

MEMORIA LENGUAJE DE MARCAS



- **DAVID GARZÓN** - Pág. HOME y memoria HOME
- **IVÁN ENCINAS** - Pág. HOME y Memoria HOME
- **SERGIO FELIPE** - Pág XML y la memoria del xml .
- **DAVID DEL PILAR** - Pág JSON y memoria JSON

ÍNDICE

XML Historia	3
Normas De Uso y Sintaxis De XML	3
Todos los elementos XML deben estar debidamente anidados	4
Todos los documentos XML deben tener un elemento raíz	4
Los valores de atributo siempre deben estar entre comillas	5
Aplicaciones del XML en el ámbito empresarial.	6
Bloques y su adaptación responsive.	7
Agregar Bootstrap al proyecto	7
Diseño de los bloques	8
Container	8
Header	9
Barra de navegación	9
Contenido	10
Implementación de grid Css	13
Historia sobre JSON	16
Reglas de uso y de sintaxis JSON	16
Aplicaciones en JSON en el ámbito empresarial	18
WEB home:	19

XML Historia

La historia de XML comienza en 1996 cuando el World Wide Web Consortium (W3C) lanzó un proyecto para crear un lenguaje de marcado extensible. La idea era tener un lenguaje que permitiera describir los datos en una forma estructurada y que fuera fácilmente interpretable por máquinas y humanos.

En 1998, el W3C publicó la primera versión de la especificación de XML 1.0, que se convirtió rápidamente en un estándar de la industria para la representación y transferencia de datos en la Web. Desde entonces, se han lanzado varias versiones y extensiones de XML, incluyendo XML Schema, XSLT y XPath, que han ampliado la funcionalidad de XML y lo han convertido en una herramienta indispensable para la integración de aplicaciones y la gestión de datos en la Web.

Hoy en día, XML es uno de los lenguajes de marcado más utilizados en la industria, y se utiliza en una amplia variedad de aplicaciones, incluyendo la gestión de contenidos, el intercambio de datos entre aplicaciones, la representación de datos en la Web y la integración de aplicaciones. Su popularidad se debe a su flexibilidad, escalabilidad y capacidad para describir los datos en una forma estructurada y fácilmente interpretable.

Normas De Uso y Sintaxis De XML

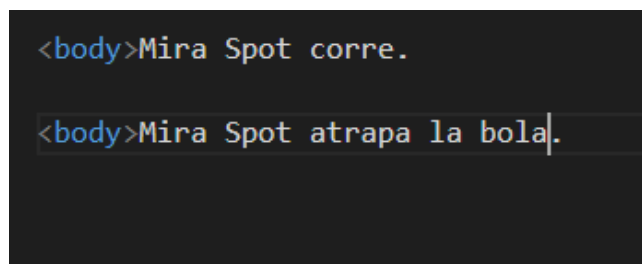
Las reglas que debemos seguir para crea la sintaxis de XML son:

1. **Todos los elementos XML deben tener una etiqueta de cierre.**
2. **Las etiquetas XML son sensibles a mayúsculas y minúsculas.**
3. **Todos los elementos XML deben estar debidamente anidados.**
4. **Todos los documentos XML deben tener un elemento raíz.**
5. **Los valores de atributo siempre deben estar entre comillas.**

Todos los elementos XML deben tener una etiqueta de cierre

No está permitido omitir la etiqueta de cierre al crear la sintaxis XML. Los elementos XML deben tener una etiqueta de cierre.

Forma incorrecta:



```
<body>Mira Spot corre.  
<body>Mira Spot atrapa la bola|.
```

Forma correcta:

```
<body>Mira Spot corre.</body>  
<body>Mira Spot atrapa la bola.</body>|
```

Las etiquetas XML son sensibles a mayúsculas y minúsculas

Cuando creamos documentos XML , la etiqueta < Body > con la b en mayúscula es diferente de la etiqueta < body> en minúscula.

Forma Incorrecta:

```
<Body>Mira Spot corre.</body>
```

Forma Correcta:

```
<body>Mira Spot corre</body>
```

Todos los elementos XML deben estar debidamente anidados

Un anidamiento incorrecto de las etiquetas no tiene sentido para XML.

Forma Incorrecta:

```
<b><i>Este texto está en negrita y cursiva.</b></i>
```

Forma Correcta:

```
<b><i>Este texto está en negrita y cursiva.</i></b>
```

Todos los documentos XML deben tener un elemento raíz

Todos los documentos XML deben contener un par de etiquetas individuales para definir un elemento raíz. Todos los demás elementos deben estar dentro de este elemento raíz, todos los elementos pueden tener subelementos (elementos hijo). Los subelementos deben estar anidados correctamente dentro de su elemento padre.

Ejemplo:

```
<padre>
|  <hijo>
|    <subhijo>.....</subhijo>|
|  </hijo>
|</padre>
```

Los valores de atributo siempre deben estar entre comillas

No está permitido omitir las comillas alrededor de los valores de atributos. Los elementos XML pueden tener atributos en pares nombre/valor, sin embargo, el valor de atributo siempre debe estar entre comillas.

Forma Incorrecta:

```
<?xml version="1.0" encoding="UTF-8"?>

<note date=05/05/05>
|  <to>Dick</to>|
|  <from>Jane</from>
|</note>
```

Forma Correcta :

```
<?xml version="1.0" encoding="UTF-8"?>

|
|<note date="05/05/05">
|  <to>Dick</to>
|  <from>Jane</from>
|</note>
```

Aplicaciones del XML en el ámbito empresarial.

