**FORMATO PARA EL DESARROLLO DE COMPONENTE FORMATIVO**

|  |  |
| --- | --- |
| PROGRAMA DE FORMACIÓN | ADSO |

|  |  |  |  |
| --- | --- | --- | --- |
| COMPETENCIA | 220501096. Desarrollar la solución de *software* de acuerdo con el diseño y metodologías de desarrollo. | RESULTADOS DE APRENDIZAJE | 220501096-03- Crear componentes *front-end* del *software* de acuerdo con el  diseño. |

|  |  |
| --- | --- |
| NÚMERO DEL COMPONENTE FORMATIVO | 017 |
| NOMBRE DEL COMPONENTE FORMATIVO | Desarrollo web con HTML, CSS y JavaScript |
| BREVE DESCRIPCIÓN | El diseño de un sitio web corresponde a un componente fundamental en el proceso de creación de aplicaciones web; por ende, este componente está orientado a la comprensión de los conceptos más importantes que se deben tener en cuenta, así como también la forma en que se deben codificar para construir de manera propia un sitio web o de apoyarse en sistemas gestores de contenidos denominados CMS. |
| PALABRAS CLAVE | HTML, CSS, JavaScript, web y CMS. |

|  |  |
| --- | --- |
| ÁREA OCUPACIONAL | 9 - PROCESAMIENTO, FABRICACIÓN Y ENSAMBLE |
| IDIOMA | Español |

1. **TABLA DE CONTENIDOS:**
2. HTML 5
   1. Etiquetas (*tags)*
   2. Textos
   3. Imágenes
   4. Hiperenlaces
   5. Tablas
   6. Formularios
3. Hojas de estilo
   1. Introducción CSS3
   2. Estructura
   3. Formulario CSS3
4. JavaScript
   1. Versiones
   2. Sintaxis
   3. Tipos de datos
   4. Estructuras de control
5. Gestores de contenido CMS
6. **INTRODUCCIÓN**

El diseño de interfaces web constituye una faceta crítica del desarrollo tecnológico moderno, actuando como el puente esencial que facilita la interacción entre los usuarios y las aplicaciones basadas en la web. A través de estas interfaces, las aplicaciones ofrecen sus servicios y funcionalidades, operando dentro de un entorno dinámico y accesible mediante navegadores web. Estas plataformas pueden ser accesibles a través de diversas redes, incluyendo LAN (Red de Área Local), WAN (Red de Área Amplia) o la omnipresente Internet, lo que les permite llegar a un público extenso y diverso.

En la era digital de hoy, la demanda por interfaces web intuitivas, eficientes y estéticamente agradables está en constante aumento. Las organizaciones y desarrolladores buscan innovar constantemente en este campo para mejorar la experiencia del usuario, reconociendo que una interacción positiva con la interfaz puede ser decisiva para el éxito de una aplicación web. Esta evolución no solo se centra en la estética visual sino también en la funcionalidad, con un enfoque particular en la adaptabilidad y la accesibilidad. Las interfaces modernas están diseñadas para ser fluidas y eficaces en una amplia gama de dispositivos, desde computadoras de escritorio y portátiles hasta tabletas y *smartphones*, garantizando así una experiencia de usuario coherente y satisfactoria independientemente del medio utilizado.

El diseño de interfaces web abarca múltiples dimensiones clave para el éxito en el entorno digital actual:

1. **DESARROLLO DE CONTENIDOS:** 
   * + 1. **HTML 5**

HTML5 es la piedra angular de la creación de contenido web moderno, permitiendo a los desarrolladores y diseñadores web construir experiencias ricas e interactivas en línea. A continuación, se detalla su definición y aplicación:

**Video 1.** HTML 5

Dejar espacio para video

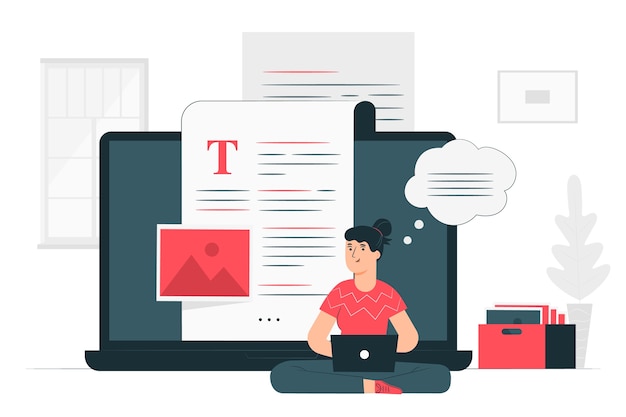


El conjunto de características y funcionalidades de cada etiqueta, que ha evolucionado con el tiempo, se conoce como HTML. Actualmente, está en la versión 5, la cual es regulada por el Consorcio World Wide Web (W3C).

* 1. **Etiquetas (*tags)***

Las etiquetas HTML representan el conjunto de funcionalidades que se pueden implementar en una página web, y poseen una sintaxis específica de escritura. Esto significa que existe una forma correcta en la que se deben escribir, cumpliendo con determinados atributos y manteniendo una interdependencia entre ellas. Por ello, algunas etiquetas dependen de otras para funcionar correctamente, a estas se les conoce como etiquetas padres. A continuación, se presenta un ejemplo de las etiquetas padres mínimas requeridas en un archivo .html:

|  |
| --- |
| Pestañas verticales  CF017\_1.1\_Etiquetas (tags) |

* 1. **Textos**

La inclusión de texto en una página web responde a la necesidad de presentar información a los visitantes. Todo texto que se añada debe estar contenido dentro de las etiquetas padres previamente mencionadas. En este contexto, se recomienda organizar el texto en párrafos, lo cual se logra mediante la utilización de la etiqueta **<p>,** seguido del texto deseado, y cerrando con la etiqueta de cierre **</p>**. De acuerdo con lo expuesto, se entiende que un conjunto de párrafos forma todo el documento, y los saltos de línea dentro de un texto se indican con la etiqueta **<br/>**. Además, otros elementos importantes para enriquecer el texto incluyen:

