**FORMATO PARA EL DESARROLLO DE COMPONENTE FORMATIVO**

| **PROGRAMA DE FORMACIÓN** | Técnico en programación de *software* |
| --- | --- |

| COMPETENCIA | **220501096**  Desarrollar la solución de *software* de acuerdo con el diseño y metodologías de desarrollo. | RESULTADOS DE APRENDIZAJE | **220501096\_03**    Codificar el *software* utilizando el lenguaje de programación y la plataforma seleccionada. |
| --- | --- | --- | --- |

| **NÚMERO DEL COMPONENTE FORMATIVO** | CF12 |
| --- | --- |
| **NOMBRE DEL COMPONENTE FORMATIVO** | Diseño de interfaces web |
| **BREVE DESCRIPCIÓN** | El diseño de un sitio web corresponde a un componente fundamental en la creación de aplicaciones web, por ello este componente se orienta a la comprensión de los conceptos más importantes y 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** | CMS, diseño, estilos, HTML, JavaScript, página web. |

| **ÁREA OCUPACIONAL** | 9 - Procesamiento, fabricación y ensamble |
| --- | --- |
| IDIOMA | Español |

1. **TABLA DE CONTENIDOS:**

**Introducción**

**1. HTML 5**

1.1. *Tags*

1.2. Textos

1.3. Imágenes

1.4 Hiperenlaces

1.5 Tablas

1.6 Formularios

**2. Hojas de estilo**

2.1 Conceptos

2.2 Introducción CSS3

2.3 Estructura

2.4 Formularios CSS3

**3. JavaScript**

3.1 Introducción

3.2 Versiones

3.3 Sintaxis

3.4 Tipos de datos

3.5 Estructuras de control

**4. Gestores de contenido**

4.1 Definición

4.2 Funciones

4.3 Características

4.4 Clasificación CMS

4.5 Tipos CMS

1. **INTRODUCCIÓN:**

Le damos la bienvenida al componente formativo denominado “**Diseño de interfaces web**”, el cual hace parte del programa de formación Técnico en programación de *software*. A continuación en el siguiente video, se conocerá un poco más acerca del contenido:



1. **DESARROLLO DE CONTENIDO:**
2. **HTML 5**

Internet, en su contexto general, está programado por códigos internos, lo cuales cuando se accede a alguna página web, siempre le llegarán esos códigos al navegador y los traduce de modo, que visualmente  el usuario o cibernauta vea lo que el diseñador de la web ha creado para brindar información.

Al conjunto de característica de cada etiqueta y sus diferentes funcionalidades que han evolucionado en el tiempo se le llama HTML*,* que a fecha de hoy, ya va en la versión 5 y es regulado por el consorcio internacional [W3C](https://es.wikipedia.org/wiki/W3C). El Consorcio WWW**,** World Wide Web Consortium (W3C), es un consorcio internacional que genera recomendaciones y estándares que aseguran el crecimiento de la *World Wide Web* a largo plazo.

En el siguiente video, se podrá conocer un poco más de HTML 5:



**1.1. Etiquetas (*Tags*)**

Representan el conjunto de funcionalidades que se puede implementar en una página web y cuentan con una sintaxis de escritura, es decir, la forma correcta como se debe escribir y que debe cumplir con unos atributos y una interdependencia entre ellas, es decir, existen algunas etiquetas que, para su funcionamiento, dependen de otras estas etiquetas se conocen como etiquetas padre, a continuación un ejemplo:

Las etiquetas padre mínimas con las que debe cumplir un archivo .html son las siguientes:

<HTML> Indica el inicio del documento y que el documento está escrito en el lenguaje html*.*

<HEAD>Es el encabezado de la página y esta información va orientada al browser.

</HEAD>indica el fin de la etiqueta head ambas etiquetas deben quedar dentro de la etiqueta padre html*.*

<BODY>Representa el cuerpo del contenido de la información que va a visualizar el usuario.