XML es ampliamente utilizado en el ámbito empresarial para intercambiar información entre diferentes sistemas y plataformas de manera estructurada y organizada. Algunas de las aplicaciones más comunes de XML en el ámbito empresarial incluyen:

1. **Integración de sistemas:** XML permite integrar diferentes sistemas y aplicaciones de manera eficiente, al proporcionar un formato estandarizado para la transferencia de datos.
2. **Documentación de procesos de negocio:** XML se utiliza para documentar los procesos de negocio de una empresa y para modelar y representar los flujos de trabajo de una forma clara y fácil de entender.
3. **Comercio electrónico:** XML se utiliza en la transmisión de información relacionada con el comercio electrónico, como facturas, órdenes de compra y cualquier otra información relevante para los negocios.
4. **Intercambio de información en tiempo real:** XML permite a las empresas intercambiar información en tiempo real, lo que aumenta la eficiencia de la comunicación y la toma de decisiones.
5. **Desarrollo de aplicaciones:** XML es una tecnología clave en el desarrollo de aplicaciones empresariales, ya que permite la representación y almacenamiento de información de una forma estructurada y accesible.

También se ha utilizado tanto para la definición de archivos de configuración, numerosos protocolos de comunicación y hasta como base para la especificación de otros sublenguajes, por ejemplo en **XHTML, RSS, RDF o Atom**.

Bloques y su adaptación responsive.

En mi parte del proyecto he decidido desarrollar una página lo más simple posible para que sea fácilmente adaptable al trabajo de mis compañeros.

Agregar Bootstrap al proyecto

Siguiendo la documentación en el sitio oficial de bootstrap,

<https://getbootstrap.esdocu.com/docs/5.1/getting-started/introduction/> copio y pego el `<link>` indicado en el `<head>`, justo antes que todas las demás hojas de estilo. En lugar de descargar todos los archivos y unirlos al proyecto, **he utilizado el enlace que apunta a un CDN remoto.**

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- metaetiqueta viewport sirve para configurar comportamientos responsive adecuados.-->

    <!-- Bootstrap CSS -->

    <!--El link de bootstrap sirve para cargar su CSS, copia y pega el <link> a la hoja de estilo en tu <head>, justo antes que todas las demás hojas de estilo.-->

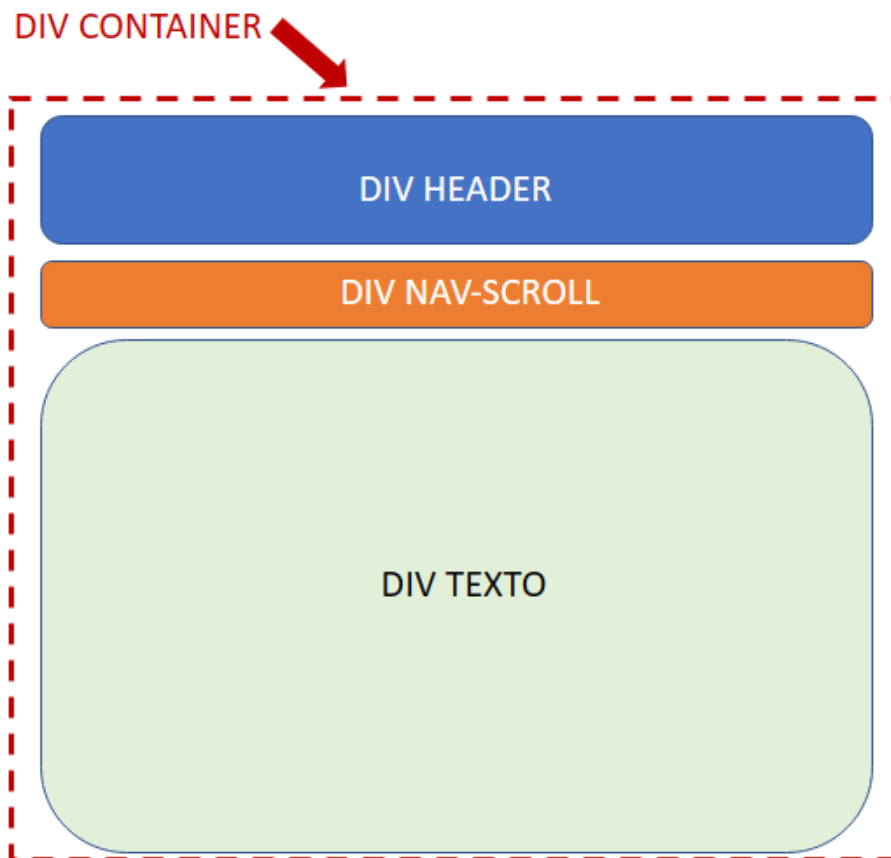
    <!--El css de bootstrap que he implementado hace que la estructura de la pagina se quede centrada-->

    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.4.1/dist/css/bootstrap.min.css" integrity="sha384-Vkoo8x4CGs03+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh" crossorigin="anonymous">
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.4.1/dist/js/bootstrap.min.js" integrity="sha384-wfSDF2E50Y2D1uUdj003uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCEx130g8ifwB6" crossorigin="anonymous"></script>

    <title>Articulo de XML</title>
    <link href="style.css" rel="stylesheet">
    <!-- con este link hago que este archivo de html este vinculado con el archivo style.css en donde hice que la pagina fuera responsive-->
  </head>
```

Diseño de los bloques

Mi diseño consta de tres bloques principales incluidos dentro de un contenedor global tal y como muestra la figura siguiente:



Container

Lo utilizo como el elemento más básico de cuadrícula de bootstrap, de ancho fijo que cambia el ancho máx en cada punto de interrupción para que sea sensible a un diseño responsive cuando añada los **@media**. Es una columna simple

```
.container
```

Para dar más claridad el div class va dentro de la etiqueta body y no en la misma