**Tabla 1.** Etiquetas HTML para textos

|  |  |
| --- | --- |
| **Etiqueta** | **Descripción** |
| <blockquote/> | Párrafo que en sí mismo corresponde a una cita de reseña. |
| <q/> | Cita contenida dentro de un texto. |
| <em/> | Enfatiza el texto de un párrafo. |
| <strong/> | Resaltado de texto. |

Un ejemplo de configuración de código HTML es:

|  |
| --- |
| <html>  <head>  </head>  <body>  Mi primer texto en una página web que no es un párrafo </br>  <p>Mi primer párrafo página web</p>  <blockquote>MI PRIMER PARRAFO CITA EN UNA PAGINAWEB<blockquote>  <p>Siempre se dice que este es el mejor sitio para ver videos online  <q cite="http://youtube.com/">www.youtube.com</q>  </p>  En ocasiones más vale <em>correr lento</em> pero seguro.  <p>Cuando el rio suena <strong>piedras lleva</strong>.</p>  </body>  </html> |

El resultado de la ejecución del código anterior es:

**Figura 1.**

Resultado del código

Primera línea de texto, con sangría: Mi primer texto en una página web que no es un párrafo.

Segunda línea de texto sin sangría: Mi primer párrafo página web.

Tercera línea de texto en mayúscula sostenida: MI PRIMER PARRAFO CITA EN UNA PAGINA WEB.

Cuarta línea, con la página web entrecomillada: Siempre se dice que este es el mejor sitio para ver videos online "www.youtube.com".

Quinta línea de texto con la palabras más, correr y lento en cursiva: En ocasiones más vale correr lento pero seguro.

Última línea, con las palabras piedras lleva en negrita: Cuando el río sueno piedras lleva.

* 1. **Imágenes**

Para anexar una imagen a un documento HTML se usa la etiqueta <img> acompañada del atributo src que corresponde a la ubicación en donde se encuentra la imagen, bien sea en formato \*.jpeg, \*.gif, \*.png, entre otros formatos.

Un ejemplo de código para imagen HTML es:

|  |
| --- |
| <html>    <head>    </head>    <body>     <img src="/misDocumentos/imagenes/2301-03.png" width="50% height:40%"> </img>    </body>  </html> |

* 1. **Hiperenlaces**

Esta funcionalidad permite la navegación entre diferentes páginas web, es decir, facilita el desplazamiento de una página a otra cuyos contenidos HTML están almacenados en archivos de distintos nombres y, posiblemente, en carpetas diferentes. Además, habilita la navegación hacia otras páginas web ya publicadas en internet.

Un ejemplo de código para hiperenlaces es:

|  |
| --- |
| <html>    <head>    </head>    <body>    <a href="http://www.google.com"> http://www.google.com </a>    </body>  </html> |

* 1. **Tablas**

Las tablas son elementos idóneos para estructurar contenido en una página web de manera matricial, es decir, organizado en filas y columnas. Para crear una tabla, se utiliza la etiqueta **<table>.** A continuación, se define la cantidad de filas mediante la etiqueta **<tr>** y, dentro de ella, se especifica el número de columnas deseado utilizando la etiqueta **<td>** para cada celda que se quiera presentar. Al final, cada combinación de fila y columna forma una celda de información.

El siguiente código ilustra la creación de una tabla con 2 filas y 3 columnas, donde el contenido de cada celda puede variar, incluyendo texto, hiperenlaces, imágenes, entre otros elementos.

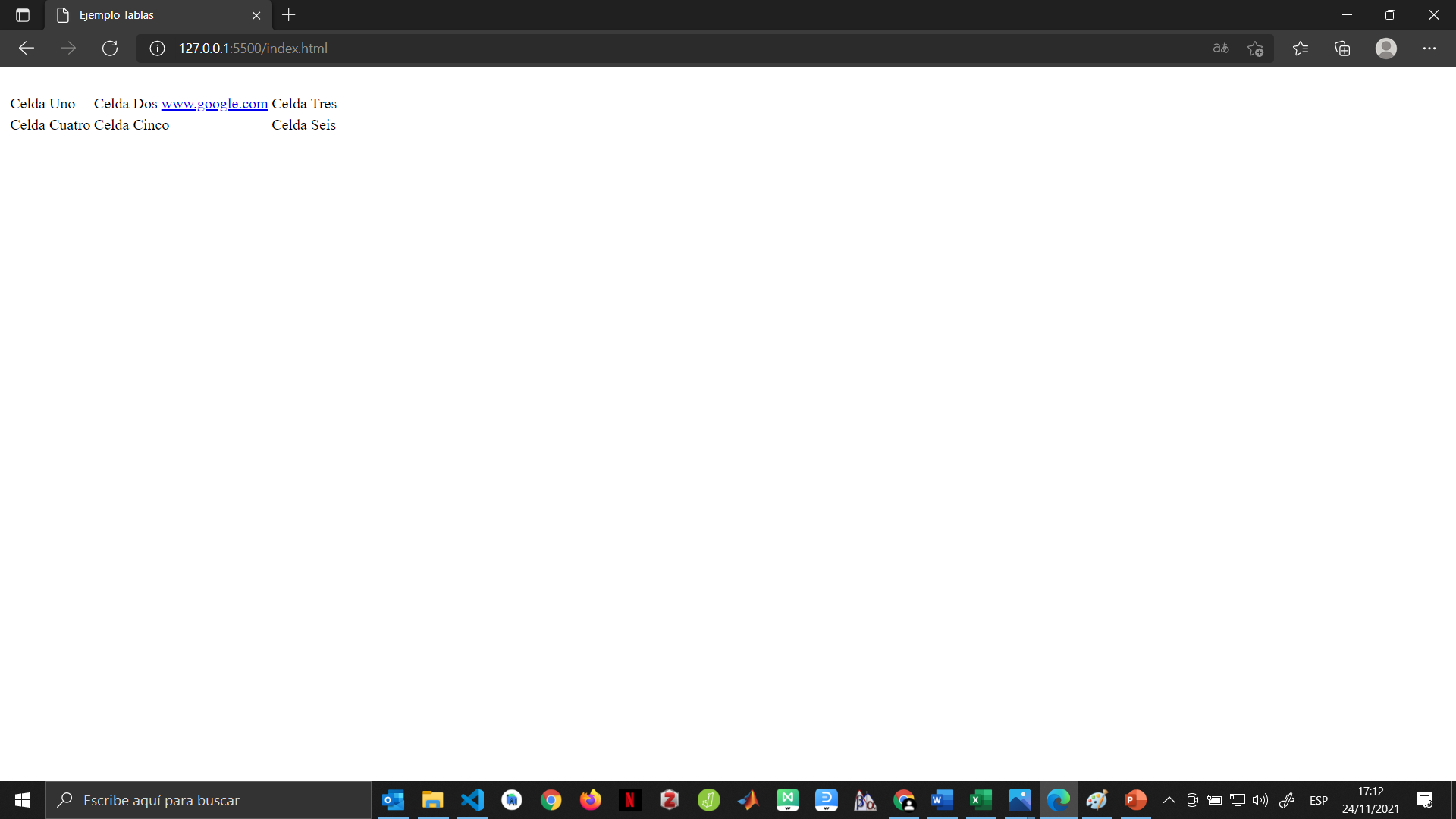
El código para una tabla de datos es:

|  |
| --- |
| <html>  <head>    </head>  <body>    <table>      <tr>        <td>Celda Uno </td>        <td>Celda Dos <a href="http://www.google.com">www.google.com</a></td>        <td>Celda Tres  </td>      </tr>      <tr>        <td>Celda Cuatro</td>        <td>Celda Cinco</td>        <td>Celda Seis</td>      </tr>      </table>  </body>  </body>  </html> |

El resultado ejecución del código de la tabla es:

**Figura 2.**

Tabla de datos



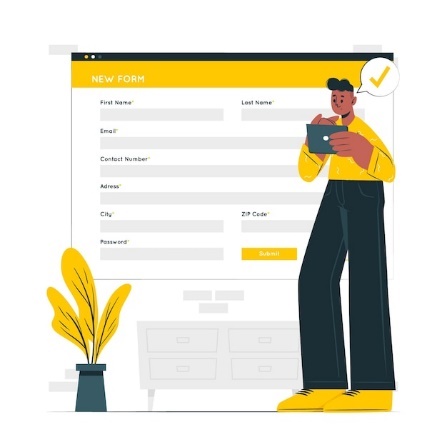
Un ejemplo de tabla usando Div:

|  |
| --- |
| <div class="tabla">     <div> Cabecera con el logo de la página “header” </div>     <div>      <div class="columna1"> Columna1 </div>      <div class="columna2"> Columna2 </div>      <div class="columna3"> Columna3 </div>     </div>     <div> Pie de página </div>  </div> |

* 1. **Formularios**

Los formularios son componentes HTML diseñados para facilitar la interacción y el intercambio de información con usuarios. Se caracterizan por agrupar elementos interactivos que permiten introducir datos de diversos tipos, como numéricos, alfanuméricos, caracteres, listas de datos, entre otros. La creación de formularios puede realizarse de manera tanto dinámica como estática.

|  |  |
| --- | --- |
| Ilustración del concepto de formularios | La etiqueta **<form>** se utiliza para definir un formulario y debe incluir, como mínimo, el atributo *action,* que especifica el destino al que se enviarán los datos. Además, es esencial definir el método de envío de los datos, que puede ser *post* o *get,* para determinar cómo se transmitirán estos datos al servidor. |

A continuación, se describen algunos tags que comúnmente conforman un formulario, estos se deben encerrar dentro de la etiqueta principal del formulario **<form>:**

* **<input>** Entrada de datos: estos pueden ser texto, número o alfanumérico; sus atributos más importantes son *type* para definir el tipo de entrada que puede ser *text,* radio, *submit, reset,* el atributo id permite identificar de manera única el objeto en el archivo HTML, el atributo *name* es para darle un nombre a la entrada o variable y *value* para recibir el dato del componente.
* **<label>** Permite la definición de un nombre para una entrada de datos.

Un ejemplo de código de un formulario es:

|  |
| --- |
| <html>             <head>             </head>  <body>         <form action=“http://misitio.com/prog/usuarionuevo” method=“post”>               <p>  <label for=“nombre”>Nombre:</label>  <input type=“text” id=“nombre”><br/>  <label for=“apellido”>Apellido:</label>  <input type=“text” id=“apellido”><br/>  <label for=“email”>email:</label>  <input type=“text” id=“email”><br/>  <input type=“radio” name=“sexo” value=“Varón”>Varón<br/>  <input type=“radio” name=“sexo” value=“Mujer”>Mujer<br/>  <input type=“submit” value=“Enviar”><input type=“reset”>           </p>                   </FORM>    </body>  </html> |

El resultado del código del formulario es:

**Figura 3.**

Resultado código del formulario

Formulario con los campos nombre, apellido y email con un campo en blanco para su diligenciamiento.

A su vez, de los campos varón y mujer para su selección, finalizando con los botones de enviar o restablecer. 