</BODY>Indica el fin del cuerpo

</HTML> Indica el final del documento html



También se pueden encontrar algunas etiquetas que permiten mostrar contenidos más específicos o permiten crear comportamientos en el documento, a continuación se presentan etiquetas que se utilizan con este fin al momento de construir el *head* o cabecera:



El cuerpo no es más que la estructura que se crea dentro del *body*, normalmente se le llama plantilla, pues permite enseñar de forma organizada por secciones la información que le llega al usuario.

| Estas son algunas estructuras del *body* que normalmente se utilizan y se identifican al momento de ingresar a un sitio web o a una aplicación web, estas estructuras pueden variar según el diseñador web pero, normalmente, se siguen algunos estándares:   * Cabecera. * Barra de navegación. * Información principal. * Barra lateral. * Institucional. | Gráfico, Gráfico de rectángulos  Descripción generada automáticamente con confianza media |
| --- | --- |

A continuación, se mostrará un ejemplo escrito en código HTML de esta estructura:



En el siguiente recurso se podrán conocer los elementos que hacen parte de la estructura dentro del *body*:



* 1. **Textos**

La inclusión de texto en una página web nace de la necesidad de presentar información a los visitantes. Todo el texto que se incluya se debe presentar dentro de los tags padres antes descritos, dentro de ese orden de ideas, la recomendación es ordenar estos textos a manera de párrafos que vienen dado por la utilización de la etiqueta <p> y a continuación el texto que se quiere incluir y se cierra con la respectiva etiqueta de cierre</p>. En función de lo planteado, se puede considerar que el conjunto de párrafos comprende todo el documento y los respectivos saltos de línea dentro de un texto viene dado por la etiqueta <br/>.

Otros elementos importantes para enriquecer un texto son los siguientes:

<blockquote/> Párrafo que en sí mismo corresponde a una cita de reseña.

<q/> Cita contenido dentro de un texto.

<em/> Enfatizando el texto de un párrafo.

<strong/> Resaltado de texto.

Ejemplo:

<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 PÁRRAFO 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 río suena < strong > piedras lleva</strong>.</p>

</body>

</html>

El resultado de la ejecución del código anterior se puede visualizar a continuación:





**1.3. 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.

Otros atributos importantes para la etiqueta <img> son width y height que sirven para definir el ancho y el alto de la imagen dentro de la página web.



**1.4 Hiperenlaces**

Es la funcionalidad que permite navegar entre diferentes páginas web, es decir, poder desplazarse por donde el contenido HTML se ha almacenado en archivos con nombres diferentes, incluso almacenados en diversas carpetas donde también se puede navegar hacia otras páginas web ya publicadas en la red internet.

Para lograr un hiperenlace se debe utilizar la etiqueta <a> acompañada del atributo href=”” que indica el destino hacia la página a la que se quiere llegar y dentro de la etiqueta el texto que describe el sitio destino.

Ejemplo:

<html>

<head>

</head>

<body>

      <a href="http://www.google.com">www.google.com</a>

</body>

</html>

El siguiente recurso permite conocer paso a paso, el funcionamiento de los hiperenlaces:



**1.5 Tablas**

Las tablas son los componentes apropiados para organizar el contenido dentro de una página web en forma de matriz; es decir, con filas y columnas.

Para definir una tabla se debe usar la etiqueta <table> acto seguido se debe definir la cantidad de filas dentro de la tabla con la etiqueta <tr> y a su vez se debe definir dentro de la etiqueta <td> el número de columnas que se quieren manejar para presentar la información, al final cada ítem fila-columna se convierte en una celda de información.

Ejemplo: el siguiente código define una tabla de datos con dos filas y tres columnas, el contenido de cada celda puede variar bien sea en su texto, hiperenlaces, imágenes u otros elementos.

<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>

</html>

El resultado de la ejecución del código anterior se puede visualizar a continuación:





* **Divs**

Si bien es cierto en el numeral anterior (1.5 Tablas) se explicó el funcionamiento de las tablas, en el diseño de hoy se recomienda usar la etiqueta **div** que corresponde a un contenedor y permite realizar lo mismo de las tablas, pero con mayores opciones en funcionalidad y dinamismo.

Ejemplo de tabla usando div:

<div class="tabla">  
 <div>Cabecera con el logo de la página "header" </div>  
 <div>  
 <div class="columna1">Columna 1 </div>  
 <div class="columna2">Columna 2 </div>  
 <div class="columna3">Columna 3  </div>  
 </div>  
 <div>Pie de página </div>  
</div>

**1.6 Formularios**

Los formularios son componentes HTML que permiten la interacción para intercambio de información con usuarios externos estos se caracterizan por agrupar a su vez a un conjunto de elementos interactivos en donde se pueden ingresar datos de tipo numérico, alfanumérico, carácter, listas de datos entre otros este ingreso y su construcción se pueden elaborar tanto de manera dinámica como estática.