```
<div class="container"> <!-- class container o contenedor es el elemnto más basico de bootstrap son el sistema basico de cuadrícula
    en este caso utilizamos uno de ancho fijo que cambia el ancho max en cada punto de interrupción para que sea sensible a un diseño
    responsive cuando
    añada los @media  creo el div class dentro de la etiqueta body y no en la misma -->
```


Header

```
<header class="header">
  <h1 class="display-4 font-italic">Articulo de XML</h1>
</header>
```

La Clase CSS "header". Los estilos incluidos en esta clase son:

1. text-align: center: Alinea el texto contenido dentro del elemento con la clase "header" al centro.
2. background-color: rgb(222, 236, 236): Establece el color de fondo del elemento con la clase "header" en un tono suave de gris.
3. margin: 0%; Establece el margen en 0% para el elemento .
4. padding: 0%; Establece el relleno en 0% para el elemento

```
.header{
  text-align: center;
  background-color: ■ rgb(222, 236, 236);
  margin: 0%;
  padding:0%;
}
```

El resultado es:

Articulo de XML

Barra de navegación

Las barras de navegación y su contenido son fluidos por defecto. La clase "nav-scroller" aplica un estilo de diseño específico que hace que el contenido de la barra de navegación se desplace horizontalmente. Ocupando todo el ancho y justificando el espacio entre los items del menú de navegación.

```
<div class="nav-scroller py-1 mb-2">
  <!--Para la etiqueta nav le he puesto la clase d-flex para alinear los elementos en linea
  y tambien tiene el justify-content-between que hace que se ajusten los elementos al ancho definido y que el
  espacio este equilibrado entre ellos -->
  <nav class="nav d-flex justify-content-between">
    <!--Para el texto del nav he utilizado la clase p-2 para agregar un padding de 2
    que aumenta un poco el espacio interno y también le he puesto el text-muted
    para cambiar el color del texto a un tono más claro-->
    <a class="p-2 text-muted" href="index.html">HOME</a>
    <a class="p-2 text-muted" href="xml.html">XML</a>
    <a class="p-2 text-muted" href="json.html">JSON</a>
    <a class="p-2 text-muted" href="contactos.html">CONTACTO</a>
  </nav>
</div>
```

```
.nav-scroller {
  position: relative;
  z-index: 2;
  height: 2.75rem;
  overflow-y: hidden;
}
```

He utilizado una plantilla de ejemplo de bootstrap por que me ha parecido interesante el concepto de superposición de capas de z-index. Bootstrap incluye varios componentes que funcionan como una superposición y entre estos componentes están las barras de navegación pudiendo crear **así efectos visuales interesantes y sofisticados**. En este caso por

ejemplo muestra el elemento HOME sin subrayar y cuando pasas el ratón por encima superpone el subrayado como si fuese un Link (enlace) .

El resultado del uso de las capas:



Contenido

Para empezar lo que hice fue crear un div con la clase texto ,dentro de este div esta todo el contenido de investigación del xml aquí muestro la historia .

```
<div class="texto">
  <h2> XML Historia </h2>
  <p> La historia de XML comienza en 1996 cuando el World Wide Web Consortium (W3C) lanzó un proyecto para crear
    En 1998, el W3C publicó la primera versión de la especificación de XML 1.0, que se convirtió rápidamente en
    Hoy en día, XML es uno de los lenguajes de marcado más utilizados en la industria, y se utiliza en una ampli
  </p>
```

Resultado:

XML Historia

La historia de XML comienza en 1996 cuando el World Wide Web Consortium (W3C) lanzó un proyecto para crear un lenguaje de marcado extensible. La idea era tener un lenguaje que permitiera describir los datos en una forma estructurada y que fuera fácilmente interpretable por máquinas y humanos. En 1998, el W3C publicó la primera versión de la especificación de XML 1.0, que se convirtió rápidamente en un estándar de la industria para la representación y transferencia de datos en la Web. Desde entonces, se han lanzado varias versiones y extensiones de XML, incluyendo XML Schema, XSLT y XPath, que han ampliado la funcionalidad de XML y lo han convertido en una herramienta indispensable para la integración de aplicaciones y la gestión de datos en la Web. Hoy en día, XML es uno de los lenguajes de marcado más utilizados en la industria, y se utiliza en una amplia variedad de aplicaciones, incluyendo la gestión de contenidos, el intercambio de datos entre aplicaciones, la representación de datos en la Web y la integración de aplicaciones. Su popularidad se debe a su flexibilidad, escalabilidad y capacidad para describir los datos en una forma estructurada y fácilmente interpretable.

Luego use una lista para los casos de uso usando la etiqueta .

Resultado:

Normas De Uso y Sintaxis De XML

Las reglas que debemos seguir para crea la sintaxis de XML son:

- **Todos los elementos XML deben tener una etiqueta de cierre.**
 - Por ejemplo -- < nombre >John < /nombre >
- **Las etiquetas XML son sensibles a mayúsculas y minúsculas.**
 - < nombre > John < /Nombre > .. (Esto es incorrecto porque las etiquetas de apertura y cierre no coinciden en mayúsculas y minúsculas)
- **Todos los elementos XML deben estar debidamente anidados:**
 - < persona > < nombre>John < /nombre > < edad >30< /edad > < /persona > (El elemento raíz aquí es "persona")
- **Los valores de atributo siempre deben estar entre comillas:**
 - < persona edad = "30" > < nombre > John < /nombre >