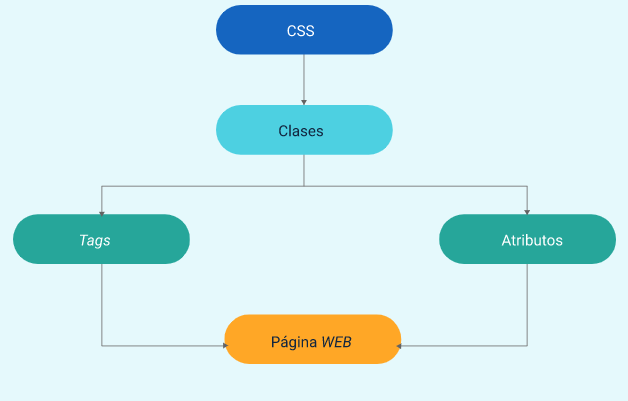
1. **Hojas de estilo**

Las hojas de estilo, conocidas como CSS (*Cascading Style Sheets*), son el mecanismo que confiere apariencia a una página web. Para ello, se crea un archivo con extensión .css, el cual contiene el código de estilización. Este código se compone de selectores (que pueden ser etiquetas HTML, clases, identificadores, entre otros) y un conjunto de propiedades y valores que definen el estilo de los elementos seleccionados.

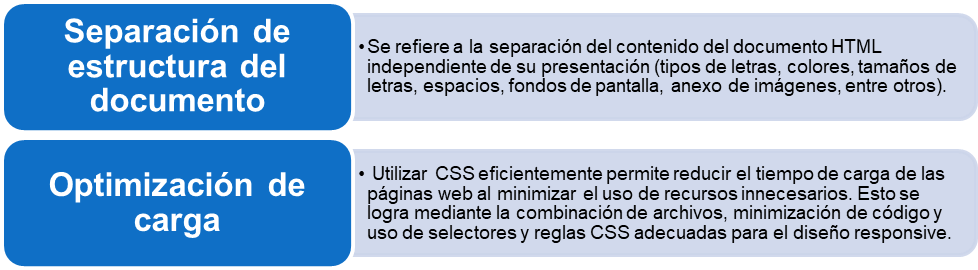
|  |  |
| --- | --- |
| Ilustración del concepto de codificación manual | La principal ventaja de utilizar CSS es la capacidad de **reutilizar el mismo código en múltiples páginas web**, lo que facilita la implementación de un diseño coherente a lo largo de todo el sitio. Además, el uso de hojas de estilo mejora significativamente el mantenimiento y control del diseño de un sitio web, permitiendo cambios globales en la presentación con modificaciones en un solo archivo. |

A continuación, se muestra un esquema general de cómo se puede implementar código CSS en una página web:

**Figura 4.** Esquema general CSS



Al momento de implementar las CSS es necesario tener en cuenta los siguientes puntos:



Un ejemplo de código de hoja interna es:

|  |
| --- |
| <head>     <style>      body{ background-color: #FFFFFF;      }      h1{       color: #000000;       text-align: center;      }     </style>  </head>  <body>  </body> |

Un ejemplo de código de hoja externa es:

|  |
| --- |
| <head>    <link rel="stylesheet" type="text/css" href="estilo.css">  </head>  <body>  </body> |

Un ejemplo de código de hoja en sitio web es:

|  |
| --- |
| <head>    <link rel="stylesheet" type="text/css" href="https://www.estilo.css">  </head>  <body>  </body> |

A continuación, algunos ejemplos:

|  |
| --- |
| **Acordeón**  **CF017\_2\_Hojas de estilo** |

* 1. **Introducción CSS3**

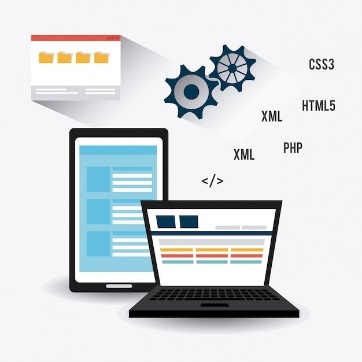
CSS3 representa una evolución significativa respecto a sus predecesores, CSS y CSS2, introduciendo una serie de mejoras que pueden categorizarse principalmente en dos aspectos:

Estas innovaciones hacen de CSS3 una herramienta poderosa para el diseño web moderno, permitiendo la creación de sitios web más atractivos, funcionales y accesibles.

Un ejemplo de estructura de uso de módulos:

|  |
| --- |
| .styles/  ..controlador.less  ..módulos/  ...configuraciones/  ….menu.less  ….comentarios.less  ….right/  …..right.less  ….left/  …..left.less  ….center/  …..center.less  ...ayudas/  ….ayudas.less  /\*\*\*\* Contenido de archivo controlador.less \*\*\*\*\*/  @import módulos/configuraciones/menu.less  @import módulos/ayudas/ayudas.less |

* 1. **Estructura**

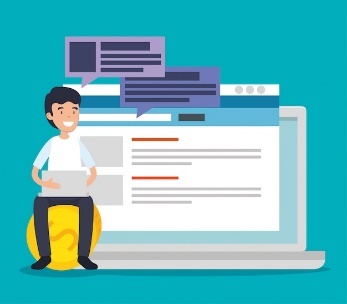
La estructura de CSS3 se basa en un conjunto de reglas almacenadas en un archivo. Cada regla consta de un selector, utilizado para identificar el elemento HTML al que se aplicarán los estilos, seguido de una o más declaraciones, que se separan entre sí por punto y coma (;). Una declaración incluye una propiedad y un valor asignado a esa propiedad. Además de definirse en un archivo CSS externo, estas declaraciones de estilo también pueden aplicarse directamente sobre un elemento HTML específico utilizando el atributo *style.* Para ello, se coloca la propiedad, seguida de dos puntos y su valor correspondiente, todo ello entre comillas dentro del atributo *style* en la etiqueta HTML deseada.