El tag que representa un formulario se denomina <form> y debe ser acompañado de los siguientes atributos action que permite llegar al sitio destino que va a recibir los datos y método que permite definir el tipo de método que puede recibir los datos bien sea post o get.

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

<input> Entrada de datos estas 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.

Ejemplo de un formulario:

<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 de la ejecución del código anterior se puede visualizar a continuación:





**2. Hojas de estilo**

Las hojas de estilo son el mecanismo mediante el cual se otorga a una página web su apariencia, para esto se crea un archivo nuevo con extensión .css que hace referencia a un tipo de archivo denominado CSS (*Style Sheet Cascade*) hojas de estilo en cascada.

Estas hojas de estilo se escriben en un código que está compuesto por *tags* con sus atributos y los valores que adquieren estos atributos. La característica principal es que este código puede ser reutilizado sobre las páginas web en donde suelen repetirse las implementaciones de las mismas etiquetas; las hojas contribuyen a la facilidad de mantenimiento y control de varias páginas incluso de todo un sitio web.

**2.1 Conceptos**

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

1. Separación de estructura del documento, se refiere a la separación del contenido del documento HTML independiente de su presentación (tipos de letras, colores, tamaños de letras, espacios, fondos de pantalla, anexo de imágenes entre otros).
2. Hojas de estilo según su manejo pueden ser internas, externas o desde sitio web. Cuando es interna el código del estilo va incluido dentro de la página al ser externa se define en un archivo diferente a la a página con extensión CSS y si es desde otro sitio web se hace referencia por medio de su URL.



**2.2 Introducción CSS3**

CSS3 es la evolución de CSS y CSS2, esta nueva versión presenta como novedad mejoras que a grandes rasgos pueden dividirse en dos características; la primera es la aplicación del concepto de módulos que permite aproximar la definición del estilo de un sitio web a partir de la programación es decir donde se puede hacer para ello la analogía a los patrones de diseño que implementan algún controlador por lo que los módulos permiten definir contextos de aplicación de estilos e identificarlos en archivos diferentes (con extensión .less) que finalmente se integran en un archivo controlador, la segunda es la posibilidad de enriquecer los elementos que conforman las páginas directamente desde el estilo con animaciones, transiciones, *layout* con columnas, gradientes para modificar colores, transformaciones para rotar elementos así como también el ingreso de uso de fuente de texto externas por medio de librerías.

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

**2.3 Estructura**

La estructura de CSS3 corresponde al conjunto de reglas recopiladas en un archivo; una regla a su vez corresponde a la utilización de algún tipo de selector como los que ya vimos en puntos anteriores y una o varias declaraciones separadas por punto y una (;) la declaración se conforma de una propiedad y un valor; dichas declaraciones se pueden escribir de manera directa en el tag de HTML deseado utilizando el atributo style y entre comillas la propiedad los dos puntos y su valor.

Ejemplo de estructura:







{ : ; : }



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>

**2.4 Formularios CSS3**

Estos formularios se construyen de manera normal básicamente como lo vimos en los puntos anteriores solo que en esta nueva versión se caracteriza por que se agregan nuevos tipos de datos (*color, tel, email, date, URL, time, month, number, week, datetime, range, search*), nuevos controles (*output*) y nuevos atributos (*placeholder, autofocus, min, required, max, step, pattern*) para estos casos por recomendación de la W3c es necesario verificar su funcionamiento en los diferentes navegadores hasta el momento solo funcionan bien solo con el navegador Opera.

Ejemplo de formulario:

<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 de la ejecución del código anterior se puede visualizar a continuación:





**3. JavaScript**

Es un lenguaje de programación que se inserta en una página web que es interpretado por los navegadores el concepto de interpretado hace referencia a la ejecución secuencial línea por línea que hace el intérprete del navegador ante todo del lado cliente, aunque hoy en día también se logra del lado servidor con el apoyo Node.js el cual no es compilado como otros lenguaje de programación.

Este lenguaje está orientado a brindar posibilidades de construcción de código HTML de manera dinámica es decir el programa que se crea construye el conjunto de *tags* HTML de manera automática multiplicando de manera sustancial el enriquecimiento del diseño de las interfaces web.