```
<!--Lo que he hecho aqui es crear una lista anidada donde enumero la norma y anido un ejemplo para practicar las listas un poco más complejas -->
<ul>
  <li><u><b>Todos los elementos XML deben tener una etiqueta de cierre.</u></b>
    <ul>
      <li>Por ejemplo -- < nombre >John < /nombre > </li>
    </ul>
  </li>
  <li><u><b>Las etiquetas XML son sensibles a mayusculas y minusculas.</u></b>
    <ul>
      <li>< nombre > John < /Nombre > .. (Esto es incorrecto porque las etiquetas de apertura y cierre no coinciden en mayú
    </ul>
  </li>
  <li><u><b>Todos los elementos XML deben estar debidamente anidados:</u></b>
    <ul>
      <li>< persona > < nombre>John < /nombre > < edad >30< /edad > < /persona > (El elemento raíz aquí es "persona") </li>
    </ul>
  </li>
  <li><u><b>Los valores de atributo siempre deben estar entre comillas:</u></b>
    <ul>
      <li>< persona edad = "30" > < nombre > John < /nombre >
    </ul>
  </li>
</ul>
```

Después use una tabla para poder estructurar las imágenes y los elementos.

Resultado:

Todos los elementos XML deben tener una etiqueta de cierre

Forma incorrecta:

```
<body>Mira Spot corre.
```

```
<body>Mira Spot atrapa la bola.
```

Forma correcta:

```
<body>Mira Spot corre.</body>
```

```
<body>Mira Spot atrapa la bola.</body>
```

```

<h3>Todos los elementos XML deben tener una etiqueta de cierre</h3>

<!--Aquí organice las imagenes dentro de las celdas de una tabla dandoles un tamaño similar para que quedase visualmente más limpio-->

<table >
  <th> <mark>Forma incorrecta:</mark></th>

  <tr>
    <td></td>

  </tr>
  <th> <mark>Forma correcta:</mark></th>
  <tr>
    <td></td>

  </tr>
</table>

```

Por último pero no menos importante vendría la aplicación del xml en el ámbito empresarial utilice una lista ordenada con la etiqueta para enumerar de forma automática los elementos y le aplique css para que el texto se distribuya de manera equitativa y para que cada línea tenga una longitud similar, con excepción de la última línea, que puede ser más corta.

```

<h1> Aplicaciones del XML en el ámbito empresarial. </h1>
<p>XML es ampliamente utilizado en el ámbito empresarial para intercambiar información entre diferentes sistemas y plataformas de manera estructurada.</p>
<!--Utilizo esta vez una lista ordenada para que enumere de forma automática los elementos de la lista, los alinee de forma justificada que significa que dejan el mismo espacio en la parte izquierda y derecha-->
<ol>
  <li style="text-align:justify"><strong>Integración de sistemas:</strong> XML permite integrar diferentes sistemas y aplicaciones de manera estructurada.</li>
  <li style="text-align:justify"><strong> Documentación de procesos de negocio:</strong> XML se utiliza para documentar los procesos de negocio.</li>
  <li style="text-align:justify"><strong> Comercio electrónico:</strong> XML se utiliza en la transmisión de información relacionada con el comercio electrónico.</li>
  <li style="text-align:justify"><strong> Intercambio de información en tiempo real:</strong> XML permite a las empresas intercambiar información en tiempo real.</li>
  <li style="text-align:justify"><strong> Desarrollo de aplicaciones:</strong> XML es una tecnología clave en el desarrollo de aplicaciones web.</li>
</ol>

<p>
  También se ha utilizado tanto para la definición de archivos de configuración, numerosos protocolos de comunicación y hasta como base para la creación de lenguajes de marcado.</p>
</div>

```

Resultado :

Aplicaciones del XML en el ámbito empresarial.

XML es ampliamente utilizado en el ámbito empresarial para intercambiar información entre diferentes sistemas y plataformas de manera estructurada y organizada. Algunas de las aplicaciones más comunes de XML en el ámbito empresarial incluyen:

1. **Integración de sistemas:** XML permite integrar diferentes sistemas y aplicaciones de manera eficiente, al proporcionar un formato estandarizado para la transferencia de datos.
2. **Documentación de procesos de negocio:** XML se utiliza para documentar los procesos de negocio de una empresa y para modelar y representar los flujos de trabajo de una forma clara y fácil de entender.
3. **Comercio electrónico:** XML se utiliza en la transmisión de información relacionada con el comercio electrónico, como facturas, órdenes de compra y cualquier otra información relevante para los negocios.
4. **Intercambio de información en tiempo real:** XML permite a las empresas intercambiar información en tiempo real, lo que aumenta la eficiencia de la comunicación y la toma de decisiones.
5. **Desarrollo de aplicaciones:** XML es una tecnología clave en el desarrollo de aplicaciones empresariales, ya que permite la representación y almacenamiento de información de una forma estructurada y accesible.

También se ha utilizado tanto para la definición de archivos de configuración, numerosos protocolos de comunicación y hasta como base para la especificación de otros sublenguajes, por ejemplo en **XHTML, RSS, RDF o Atom**.

Implementación de grid Css

Utilice varias media query para que la página fuera responsive aplicando varios tamaños de ancho y metiendo todos los elementos.

```
/*@media permite consultar el tipo de pantalla que esta accediendo al contenido html y configurar dicho contenido para que se vea bien 'responsive'
Todo lo que va entre {} es lo que cambia. Por ejemplo si la pantalla es mayor o igual a 350px y menor o igual a 414px 'dispositivos móviles'
adapta todos los elemntos a las propiedades indicadas.Por ejemplo el tamaño de fuente se establece en 14px para todo el elemento body.*/
@media only screen and (max-width: 414px) and (min-width:350px) {
  body {
    font-size: 14px;
  }

  header {
    text-align: center;
    padding: 0%;
    margin: 0%;
  }

  .container {
    width: 100%;
    padding: 0%;
  }

  td img {
    width: 70%;
  }
  table {
    width: 70%;
  }
}
```

```

@media only screen and (max-width: 600px) and (min-width:540px) {
  body {
    font-size: 14px;
  }