**Figura 5.** Ejemplo de estructura CSS3Código que incluye dos reglas:

SELECTOR { "PROPIEDAD : VALOR ; PROPIEDAD DOS : VALOR;"}

Un ejemplo con uso del atributo *style*

|  |
| --- |
| <html>  <head>  <title>  </title>  </head>  <body>  <h1 style=”color:red; size:30 px”>Texto1</h1>  <h2 style=”color:blue; size:40 px”>Texto2</h2>  <h3 style=”color:green; size:50 px”>Texto3</h3>  </body>  </html> |

* 1. **Formulario CSS3**

Los formularios en HTML se construyen siguiendo los principios básicos discutidos anteriormente. Sin embargo, con la introducción de HTML5, se han agregado nuevas funcionalidades que enriquecen los formularios web, incluyendo nuevos tipos de datos (como color*, tel, email, date, url, time*, *month, number, week, datetime-local, range, y search*), nuevos controles (output) y nuevos atributos (p*laceholder, autofocus, min, max, required, step, y pattern).* Según la recomendación del W3C, es importante verificar la compatibilidad de estas nuevas características con diferentes navegadores. Aunque inicialmente algunas de estas innovaciones se soportaban principalmente en navegadores específicos como Opera, la compatibilidad ha mejorado significativamente con el tiempo, y actualmente, muchos navegadores modernos soportan estas características.

Un ejemplo de formulario CSS3 es:

|  |
| --- |
| <html>  <head>  <meta charset="utf-8" /> <title>Nuevos componentes</title>  </head>  <body>  <form action="." oninput="range\_control\_value.value = range\_control.valueAsNumber">  <p>  Nombre: <input type="text" name="name\_control" autofocus required />  <br/>  Email: <input type="email" name="email\_control" required />  <br/>  URL: <input type="url" name="url\_control" placeholder="Escripe la URL de tu página web personal" />  <br/>  Fecha: <input type="date" name="date\_control" />  <br/>  Tiempo: <input type="time" name="time\_control" />  <br/>  Fecha y hora de nacimiento: <input type="datetime" name="datetime\_control" />  <br/>  Mes: <input type="month" name="month\_control" />  <br/>  Semana: <input type="week" name="week\_control" />  <br/>  Número (min -10, max 10): <input type="number" name="number\_control" min="-10" max="10" value="0" />  <br/>  Intervalo (min 0, max 10): <input type="range" name="range\_control" min="0" max="10" value="0" />  <output for="range\_control" name="range\_control\_value" >0</output>  <br />  Teléfono: <input type="tel" name="tel\_control" />  <br/>  Término de búsqueda: <input type="search" name="search\_control" />  <br />  Color Favorito: <input type="color" name="color\_control" />  <br/>  <input type="submit" value="Submit!" />  </p>  </form>  </body>  </html> |

El resultado del código anterior de formulario CSS3 es:

**Figura 6.**

Resultado código del formulario

Derivado del código anterior se muestra el formulario con todos sus campos configurados.

Donde se configuran los campos: nombre, email, url, fecha, tiempo, fecha de nacimiento, entre otros.

1. **JavaScript**

JavaScript es un lenguaje de programación que se integra en páginas web y es interpretado por los navegadores. La característica de ser "interpretado" se refiere a que el código se ejecuta de manera secuencial, línea por línea, por el intérprete del navegador, primordialmente en el lado cliente. Sin embargo, con el desarrollo de tecnologías como Node.js, JavaScript también se puede ejecutar en el lado del servidor, a diferencia de otros lenguajes de programación que requieren compilación.

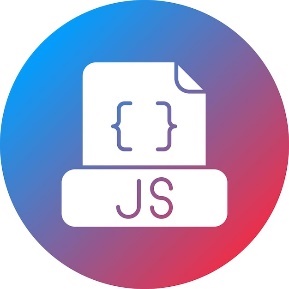
Es crucial distinguir entre JavaScript y Java, ya que, a pesar de la similitud en sus nombres, son lenguajes de programación completamente diferentes. Esto puede llevar a confusiones entre quienes no están familiarizados con ellos. Para integrar código JavaScript en una página web, existen varias maneras, entre ellas:



* Con código que se agrega dentro de la misma página.
* Con archivos independientes que se almacenan en rutas diferentes a la página.
* Con código referenciado en otras páginas web ya publicadas en la red internet.

Explorando la integración de JavaScript en la web, esta guía visual proporciona ejemplos claros y concisos de cómo insertar código JavaScript directamente en el HTML, así como cómo incluirlo desde archivos externos, y los resultados esperados de dicha implementación.

|  |
| --- |
| **Slides**  **CF017\_3\_Javascript** |

**3.1. Versiones**

El lenguaje JavaScript fue creado a mediados de la década de 1990 por Brendan Eich, con el objetivo de mejorar la experiencia del usuario. Durante esa época, la velocidad de internet era considerablemente lenta, y surgió la necesidad de un lenguaje de programación ejecutado del lado del cliente para agilizar las peticiones web.

Al año siguiente de su creación, se reconoció la importancia de estandarizar el lenguaje, dado que cada navegador lo interpretaba de manera diferente. Esto llevó a la propuesta y adopción de un estándar conocido como ECMA-262 Edición 1, también referido como ES1 (*ECMAScript 1*).

