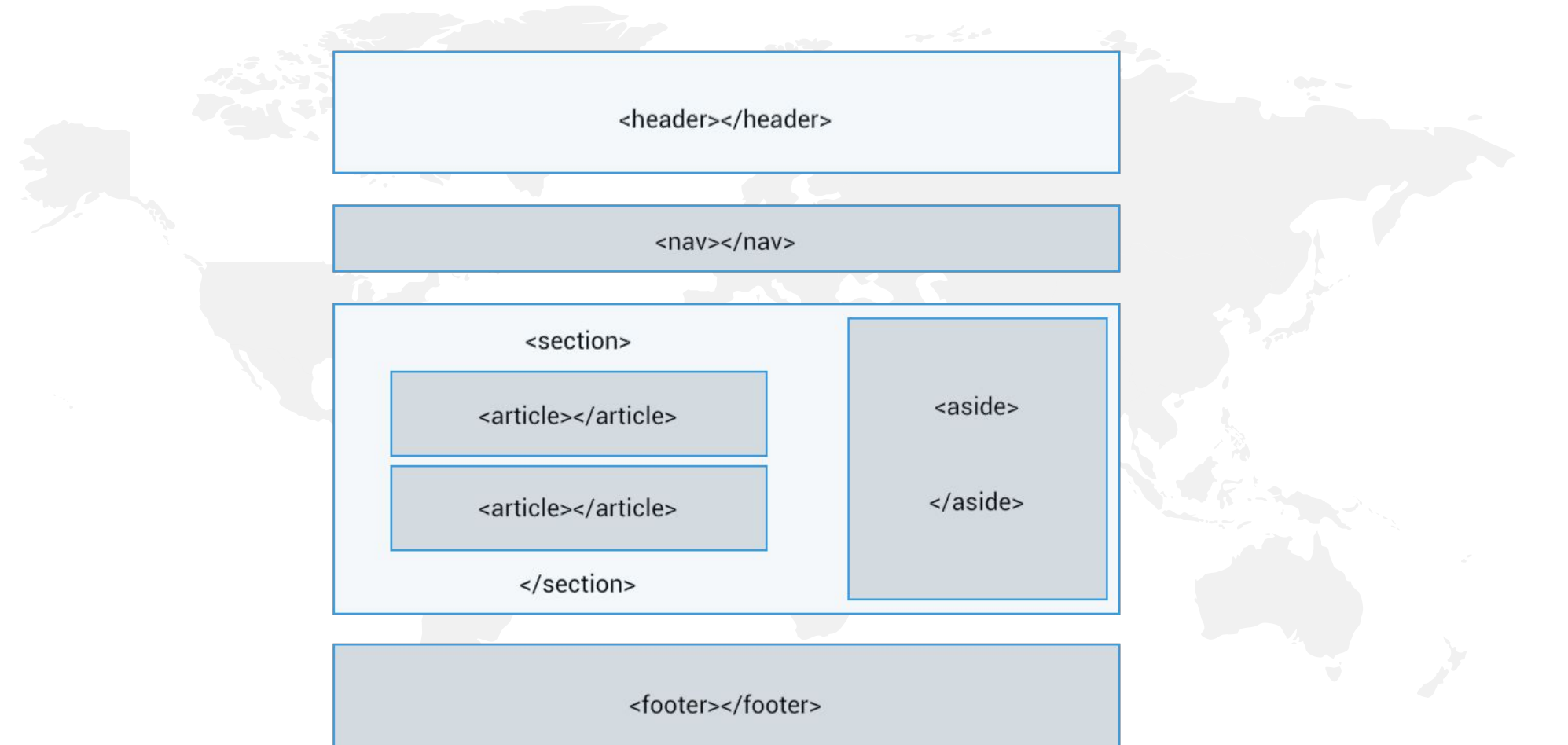
**Jhon Anderson Hoyos
Ingeniero de Sistemas**

Herramientas





ESTRUCTURA HTML



```
graph TD; header[<header></header>] --- nav[<nav></nav>]; nav --- section[<section>]; section --- article1[<article></article>]; section --- article2[<article></article>]; section --- aside[<aside><br></aside>]; section --- footer[<footer></footer>];
```

<header></header>

<nav></nav>

<section>

<article></article>

<article></article>

</section>

<aside>

</aside>

<footer></footer>

es utilizado para indicar el **idioma del contenido** del documento html

Estructura Html

cabecera del documento
html

Cuerpo del documento
html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
    rel="stylesheet" integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rUkQ2Bzep5IDxbcnCeu0xjzrPF/et3URy9Bv1WTRi"
    crossorigin="anonymous">