  header {
    text-align: center;
    padding: 0%;
    margin: 0%;
  }

  .container {
    width: 100%;
    padding: 0%;
  }

  td img {
    width: 70%;
  }
  table {
    width: 70%;
  }
}

```

```

@media only screen and (max-width: 768px) and (min-width:720px) {
  /* CSS específicos para pantallas de tamaño máximo de 768px */
  body {
    font-size: 14px;
  }

  header {
    text-align: center;
    padding: 0%;
    margin: 0%;
  }

  .container {
    width: 100%;
    padding: 0%;
  }

  td img {
    width: 70%;
  }
  table {
    width: 70%;
  }
}

```

```

@media only screen and (max-width: 992px) and (min-width:960px) {
  body {
    font-size: 14px;
  }

  header {
    text-align: center;
    padding: 0%;
    margin: 0%;
  }

  .container {
    width: 100%;
    padding: 0%;
  }

  td img {
    width: 70%;
  }
  table {
    width: 70%;
  }
}

```

```

@media only screen and (max-width: 1200px) and (min-width:1140px) {
  body {
    font-size: 14px;
  }

  header {
    text-align: center;
    padding: 0%;
    margin: 0%;
  }

  .container {
    width: 100%;
    padding: 0 10px;
  }

  td img {
    width: 70%;
  }
  table {
    width: 70%;
  }
}

```

Historia sobre JSON

El nombre de JSON deriva de la expresión de la lengua inglesa JavaScript Notation. Se trata de un formato de texto ligero que permite intercambiar datos.

JSON se basa en un subconjunto de JavaScript, un lenguaje de programación imperativo, orientado a objetos e interpretado que fue creado por el estadounidense **Brendan Eich**.

En sus orígenes, **JSON** surgió como una alternativa a XML, un meta-lenguaje de etiquetas. Por la rapidez de lectura y su menor tamaño, **JSON** logró una rápida aceptación.

Para nosotros leer y escribir son acciones que resultan bastante sencillas. Los ordenadores no tienen dificultad para generarlo e interpretarlo. Por eso se usa con frecuencia para la transmisión de datos en aplicaciones web.

Reglas de uso y de sintaxis JSON

Todos los datos del archivo deben estar rodeados de llaves si quieres representar un objeto y entre corchetes cuadrados si es un arreglo.

- Llaves para representar objetos: los archivos JSON que representan objetos comienzan siempre con una llave de inicio y acaban con la llave de cierre.

```
{  
  "id": 44,  
  "game": "Colonos de Catan",  
  "author": "Klaus Teuber",  
  "ages": "10+"  
}
```

- Corchetes para representar los arrays: los archivos JSON que representan una lista de valores, es decir, un arreglo, comienzan por corchetes y terminan con corchetes.


```
[
  {
    "id": 56431,
    "name": "El Padrino",
    "year": 1972
  },
  {
    "id": 7553,
    "name": "El Padrino 2",
    "year": 1974
  },
  {
    "id": 19563,
    "name": "El Padrino III",
    "year": 1990
  }
]
```

- Valores representables con JSON: todos los valores que podemos encontrar representan lo siguiente:
 1. Un valor primitivo, que pueden ser de tipos diversos de JavaScript, numérico, cadena de caracteres, y booleanos (true / false).
 2. Arrays, es decir, una secuencia de valores separados por comas. En los arrays utilizamos los corchetes para delimitar el principio y fin de los valores que contienen.
 3. Objetos, es decir, otros elementos JSON con pares clave / valor.
 4. Además, también es válido el valor null.

Uso de comillas dobles

Se deben usar siempre comillas dobles a la hora de encerrar cadenas y nombres de los atributos del objeto.

Este punto es importante, porque la sintaxis de JavaScript para los literales de objetos permite colocar los nombres de las propiedades con y sin comillas, mientras que en JSON estamos obligados a colocar los nombres de las propiedades siempre con comillas dobles.

```
data.json / ...  
{  
  "id": 1,  
  "name": "Patatas nuevas",  
  "description": "Patatas cosechadas este año",  
  "available": true  
}
```

Aplicaciones en JSON en el ámbito empresarial

El formato JSON es ampliamente utilizado en aplicaciones web, especialmente, por su ligereza y velocidad. Existen diversas grandes empresas que utilizan este formato en sus páginas y sistemas, entre ellas: **Google, Yahoo, Facebook, Twitter** y muchas otras.

JSON es una notación que permite estructurar datos en formato texto para utilizarse en diferentes tipos de sistemas.

Se trata de un formato sencillo y liviano, que ofrece una serie de beneficios, como mayor velocidad en el tráfico en red y más agilidad en el procesamiento. Por eso, es ampliamente utilizado en aplicaciones web, aplicaciones móviles y como archivo de configuración.

WEB home:

home principal



```

1  <html lang="en">
2
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <link rel="stylesheet" href="diseño.css">
8      <title>HOME HTML</title>
9
10 </head>
11
12 <body>
13     <header>
14         <div class="container_titulo">
15             <h1 class="titulo">BLOG DE PROGRAMACION</h1>
16         </div>
17
18         <nav>
19             <a href="index.html">Home</a>
20             <a href="xml.html">XML</a>
21             <a href="json.html">Json</a>
22             <a href="contactos.html">Contacto</a>
23         </nav>
24     </header>
25
26     <aside>
27         <div class="grid">
28             <div class="container1">
29                 <a href="dondeestamos.html" class="item1">Ubicacion</a>
30             </div>
31             <div class="container2">
32                 <a href="https://www.json.org/json-en.html" class="item2">Ultimos Post</a>
33             </div>
34             <div class="container3">
35                 <a href="https://www.instagram.com/programacion.es/?hl=es" class="item3">Siguenos en redes sociales</a>
36             </div>
37             <div class="desplegable">
38                 <button class="boton">Categorias</button>
39                 <div class="menu">
40                     <a href="#">Desarrollo web</a>
41                     <a href="#">Desarrollo móvil</a>
42                     <a href="#">Buenas prácticas</a>
43                     <a href="#">Productividad y motivación</a>

```