La evolución del lenguaje se muestra en la siguiente línea de tiempo:

**3.2. Sintaxis**

En JavaScript, se dispone de varios elementos fundamentales para la construcción de programas, tales como variables, constantes, operadores y expresiones. Las expresiones se forman mediante la combinación de variables, constantes y operadores. También se utilizan sentencias, que organizan el flujo del programa en un orden lógico, y funciones, que agrupan conjuntos de sentencias para resolver problemas específicos. Para manejar adecuadamente estos elementos, existen ciertas reglas de sintaxis y buenas prácticas que se deben seguir al escribir el código, las cuales son:

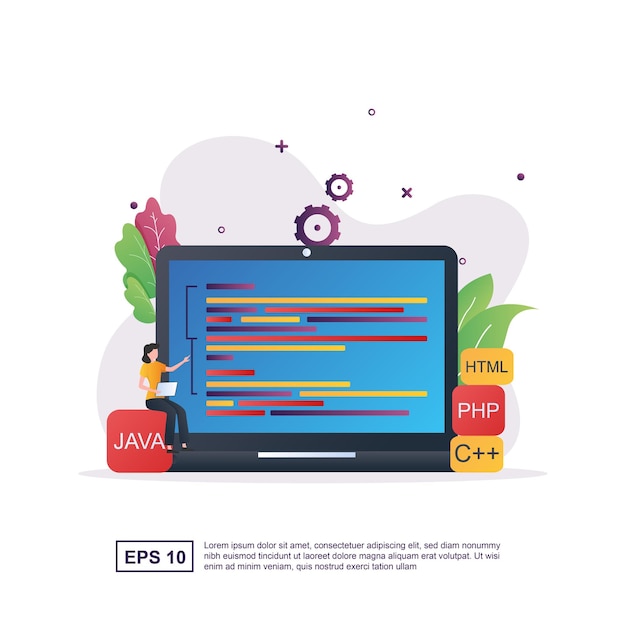
|  |
| --- |
| Acordeón  CF017\_3.2\_Sintaxis |

**3.3. Tipos de datos**

En JavaScript, una característica fundamental es su tipado dinámico, lo cual significa que el tipo de dato de una variable puede cambiar según el dato asignado por el usuario. Esto ocurre cuando se define una variable utilizando la palabra reservada var, let o const.

|  |
| --- |
| Pestañas  CF017\_3.3\_Tipos de datos |

**3.4. Estructuras de control**

En un programa JavaScript, las líneas de código se ejecutan de forma secuencial, es decir, en el orden en que aparecen: primero la primera línea, luego la segunda, la tercera, y así sucesivamente, nunca invirtiendo este orden, lo cual determina el flujo del programa. Durante este flujo, puede surgir la necesidad de tomar decisiones basadas en condiciones específicas, momento en el cual se emplean las estructuras condicionales, que pueden derivar en acciones diferentes dependiendo de si la condición evaluada resulta verdadera o falsa. Asimismo, en ciertos puntos del programa puede ser necesario iterar sobre estructuras de datos, como arreglos, para lo cual se utilizan bucles o ciclos.

**El condicional *if*** El más conocido de los condicionales, si se cumple la condición continúa ese camino. Se escribe entre paréntesis después de la palabra reservada *if*. Ejemplo con condicional *if:*

|  |
| --- |
| var edad=17;  //Condicional para saber si es mayor de edad  if(edad > 18 ){  alert(“Ya puede solicitar la cédula”);  } |

**Condicional *if/else*** Este condicional indica el camino alterno que puede seguir el programa en el caso que no se cumpla la condición de la palabra reservada *if*. Ejemplo con condicional *if/else:*

|  |
| --- |
| var edad=17;  var mensaje;  //Condicional para saber si es mayor de edad  if(edad > 18 ){  mensaje=“Ya puede solicitar la cédula”;  }else{ |

**Operador ternario** Es una forma de escribir el condicional *if/else* pero de una manera más corta; se identifica con el símbolo “?”. Ejemplo con operador ternario

|  |
| --- |
| var edad=17;  //Operador ternario: (condición ? Verdadero : Falso)  var mensaje = edad>18 ? ‘Cédula’: ’Tarjeta Identidad’; |

**Condicional *if* múltiple** Sirve para evaluar múltiples condiciones conservando la estructura del condicional *if* y para el caso alternativo se escribe la palabra reservada *if else* acompañada de la nueva condición a cumplir. Ejemplo con condicional *if* múltiple

|  |
| --- |
| var edad=17;  var mensaje;  //Condicional para saber si es mayor de edad  if(edad == 18 ){  mensaje=“Ya puede solicitar la cédula”;  } if else (edad > 18 ){  mensaje=“Tu cédula ya fué tramitada”;  } if else (edad < 18 ){  mensaje=“Aún eres menor de edad”;  }else{  mensaje=“Verifica el número de edad ingresado”;  }  alert(mensaje); |

**Condicional *switch*** Define casos específicos a realizar en el caso de que la variable expuesta como condición sea igual a los valores que se especifican mediante los *case*. Ejemplo con condicional *switch:*

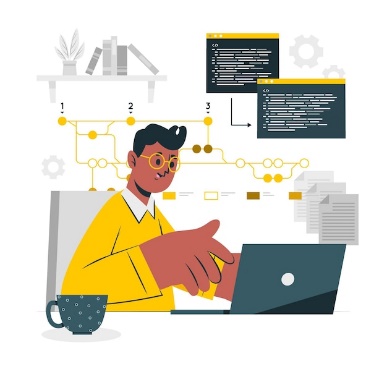
|  |
| --- |
| var nota = 7;  switch (nota) {  case 10:  calificacion = "Sobresaliente";  break;  case 9:  case 8:  calificacion = "Notable";  break;  case 7:  case 6:  calificacion = "Bien";  break;  case 5:  calificacion = "Suficiente";  break;  case 4:  case 3:  case 2:  case 1:  case 0:  calificacion = "Insuficiente";  break;  default:  // Cualquier otro caso  calificacion = "Nota errónea";  break;  }  alert(“Resultado”+calificacion); |

**Bucles e iteraciones**: los bucles son herramientas valiosas para automatizar tareas repetitivas, especialmente cuando se manejan grandes volúmenes de datos. Para implementar un bucle eficazmente, es crucial entender los siguientes conceptos:

|  |
| --- |
| Tarjetas  CF017\_3.4\_Estructuras de control |

A continuación, se presentan varios ejemplos de bucles:

**Bucle *while***

* Antes de entrar en el bucle *while*, se inicializa la variable i.
* Antes de realizar la primera iteración del bucle, se comprueba la condición.
* Si la condición es verdadera, se hace lo que está dentro del bucle.
* Se muestra por pantalla el valor de i y luego incrementamos el valor actual de i en 1.
* Se vuelve al inicio del bucle para hacer una nueva iteración. Se comprueba de nuevo la condición del bucle.
* Cuando la condición sea falsa, se sale del bucle y continúa el programa.