**3.1 Introducción**

Es importante tener en cuenta que JavaScript y Java con lenguajes diferentes dado que por llevar el mismo nombre se puede presentar para confusiones; para insertar código JavaScript a un página web se puede hacer de las siguientes formas; con código que se agrega dentro de la misma página, con archivos independientes que se almacenan en rutas diferentes a la página y con código referenciado en otras páginas web ya publicadas en la red internet.

Ejemplo de código interno aquí se incluye el código dentro del tag <script></script> después del tag <title>:

<html>

<head>

<title>

Página web con código JavaScript embebido

</title>

<script type=”text/javascript”> insertar tag con atributo type indicando el uso del lenguaje java JavaScript

alert(“Mi primer mensaje usando JavaScript”);

</script>

</head>

<body>

<p>Contenido de la pagina web</p>

</body>

</html>

Ejemplo de inclusión de código desde archivos con extensión .js en rutas diferentes a la de la página:

<html>

<head>

<title>

Pagina web con código JavaScript embebido desde otros archivos fuera de la página actual.

</title>

<script src="static/js/miCodeUno.js"></script>

<script src="static/js/miCodeDos.js"></script>

<script src="static/js/miCodeTres.js"></script>

</head>

<body>

<p>Contenido de la pagina web</p>

</body>

</html>

Ejemplo de inclusión de código desde archivos con extensión .js en url diferentes a la de la página:

<html>

<head>

<title>

Pagina web con código JavaScript embebido desde otros archivos fuera de la página actual.

</title>

<script src="https://www.sitiouno.com/developmentUno.js" crossorigin></script>

</head>

<body>

<p>Contenido de la pagina web</p>

</body>

</html>

El resultado de la ejecución del código anterior se puede visualizar a continuación:



Ejemplo de inclusión de código desde archivos con extensión .js en url diferentes a la de la página:

<html>

<head>

                     <title>

Página web con código JavaScript embebido desde otros archivos fuera de la página actual.

      </title>

<script src="https://www.sitiouno.com/developmentUno.js" crossorigin></script>

</head>

<body>

<p>Contenido de la página web</p>

</body>

</html>

**3.2 Versiones**

El lenguaje Java Script aparece aproximadamente a mediados de los años noventa elaborado por Brendan Eich como una forma de mejorar la experiencia de usuario, puesto que en aquella época la velocidad de internet era de demasiado lenta y se requería de un lenguaje del lado cliente al momento de realizar una petición web al año siguiente 1996 se propuso determinar un estándar puesto que cada navegador lo interpreta a su manera y por eso surgió ECMA-262 Ed.1 ES1; la evolución del lenguaje muestra en la siguiente tabla.

**Tabla 1**

*Versiones*

| **Año** | **Versión** |
| --- | --- |
| Mayo 1995 | Mocha |
| Septiembre 1995 | Live Script |
| Diciembre 1995 | Java Script |
| 1996-1997 | ECMA-262 Ed.1 ES1 |
| 1998 | ES2 |
| 1999 | ES3 Moderno js baseline |
| 2005 | Ajax Revolución |
| 2008 | ES4 RIP |
| 2009 | ES5 |
| 2012 | ES6 Under way |
| 2015 | ES6 |
| 2016 | ES7 |
| 2017 | ES8 |
| 2018 | ES9 |
| 2019 | ES10 |
| 2020 | ES11 |

**3.3 Sintaxis**

En JavaScript existe un conjunto de elementos que permiten conformar un programa, variables, constantes, operadores, expresiones que corresponden a una forma de combinación de variables, constantes y operadores, sentencias encargadas de guardar un orden lógico y funciones como el conjunto de sentencias que permiten resolver un problema; para todos estos elementos existe una serie de reglas que se deben cumplir al momento de escribir el código así:

Las variables se definen como espacios de memoria reservada para almacenar algún dato y que se identifican con un nombre para definirlas se debe utilizar la palabra var seguida del nombre el símbolo igual un valor finalizado con punto y coma.



Las constantes como su nombre lo indica sirven para definir valores únicos que se pueden utilizar en diferentes partes del programa se definen con la palabra cons seguida del nombre de variable el símbolo igual y termina con punto y coma.



Los operadores se definen en tres grupos operados de asignación, operadores de relación y los operadores lógicos. Los operadores de asignación son igual =, suma +=, resta -=, multiplicación \*=, división /= y módulo % los operadores de relación son mayor >, mayor igual >=, menor <, menor igual <= , igual que == y diferente de != finalmente los operadores lógicos para evaluar verdadero o falso son AND && , OR || y NOT !.