```

44                 </div>
45             </div>
46         </div>
47     </aside>
48
49     <figure>
50         
51         <figcaption><p>Hecho por...</p>
52             <marquee behavior="scroll" direction="right" scrollamount="6">Ivan Encinas</marquee>
53             <marquee behavior="scroll" direction="right" scrollamount="7">David Garzon</marquee>
54             <marquee behavior="scroll" direction="right" scrollamount="8">David Del Pilar</marquee>
55             <marquee behavior="scroll" direction="right" scrollamount="9">Sergio Felipe</marquee></figcaption>
56     </figure>
57 </body>
58
59 </html>

```

Esta es la home de nuestra web, para comenzar el html y posteriormente explicaremos el código en css. Hemos realizado una barra de navegación mediante un nav incluyendo los elementos pedidos en el hito y con enlaces para ir a sus respectivas páginas. Más tarde una vez terminado con el header, realizamos una barra de navegación en el lado izquierdo de la web con enlaces a otras páginas y al final un menú desplegable. Para la home también hemos añadido una foto y al final hemos puesto quién ha realizado el trabajo utilizando un efecto en el que los nombres aparecen en movimiento mediante el marquee.

```
body{
  background-color: plum;
}

nav {
  display: flex;
  justify-content: space-between;
  align-items: center;
  height: 50px;
  background-color: rgb(204, 88, 181);
}

nav a {
  color: #ffffff;
  text-decoration: none;
  padding: 10px 20px;
}

nav :hover{
  color:rgb(97, 67, 133);
}

.container_titulo{
  background: linear-gradient(to right, rgb(97, 67, 133), rgb(81, 99, 149));
  height: 3px;
  padding-top: 100px;
}

.titulo {
  text-align: center;
  color: transparent;
  -moz-text-stroke-width: 2px;
  -webkit-text-stroke-width: 2px;
  -moz-text-stroke-color: #000000;
```

```

    -webkit-text-stroke-color: #ffffff;
    margin-top: -75px;
}

.titulo {
    text-shadow: 7px 7px black;
}
h1{
    margin: 0%;
}

.grid {
    display: inline-grid;
    row-gap: 20px;
    margin-top: 20px;
}

.item1{
    color: #ffffff;
    text-decoration:none;
}

.item2{
    color: #ffffff;
    text-decoration:none;
}

.item3{
    color: #ffffff;
    text-decoration:none;
}

.container1{
    padding:10px;
    background-color: rgb(204, 88, 181);
    border-radius: 50px;
}

.container2{
    padding:10px;
    background-color: rgb(204, 88, 181);
    border-radius: 50px;
}

.container3{
    padding:10px;
    background-color: rgb(204, 88, 181);
    border-radius: 50px;
}

```

```

}

.container1:hover{
  cursor: pointer;
  background-color: rgb(97, 67, 133);
  transform: scale(1.05);
  transition: all 0.6s ease-in-out;
}

.container2:hover{
  cursor: pointer;
  background-color: rgb(97, 67, 133);
  transform: scale(1.05);
  transition: all 0.6s ease-in-out;
}

.container3:hover{
  cursor: pointer;
  background-color: rgb(97, 67, 133);
  transform: scale(1.05);
  transition: all 0.6s ease-in-out;
}

.image {
  max-width: 100%;
  height: auto;
  display: block;
  margin: auto;
  margin-top: 10px;
  border-radius: 50px;
}

figcaption{
  color: #ffffff;
  font-family: fantasy;
  font-size: 20;
  text-align: center;
}

@media all and (max-width: 1000px){
  img{
    width:180px;
  }
}

@media all and (max-width: 400px){

```

```

img{
  width:100%;
}
}

.boton {
  background-color: rgb(204, 88, 181);
  color: white;
  border: none;
  cursor: pointer;
  font-size: 19px;
}

.desplegable {
  position: relative;
  display: inline-block;
  padding:10px;
  background-color: rgb(204, 88, 181);
  border-radius: 50px;
}

.desplegable:hover{
  cursor: pointer;
  transform: scale(1.05);
  transition: all 0.6s ease-in-out;
}

.menu {

  display: none;
  position: absolute;
  background-color: rgb(204, 88, 181);
  min-width: 160px;
  box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);
}

.menu a {

```



```

    color: #ffffff;
    padding: 12px 16px;
    text-decoration: none;
    display: block;
}

.menu a:hover {background-color: rgb(97, 67, 133)}

.desplegable:hover .menu {
    display: block;
}

p{
    font-size: 1.5rem;
}

.david{
    display: flex;
    justify-content: center;
    color: #ffffff;
    border-radius: 15px;
    background-color: rgb(97, 67, 133);
    margin-left: 30rem;
    margin-right: 30rem;
    margin-top: 10px;
}

.maps{
    border-radius: 54px;
}

.contenedor_maps{
    margin-top: 40px;
    display: flex;
    justify-content: center;
}

```

```

.firma{
  margin: 2rem;
}

@media only screen and (max-width: 900px) {
  .icono_compra {
    height: 1rem;
  }
}