    <title>Document</title>
  </head>
  <body>

    <div>
      <button type="button" class="btn btn-primary">Primary</button>
    </div>

  </body>
</html>
```

informa al navegador que versión de HTML (o XML) se usó para escribir el documento. Doctype es una declaración no una etiqueta.

Titulo de la
página

Librería Utilizada

elementos
de la página



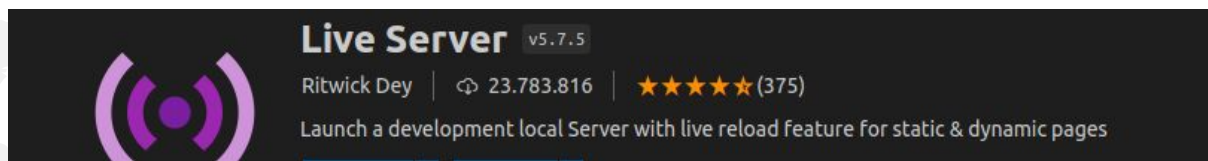
¿Qué es una librería?

- Una librería es uno o varios archivos escritos en un lenguaje de programación determinado, que proporcionan diversas funcionalidades. A diferencia de un framework, una librería no aporta la estructura sobre cómo realizar el desarrollo, sino que proporciona funcionalidades comunes, que ya han sido resueltas previamente por otros programadores y evitan la duplicidad de código. Además reducen el tiempo de desarrollo y aumentan la calidad del mismo.



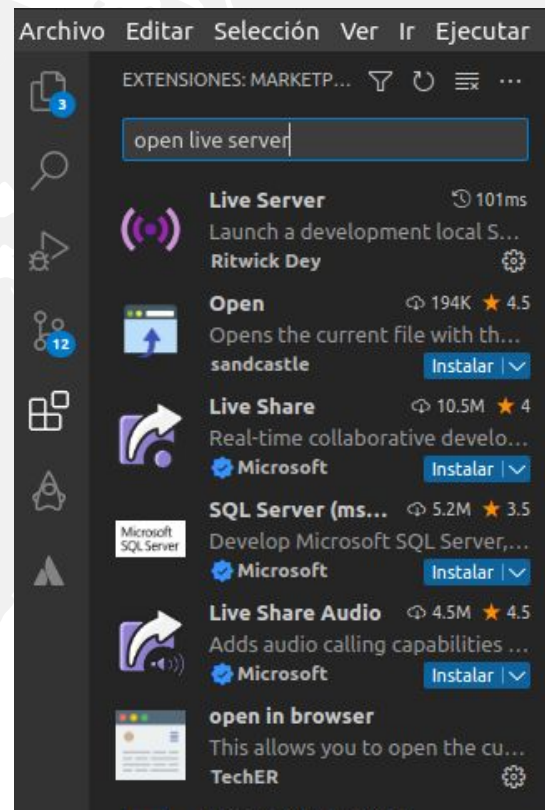
Subiendo el proyecto desde visual studio code:

Librería live server.



Instalación desde visual studio code:

Extensiones(ctrl+mayud+x):





¿Qué es un Framework?

Un framework es un conjunto de archivos y pautas que definen la estructura y metodología, sobre cómo hacer el desarrollo de un proyecto software. Se podría decir que es una guía o esquema que nos ayuda a programar de forma sencilla y rápida.





Bootstrap: *el framework más popular del mundo para crear sitios adaptables para dispositivos móviles, con jsDelivr y con una plantilla como página de inicio.*

Recurso: <https://getbootstrap.esdocu.com/docs/5.1/getting-started/introduction/>

Instalación:

CDN via jsDelivr

Omite la descarga con [jsDelivr](#) para entregar la versión en caché del CSS y JS compilados de Bootstrap a tu proyecto.

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet">
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ec338846365446b0d35b56bf6cdd9811d639f8dd1383999256be31f7995d3779" crossorigin="anonymous"></script>
```

Si estás utilizando nuestro JavaScript compilado y prefieres incluir Popper por separado, agrega Popper antes de nuestro JS, preferiblemente a través de un CDN.

```
<script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.2/dist/umd/popper.min.js" integrity="sha384-8vBTkDj2naZ14VeVGeDjm9LLsYVDWNTyUXpxY0JXmvhYtGf6Cd2b+5a5hd9v1" crossorigin="anonymous"></script>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.min.js" integrity="sha384-ec338846365446b0d35b56bf6cdd9811d639f8dd1383999256be31f7995d3779" crossorigin="anonymous"></script>
```

```
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet"
  integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeu0XjzrPF/et3URy9Bv1WTRi" crossorigin="anonymous">
  <title>Document</title>
</head>
```