Las sentencias se conforman como la combinación de contantes, variables y operadores



Las funciones permiten identificar con un nombre una acción dentro de un programa, es la combinación de sentencias, variables, constantes y expresiones se escribe utilizando la palabra function seguida de una lista de parámetros de entrada separados por comas encerrados entre paréntesis () en donde el conjunto de sentencias que operan esas entradas se encierran entre llaves {}.



**3.4 Tipos Datos**

Con relación a los tipos de datos JavaScript presenta un característica fundamental y es que cuenta con la capacidad de adaptarse al dato ingresado por el usuario es decir que el tipo de dato es dinámico cuando fué definido con la palabra de reserva var.

* **Variables numéricas:** se utilizan para definir variables enteras (integer) , reales (float), hexadecimal(0x) y octal(0).



* **Variable de caracteres:** permite almacenar cadenas de texto dentro de las comillas para bajar un renglón en el texto se debe utilizar \n para agregar un tabulador \t para comillas simple \’ para comillas dobles \” y para barra inclinada \\.
* **Variables tipo lógico:** permiten definir variables booleanas falso o verdadero.



* **Variables tipo arreglo:** denominados también como matrices o vectores y se usan para definir variables que se relacionan en un mismo conjunto se definen entre corchetes [] separados por coma y valores entre comillas.
* **Variables tipo objeto:** son variables que a diferencias de los arreglos se define dentro de llaves con el nombre de propiedad y su valor.



**3.5 Estructuras de control**

Cuando se hace un programa JavaScript las líneas de código se ejecutan de manera secuencial es decir primera, segunda, tercera, cuarta, etc., pero nunca cuarta y luego tercera siempre se enumeran de arriba hacia abajo correspondiendo esto con el flujo del programa; durante este flujo en algún punto de las secuencias es necesario tomar una decisión y es ahí donde se debe utilizar el concepto de condicionales con respuesta positiva o negativa. En otro punto del programa se necesitará realizar a recorridos a estructuras de datos por ejemplo un arreglo y para ello se utilizará el concepto de bucle.

**Condicional if**, el más conocido de los condicionales si se cumple la condición continúa ese camino la condición se escribe entre paréntesis después de la palabra reservada if.



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



**Operador ternario**, es una forma de escribir el condicional if/else, pero de una manera más corta se identifica con el símbolo?.

**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 acompaña de la nueva condición a cumplir.



**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.



**Bucles e Iteraciones**, útiles para programar tareas que se convierten en repetitivas en donde se involucran gran cantidad de volumen de datos permitiendo su automatización para realizar un bucle, es necesario tener en cuenta los siguiente conceptos: la condición tal como se vio en los if, esta se evalúa repetidamente, la iteración es la cantidad de repeticiones que ejecutó la condición, el contador es la variable que guarda en memoria el número en el cual va la repetición, el incremento es el número que se adiciona un vez se termine un ciclo y bucle infinito cuando no se realiza el incremento al contador esto es necesario evitarlo.

* **Bucle While** 
* Antes de entrar en el bucle while, se inicializa la variable i a.
* 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.
* **Bucle for**

Incremento múltiple



* **Bucle en Arreglos**



o **Selectores**

| Los selectores funcionan de forma muy similar a los del CSS y permite seleccionar y encontrar elementos dentro del HTML. |  |
| --- | --- |
|  |  |
| El selector se puede utilizar buscando por la clase o por el tipo. |  |

* **Eventos**

La interacción del usuario con una aplicación, página o sitio web se establece como evento, por lo tanto un evento es una acción que el usuario quiere hacer en determinada parte de la aplicación, por ejemplo darle clic en un botón o cambiar un valor de una elemento de texto entre otros.

Estas acciones se capturan mediante manejadores de eventos ya establecidos y se programan mediante funciones en JavaScript.



**4. Gestores de Contenido CMS**

Los gestores de contenidos o CMS nacieron para facilitar el trabajo a los administradores de sitios web normalmente denominados *webmasters*, y han logrado avanzar tanto que se han convertido en opciones importantes en el momento de exponer información y contenido propio en la web sin tener que externalizar el servicio o contratar a un profesional evitando así gastos innecesarios.