@media (max-width: 767px) {
  nav {
    flex-direction: column;
    height: auto;
  }

  nav a {
    padding: 10px 0;
    text-align: center;
  }

  .titulo{
    font-size: 20px;
    margin-top: -25px;
  }

  .container_titulo {
    padding-top: 50px;
  }

  .david{

    margin-left: 15px;
    margin-right: 15px;
  }
}

@media (max-width: 437px) {
  nav {
    flex-direction: column;
    height: auto;
    width: 360px;
  }

  div{
    width: 360px;
  }
}

```

```

}
nav a {
    padding: 10px 0;
    text-align: center;
}
.container1{
    width: 200px;
}
.container2{
    width: 200px;
}
.container3{
    width: 200px;
}
.desplegable{
    width: 200px;
}

.titulo{
    font-size: 20px;
    margin-top: -25px;
}
.container_titulo {
    padding-top: 50px;
}
.maps{
    width: 200px;
    height: auto;
    display: flex;
    justify-content: center;
}
.david{
    margin-left: 15px;
    margin-right: 15px;
}
}

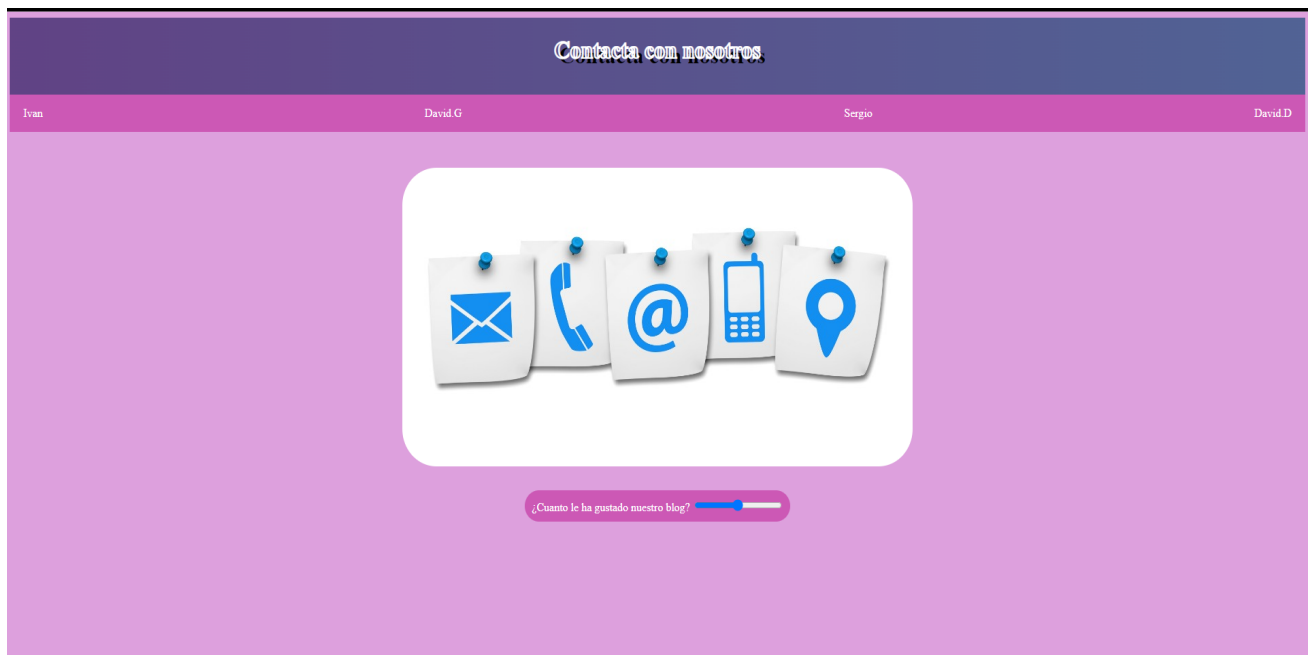
@media (max-width: 1226px) {
    .david{
        margin-left: 15px;
        margin-right: 15px;
    }
}

```

```
}  
@media (max-width: 390px){  
  .image_contact{  
    margin-left: 15px;  
    margin-right: 15px;  
  }  
}
```

En el css hemos utilizado la propiedad flex para colocar los elementos. En el título hemos aplicado elementos para dar sombra a las letras y así estilizarlo más. Para el fondo de pantalla hemos utilizado un color común para todas las páginas y enlaces. También hemos utilizado la función hover para que al pasar el ratón por encima de los elementos en los que hemos utilizado el hover cambie de color y se haga un poco más grande además en el último hemos añadido un desplegable con diferentes categorías. También para algunos elementos hemos añadido el border-radius. Hemos utilizado la función media query para hacer la web responsive en los diferentes tamaños de pantallas haciendo que los elementos cambien de tamaño mantengan margen a su lado y otras diversas funciones.

Web html de contacto



```
<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 rel="stylesheet" href="diseño.css">
  <title>contactos</title>
</head>
<body>
  <header>
    <div class="container_titulo">
      <h1 class="titulo">Contacta con nosotros</h1>
    </div>

    <nav>
      <a href="">Ivan</a>
      <a href="">David.G</a>
      <a href="">Sergio</a>
      <a href="">David.D</a>
    </nav>
  </header>
  <div class="image_contact" >
    
  </div>
  <div class="contenedor">
    <div class="contenedorizquierda"></div>
```

```

        <div class="formulario">
            ¿Cuanto le ha gustado nuestro blog?
            <input type="range" name="rango"
                min="0" max="1" step="0.1">
        </div>
        <div class="contenedorderecha"></div>
    </div>
</body>
</html>

```

Para realizar la web del contacto hemos utilizado la misma cabecera que utilizamos en la home, solo que cambiamos los elementos de la barra por nuestros nombres para simular nuestros contactos. Esta web es bastante básica ya que solo tiene una foto de decoración y un formulario para que el usuario pueda puntuar cuanto le ha gustado nuestra web.

CSS del contacto.