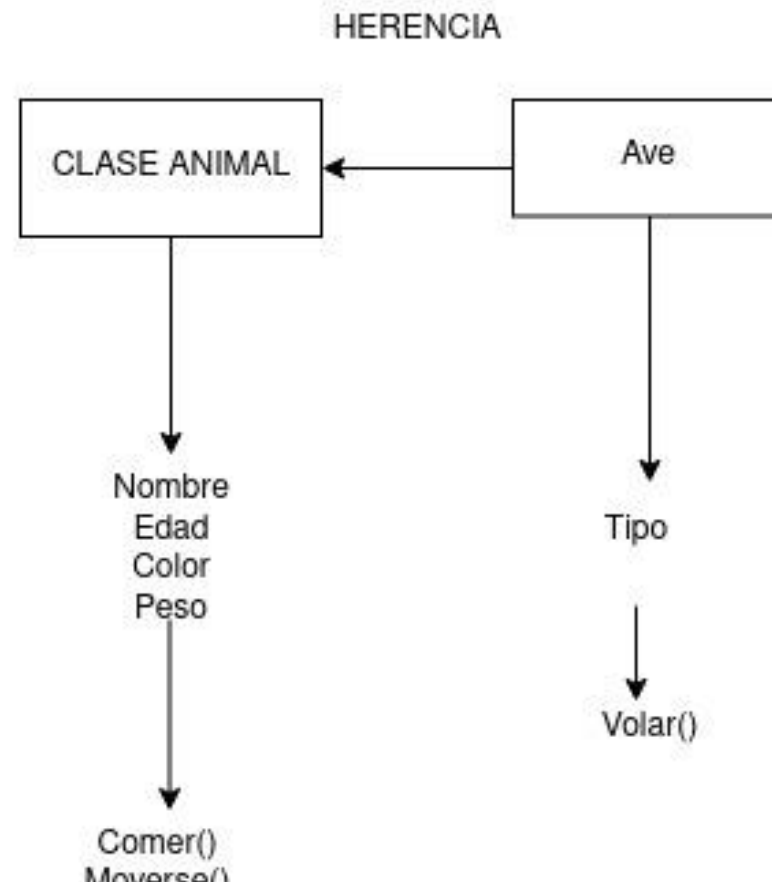
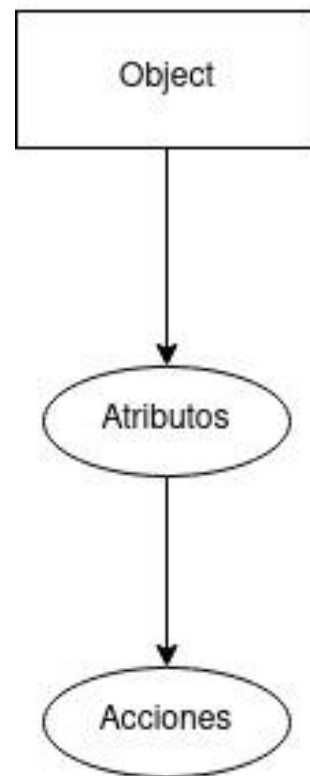

Codificación de caracteres:

UTF 8: **UTF-8** es un formato estándar **para** almacenar caracteres Unicode. html está en formato **UTF-8**. **UTF-8** utiliza una secuencia única **de** 1, 2, 3 o 4 bytes **para** codificar cada carácter en el juego **de** caracteres Unicode.

¿Qué es UNICODE?