**4.1 Definición**

Un gestor de contenidos o CMS es un sistema manejador de contenidos que corresponde a un *software* orientado a la web que permite crear, administrar y publicar una página web en la red Internet sin la necesidad de contar con conocimientos en desarrollo de *software* y mucho menos en programación de computadores. Estas aplicaciones cumplen un papel de administración virtual en línea que permiten crear, editar y publicar contenido web.

**4.2 Funciones**

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

* + - * **1. Independencia entre el código y el contenido**

Permiten independizar la parte que corresponde a la gestión de la estructura y funcionalidad de la web, de la parte de administración de contenidos. De esta manera, se puede mantener trabajando sobre el sistema CMS perfiles totalmente separados. Por un lado, el perfil desarrollador puede configurar las capacidades del CMS o extenderlas, mientras que el perfil constructor de contenidos estará enfocado en la publicación de los contenidos.

* + - * **2. Plantillas en el diseño web**

Permiten modificar la capa de visualización y el diseño de la estructura del sitio web de una forma sencilla mediante el uso de plantillas que corresponde a colores, tipos de letras, imágenes, posicionamiento etc. que son comunes en las diferentes páginas. De esta manera se puede aplicar cada plantilla con el diseño que se desee sobre el contenido común para que este pueda verse de múltiples formas.

Se pueden encontrar por un lado plantillas sencillas y de carácter gratuito y por otro parte plantillas de orden profesional con un gran diseño y así mismo con gran coste asociado.

* + - * **3. Extensión de capacidades**

Los CMS o gestores de contenido ofrecen una serie de funcionalidades básicas que en ocasiones no pueden llegar a cubrir todas las necesidades de un sitio web es así como se dispone de la posibilidad de crear sistemas de extensión (*plugins*) para añadir nuevas funcionalidades de una forma sencilla sobre la estructura del sitio web.

* + - * **4. Enfocados al SEO**

Permiten implementar en los sitios web el posicionamiento SEO (*Search Engine Optimization*) el cual consiste en ejecutar las labores de posicionamiento, visualización en los principales motores de búsqueda de los navegadores por tanto el contenido que es la parte más importante debe quedar siempre bien posicionado cuando se realizan las búsquedas.

* + - * **5. Gestión de enlaces**

Cada ingreso al sitio web denominado *landing home* tiene asignado un enlace o *permalink*. Los CMS o gestores de contenidos permiten gestionar de una forma muy ágil, rápida y sencilla el enlace asociado a cada contenido. Este elemento está correlacionado con el anterior en donde se busca llevar una buena concordancia entre el título, contenido y enlace de la web como técnica primordial en las técnicas SEO. Así se puede asociar de forma personalizada un enlace con cada contenido.

* + - * **6. Flujos de trabajo**

Hace referencia a la posibilidad de definir responsabilidades dentro del grupo humano que contribuye en la administración del gestor de contenidos donde, normalmente, se encuentran dos perfiles, el perfil que genera el contenido y lo publica al CMS o gestor de contenidos y al perfil que revisa y aprueba el contenido y que finalmente lo hace público.

Los CMS ofrecen flujos de trabajo de aprobación un poco complejos en los que estos dos perfiles puedan realizar el procedimiento de publicación de contenidos.

* + - * **7. Comentarios**

Mucho del contenido que se sirve es susceptible de ser comentado. De esta manera, casi todos los sistemas CMS o gestores de contenidos llevan incluido un sistema de gestión de comentarios dónde se podrá validar el autor del comentario, así como hacer una administración de los mismos.

* + - * **8. Control de versiones**

El contenido cambia con el tiempo, incluso pueden existir varios usuarios que estén modificando el contenido, realizando modificaciones o anotaciones. En este punto es importante el control de versiones, es decir, que se guarde como versión, ya sea una versión importante o una simple revisión, el contenido modificado. Por ello, los CMS ofrecen el poder ir versionando el contenido, recuperar versiones anteriores a la que estamos trabajando y publicar una versión en concreto del contenido.

* + - * **9. Múltiples idiomas**

Muchas de las webs que se construyen publican contenido en múltiples idiomas. La estructura de la página es la misma, pero el contenido está publicado en varios idiomas. Así, los CMS permiten realizar la gestión de contribuir el contenido en varios idiomas y la capacidad de mostrarlo atendiendo a las preferencias del visitante.