Ejemplo de bucle *while:*

|  |
| --- |
| i = 0; // Inicialización de la variable contador  // Condición: Mientras la variable contador sea menor de 5  while (i < 5) {  console.log("Valor de i:", i);  i = i + 1; // Incrementamos el valor de i  } |

Ejemplo bucle *for*

|  |
| --- |
| // for (inicialización; condición; incremento)  for (i = 0; i < 5; i++) {  console.log("Valor de i:", i);  } |

Ejemplo incremento múltiple

|  |
| --- |
| for (i = 0, j = 5; i < 5; i++, j--) {  console.log("Valor de i y j:", i, j);  } |

Ejemplo bucle en arreglos

|  |
| --- |
| var dato = [2, 6, 5, 1, 18, 44];  var msgForNormal = '';  var msgForIn = '';  //for normal  for (var i=0; i<dato.length; i++) {  msgForNormal = msgForNormal + dato[i] + ' - ';  } |

1. **Gestores de contenido CMS**

Los gestores de contenidos, conocidos como CMS (*Content Management System*), surgieron para simplificar la labor de los administradores de sitios web, comúnmente llamados *WebMasters.* Su desarrollo ha avanzado significativamente, posicionándolos como herramientas clave para publicar información y contenido propio en la web. Esto permite a los usuarios gestionar sus sitios sin la necesidad de externalizar el servicio o contratar a un profesional, evitando así costos innecesarios.

A continuación, se presenta una figura que ilustra la integración general de un CMS en un sitio web:

**Figura 7.**

Integración general del CMS



Un gestor de contenidos o CMS, es un *software* web diseñado para facilitar la creación, gestión y publicación de contenido en páginas web, sin requerir conocimientos avanzados en desarrollo de *software* o programación informática. Estas herramientas proporcionan una plataforma de administración virtual que permite a los usuarios crear, editar y publicar contenido web de manera eficiente.

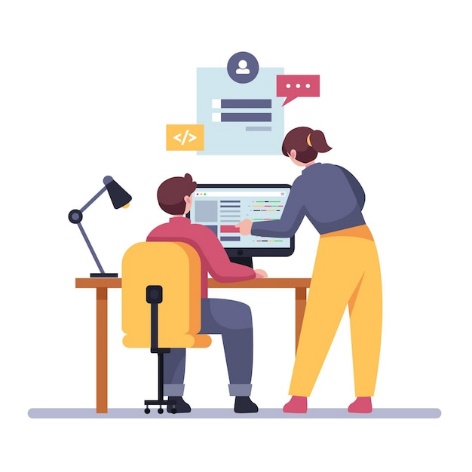
**Funciones**

A continuación, se describen las principales funciones que debe cumplir un CMS.

|  |
| --- |
| Acordeón  CF017\_4\_Gestores de contenido CMS (1) |

**Características**

A continuación, se presentan las características de los gestores de contenido CMS:

* El *frontend*, corresponde a la parte visible del CMS para los usuarios que acuden como invitados o usuarios registrados.
* Conformación modular para la configuración y personalización de los sitios web.
* Administración de usuarios, roles, perfiles y permisos de acceso.
* Componentes para la creación, edición, publicación de contenidos.
* Plantillas para distribuir y mantener el diseño del sitio web.
* Posibilidad de extensiones que complementan las funcionalidades en el gestor de contenidos.
* Características de manejo de internacionalización en varios idiomas.

**Clasificación CMS**

Se clasifican normalmente teniendo en cuenta los siguientes tres aspectos, de acuerdo al lenguaje de programación, a la distribución de la licencia y a las funcionalidades que presenta:

Teniendo en cuenta el lenguaje de programación:

|  |
| --- |
| Tarjetas  CF017\_4\_Gestores de contenido CMS (2) |

Teniendo como base la distribución de la licencia:

**De *software* libre (*Open Source)*:** significa que esta herramienta no presenta costos de licencia y el código fuente es accesible para ser modificado por los desarrolladores. El soporte técnico de este tipo de CMS es asumido por la entidad que lo implementa o por medio de comunidades online que comparten y mantienen su documentación.

**De código privado:** presentan costos de licencia y servicio, el código fuente es propietario por tanto sólo puede ser accedido y modificado por la compañía o personas que lo desarrollaron. En cuanto al soporte técnico es parte de la oportunidad de negocio y normalmente es de excelente calidad, así como una gran cantidad de información documental.