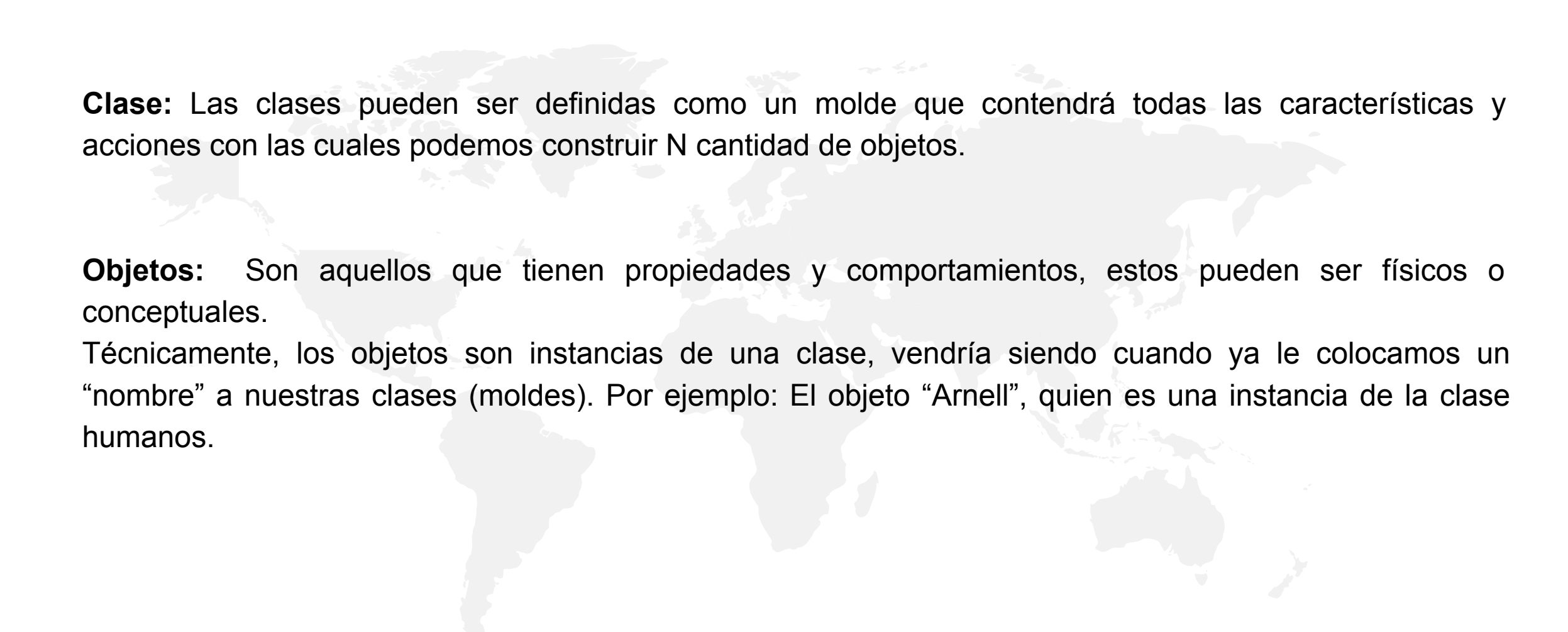
Unicode se creó para solucionar el problema ocasionado por la profusión de juegos de códigos. Desde el principio de la programación de sistemas se han desarrollado cientos de codificaciones, cada una de ellas para pequeños grupos de idiomas y con fines específicos. Como resultado, la interpretación del texto, la entrada, la clasificación, la visualización y el almacenamiento depende del conocimiento de los distintos tipos de juegos de caracteres y sus codificaciones. Se escriben programas para manejar una sola codificación cada vez y cambiar entre ellas, o para convertir las codificaciones entre externas e internas.

Programación Orientada a Objetos





Programación Orientada a Objetos: Elementos básicos



Clase: Las clases pueden ser definidas como un molde que contendrá todas las características y acciones con las cuales podemos construir N cantidad de objetos.

Objetos: Son aquellos que tienen propiedades y comportamientos, estos pueden ser físicos o conceptuales.

Técnicamente, los objetos son instancias de una clase, vendría siendo cuando ya le colocamos un “nombre” a nuestras clases (molde). Por ejemplo: El objeto “Arnell”, quien es una instancia de la clase humanos.

Instanciamiento de una clase

```
class Rectangulo {  
    constructor (alto, ancho) {  
        this.alto = alto;  
        this.ancho = ancho;  
    }  
    // Getter  
    get area() {  
        return this.calcArea();  
    }  
    // Método  
    calcArea () {  
        return this.alto * this.ancho;  
    }  
}  
  
const cuadrado = new Rectangulo(10, 10);  
  
console.log(cuadrado.area); // 100
```

```
1  public class Person  
2  {  
3      // Properties  
4      public int Id { get; private set; }  
5      public string FirstName { get; set; }  
6      public string LastName { get; set; }  
7      public DateTime DateOfBirth { get; set; }  
8      public char Sex { get; set; }  
9  
10     // Constructors  
11     public Person()  
12     {  
13  
14     }  
15  
16     public Person(int id, string firstName, string lastName, DateTime dateOfBirth, char sex)  
17     {  
18         this.Id = id;  
19         this.FirstName = firstName;  
20         this.LastName = lastName;  
21         this.DateOfBirth = dateOfBirth;  
22         this.Sex = sex;  
23     }  
24  
25     // Methods - Publicos  
26  
27     // Mthos - Privates
```



Atributos, propiedades y Métodos:

Atributos: Los **atributos** son características o rasgos que describen una persona, una organización, un lugar o un elemento.