```

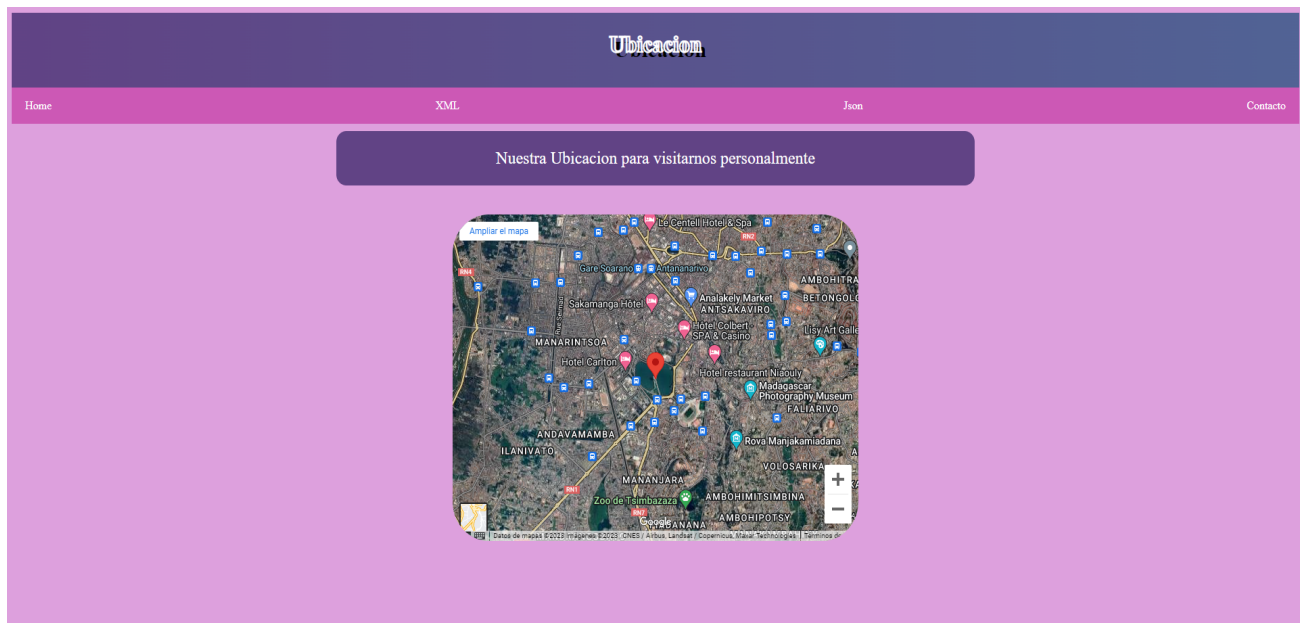
.image_contact{
    max-width: 100%;
    height: auto;
    display: flex;
    justify-content: center;
    margin-top: 10px;
    border-radius: 50px;
    margin-top: 3rem;
}
.border{
    border-radius: 50px;
}
.contenedor{
    display: flex;
    justify-content: center;
}
.formulario{
    color: #ffffff;
    margin-top: 2rem;
    background-color: rgb(204, 88, 181);
    margin-left: auto;
    margin-right: auto;
    border-radius: 25px;
    padding: 10px;
}

```

```
}  
  
@media all and (max-width: 1000px){  
  img{  
    width:180px;  
  }  
}  
  
@media all and (max-width: 400px){  
  img{  
    width:100%;  
  }  
}
```

En cuanto al css, hemos utilizado los mismos elementos y funciones que en la home para hacerlos responsive y colocar los elementos.

Web Ubicación



Codigo ubicacion

```
<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 rel="stylesheet" href="diseño.css">
  <title>info</title>
</head>

<body>
  <header>
    <div class="container_titulo">
      <h1 class="titulo">Ubicacion</h1>
    </div>

    <nav>
      <a href="index.html">Home</a>
      <a href="xml.html">XML</a>
      <a href="json.html">Json</a>
      <a href="contactos.html">Contacto</a>
    </nav>

  </header>
  <div class="david">
```



```

        <p>Nuestra Ubicacion para visitarnos personalmente</p>
    </div>
    <div class="contenedor_maps">
        <iframe class="maps"

src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d16018.1532878455
25!2d47.52581514440167!3d-18.913884096341352!2m3!1f0!2f0!3f0!3m2!1i1024!2i7
68!4f13.1!3m3!1m2!1s0x0%3A0x156cb0025057cacc!2zMTJjCsDU0JzU2LjEiUyA0N8KwMzEn
MTguMiJF!5e1!3m2!1ses!2ses!4v1675677653994!5m2!1ses!2ses"

        width="600" height="450" style="border:0;" allowfullscreen=""
loading="lazy"
        referrerpolicy="no-referrer-when-downgrade"></iframe>
    </div>
</body>

</html>

```

Para el html hemos vuelto a utilizar la barra de navegación anterior y lo único que metimos en esta página fue el maps para simular la ubicación de nuestra empresa para los clientes que utilicen nuestra web. Para introducir esto hemos utilizado un iframe, en el que hemos introducido el enlace de nuestra ubicación.

CSS de la ubicacion

```
208
209  ✓ .david{
210      display: flex;
211      justify-content: center;
212      color: ■ #ffffff;
213      border-radius: 15px;
214      background-color: ■ rgb(97, 67, 133);
215      margin-left: 30rem;
216      margin-right: 30rem;
217      margin-top: 10px;
218
219  }
220
221
222
223  ✓ .maps{
224      border-radius: 54px;
225  }
226
227  ✓ .contenedor_maps{
228      margin-top: 40px;
229      display: flex;
230      justify-content: center;
231  }
232
```

Básicamente en este css hemos utilizado la propiedad flex para colocar el div del titulo de encima de la imagen y también para colocar el mapa en el centro, además de aplicarle un border radius a este para que quede redondeado.