Teniendo en cuenta las funcionalidades que presenta:

|  |
| --- |
| Slides  CF017\_4\_Gestores de contenido CMS (3) |

Como se indicó previamente, recordamos que la clasificación de los CMS se presenta de la siguiente forma:

**Figura 8.** Esquema de clasificación de los CMS

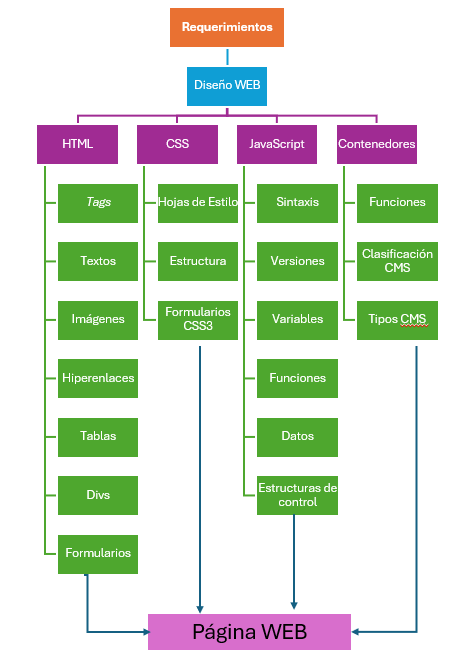


**Tipos CMS**

Los tipos de gestores de contenidos varían según los tipos de páginas web que se deseen desarrollar o crear. Para satisfacer estas necesidades, existen gestores de contenidos altamente especializados, equipados con herramientas y características diseñadas para cumplir con requerimientos específicos. A continuación, se presentan algunos de los gestores de contenidos disponibles para su uso en diferentes contextos:

1. **SÍNTESIS**

A continuación, se presenta una síntesis de la temática estudiada en el componente formativo.



1. **ACTIVIDADES DIDÁCTICAS (Se debe incorporar mínimo 1, máximo 2)**

|  |  |
| --- | --- |
| DESCRIPCIÓN DE ACTIVIDAD DIDÁCTICA | |
| Nombre de la Actividad | Estructuradores del HTML |
| Objetivo de la actividad | Reforzar el conocimiento de la estructura básica de un documento HTML, y el propósito de las etiquetas fundamentales que componen la estructura de cualquier página web. |
| Tipo de actividad sugerida | Arrastrar y soltar |
| Archivo de la actividad  (Anexo donde se describe la actividad propuesta) | *CF017\_Actividad didactica* |

1. **MATERIAL COMPLEMENTARIO:**

|  |  |  |  |
| --- | --- | --- | --- |
| Tema | Referencia APA del Material | Tipo de material  (Video, capítulo de libro, artículo, otro) | Enlace del Recurso o  Archivo del documento o material |
| HTML 5 | HTMLed.it. (2021). *Editor HTML gratuito en línea, limpiador y convertidor.* | Página web | <https://htmled.it/or/> |
| JavaScript | Cubic Factory. (2021). *Ejecutar JavaScript.* | Página web | <https://www.cubicfactory.com/jseditor/> |

1. **GLOSARIO:**

*s*

|  |  |
| --- | --- |
| TÉRMINO | SIGNIFICADO |
| **CMS:** | Sistema manejador de contenidos. |
| **CSS3:** | Hojas de estilo en cascada, versión 3. |
| **ES6:** | Abreviación de ECMAScript, versión 6. |
| ***Frontend*:** | Capa de presentación de una aplicación web. |
| ***Get*:** | Método de envío de datos por URL web. |
| **JavaScript:** | Lenguaje de programación. |
| **LAN:** | Red de área local. |
| **Landing Page:** | Página web de aterrizaje, diseñada específicamente para una campaña de marketing o publicidad. |
| **LESS:** | Hoja de estilo dinámica, extensión que amplía las capacidades de CSS. |
| **MAN:** | Red de área metropolitana. |
| **NODE.js:** | Entorno de ejecución para JavaScript del lado del servidor. |
| ***Plugin:*** | Aplicación que extiende las funcionalidades de un programa principal. |
| **Post:** | Método de envío de datos por formulario web. |
| **SEO:** | Optimización para motores de búsqueda, estrategias para mejorar la visibilidad de un sitio web. |
| **WAN:** | Red de área extensa o mundial. |

1. **REFERENCIAS BIBLIOGRÁFICAS:**

Acibeiro. M. (2021). *Qué es un gestor de contenidos y cuál es el mejor para crear una web.* <https://www.lucushost.com/blog/gestor-de-contenidos/>

Beati, H. (2016). *HTML5 y CSS3.*

Cuervo, P. V. (2019). *10 características de un CMS*. <https://www.arquitectoit.com/cms/10-caracteristicas-cms/>

Gutierrez, R. (2018). *Understanding the role of digital commons in the web; The making of HTML5.*

Hverbeke, M. (2018). *Eloquent JavaScript: A Modern Introduction to Programming.*

Lenguaje JS. (2021). *¿Qué son los bucles? Bucles, iteraciones y repetición de código.* <https://lenguajejs.com/javascript/introduccion/bucles/>

Mooc, Aprendizaje Online.(2015). *Clasificación de los sistemas de gestión de contenidos*. <https://mooccontenidosweb.wordpress.com/2015/05/01/clasificacion-de-los-sistemas-de-gestion-de-contenidos/>

Mora, S. L. (2021). *HTML5 y CSS3: <6> HTML5 – Formularios</6>.* <http://desarrolloweb.dlsi.ua.es/cursos/2011/html5-css3-es/html5-formularios>

1. **CONTROL DEL DOCUMENTO**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Nombre | Cargo | Dependencia | Fecha |
| Autor (es) | Carlos Hernán Muñoz Carvajal | Experto Temático | Regional Cauca - Centro de Teleinformática y Producción Industrial | Abril 2024 |
| Paola Alexandra Moya | Evaluadora instruccional | Regional Antioquia - Centro de Servicios de Salud | Abril 2024 |
|  | Olga Constanza Bermúdez Jaimes | Responsable Línea de Producción Antioquia | Regional Antioquia - Centro de Servicios de Salud | Abril 2024 |

1. **CONTROL DE CAMBIOS**

**(Diligenciar únicamente si realiza ajustes a la Unidad Temática)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Nombre | Cargo | Dependencia | Fecha | Razón del Cambio |
| Autor (es) |  |  |  |  |  |