Los Setters & Getters son métodos de acceso lo que indica que son siempre declarados públicos, y nos sirven para dos cosas:

Setters: Del Inglés Set, que significa establecer, pues nos sirve para asignar un valor inicial a un atributo, pero de forma explícita, además el Setter nunca retorna nada (Siempre es void), y solo nos permite dar acceso público a ciertos atributos que deseemos el usuario pueda modificar.

Getters: Del Inglés Get, que significa obtener, pues nos sirve para obtener (recuperar o acceder) el valor ya asignado a un atributo y utilizarlo para cierto método.



Constructores: nos permite asignar valores por defecto a los campos o llamar a otro constructor.

Un **constructor** es un elemento de una clase cuyo identificador coincide con el de la clase correspondiente y que tiene por objetivo obligar a y controlar cómo se inicializa una instancia de una determinada clase, ya que el lenguaje **Java** no permite que las variables miembro de una nueva instancia queden sin inicializar.

Métodos en JavaScript

En la programación orientada a objetos, las propiedades y los métodos se dividen en dos grupos:

- *privado* – métodos y propiedades, accesibles desde otros métodos de la clase, pero no desde el exterior.
- *público*– métodos y propiedades, accesibles también desde fuera de la clase.



Funciones: Una función en JavaScript es similar a un procedimiento — un conjunto de instrucciones que realiza una tarea o calcula un valor, pero para que un procedimiento califique como función, debe tomar alguna entrada y devolver una salida donde hay alguna relación obvia entre la entrada y la salida. Para usar una función, se debe definir en algún lugar del ámbito que se desee llamar.

```
function square(number) {  
  return number * number;  
}
```

```
function myFunc(theObject) {  
  theObject.make = 'Toyota';  
}  
  
[parcial]var mycar = { make: 'Honda', model: 'Accord', year: 1998 };  
var x, y;  
  
x = mycar.make; // x obtiene el valor "Honda"  
  
myFunc(mycar);  
y = mycar.make; // y obtiene el valor "Toyota"  
                // (la propiedad make fue cambiada por la función)
```



Comentarios:

Un comentario es **texto que el compilador omite pero que es útil para facilitar la lectura del código a los desarrolladores de software**. Los comentarios se usan normalmente para anotar código para su referencia futura. El compilador los trata como si fueran espacios en blanco

El **comentario** deberá estar estructurado en tres partes: Título: “**Comentario** del artículo ‘Añádase el nombre’ a cargo de ‘Añádase el nombre del autor’”. Introducción: una breve descripción acerca del argumento sobre el que versa el **comentario**

// taquí va el texto del comentario// => una línea

/*aquí va el texto del comentario*/ => varias líneas

<!-- aqui va el texto del comentario --> => comentarios en html



Recursos:

Google Icons: <https://fonts.google.com/icons>

Visualización de iconos en visual studio code:

wSchools: https://www.w3schools.com/html/html_basic.asp

Documentación bootstrap para pie de página:: <https://mdbootstrap.com/docs/standard/navigation/footer/#!>



Referencias

<https://altruistas.org/la-herencia-en-la-programacion-orientada-a-objetos-poo/>

<https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Classes>

<https://lenguajejs.com/javascript/oop/clases/>

<https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Functions>

Elementos básicos de POO:

https://platzi.com/tutoriales/1474-oop/6108-4-elementos-y-pilares-basicos-de-la-programacion-orientada-a-objetos-poo/?utm_source=google&utm_medium=cpc&utm_campaign=17739691128&utm_adgroup=&utm_content=&gclid=Cj0KCQiAsdKbBhDHARIsANJ6-jdiVe5NfKIZ9t5i_sTbf8mkXPLpqSJNH9DA8spmVcVeNYeJDD6auQVgaAInSEALw_wcB&gclidsrc=aw.ds

<https://es.javascript.info/private-protected-properties-methods>