* + - * **10. Múltiples sitios**

Las webs que desarrollamos ya no suelen tener un único sitio (o site) si no que suelen tener diferentes secciones de entrada o *landing pages*. Una de las características de los CMS es gestionar varios sitios desde una plataforma centralizada, evitando realizar operaciones de mantenimiento en diferentes sitios, con el consecuente coste asociado.

* + - * **11. Seguridad**

La seguridad es uno de los puntos importantes de los CMS, de tal manera que se enfocan a que se pueda perfilar el tipo de contenido dependiendo del usuario que lo consume. Y ofrecen una seguridad general enfocada a dar protección a todo el sitio web.

El resumen de las funcionalidades de un CMS se presenta a continuación:



**4.3 Características**

* El *frontend*, corresponde a la parte visible del CMS para los usuarios que acceden 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.

**4.4 Clasificación CMS**

Se clasifican normalmente teniendo en cuenta los siguientes tres aspectos: de acuerdo con el lenguaje de programación, la distribución de la licencias y las funcionalidades que presenta.

**1. Teniendo en cuenta el lenguaje de programación**

* Lenguaje Java: jAPS, Liferay, DSpace, Fedora, Nuxeo EP, Magnolia, Hippo CMS, Calenco, Polopoly, IBM Lotus web Content Management, Day Communiqué WCM, Jarimba, Vignette.
* Lenguaje PHP: Drupal, CMS Made Simple, Joomla!, Mambo, PHP-Nuke, TikiWiki, TYPO3, WordPress, Xoops, Zikula, Jadu, ExpressionEngine, Accrisoft Freedom, CMS 10, Dim Works CMS, Content-SORT, Prodigia Easy Site Manager, PipePS, SiteAd CMS.
* ASP.NET de Microsoft: DotNetNuke Community Edition, Umbraco, mojoPortal, Kentico CMS, SharePoint Server, Telligent Community, Ektron CMS400.NET, Quantum Art QP7, webControl CMS.

**2. Con base en 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. En esta categoría, algunos de los gestores más reconocidos son WordPress, Drupal, Joomla, Plone, TYPO3, OpenCMS, PHPNuke o Moodle.
* **De código privado:** presentan costos de licencia y servicio, el código fuente es propietario por tanto solo 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 calidad información documental. Los gestores más destacados son CMS10, Eximius2 CMS, Contendo CMS, Jarimba, CMS HYDRAportal, OnBase, IWEB, Oracle Portal, PipePS, Paloo, Smartone CMS, Vbulletin, XCM – Xeridia Content Manager, Zweb Publisher CMS.

**3. De acuerdo con su funcionalidad**

* **Plataformas en general:** se destacan los siguientes nombres (Drupal, Gekko, E107, Joomla, Mambo, PHP-Nuke, TYPO3, TYPOLight, XOOPS, Zweb Publisher CMS, ADSM Portal 2.0, 360 web Manager *software*, GTLive).
* **Sitios de educación:** entre los cuales se encuentran (ATutor, Claroline, Dokeos, eCollege, FrogTeacher, Moodle, Sakai Project, Scholar360, Synergeia, Teleto).
* **Blogs:** WordPress, bBlog, DotClear, Lifetype, Plone, Nucleus CMS, Blogger, Textpatter.
* **Galerías:** Gallery, Pixelpost, Expression Engine.
* **Wikis:**sitios donde grupos de personas publican contenido de manera rápida (MediaWiki, TikiWiki, Twiki).
* **Comercio electrónico:** para ventas *online* (osCommerce, Magento, Zen Cart, Drupal e-Commerce, CubeCart, Open cart, VirtueMart).
* **Groupware:** para coordinar trabajo colaborativo en procesos de negocio (webcollab, eGroupWare, Groupware).

A continuación, se presenta la clasificación de los CMS:



**4.5 Tipos CMS**

Los tipos dependen a su vez de los tipos de páginas web en que se desee involucrar o que se desee crear para ello cuentan con una serie de gestores de contenidos muy especializados equipados con herramientas y características acordes a cada necesidad.

Se pueden encontrar los siguientes gestores de contenidos para utilizarlos en:

1. Blogs.
2. Páginas corporativas.
3. Tiendas online o e-commerce.
4. Sitios de e-learning.
5. Foros.
6. Wikis.
7. **Síntesis**



1. **ACTIVIDADES DIDÁCTICAS**

| DESCRIPCIÓN DE ACTIVIDAD DIDÁCTICA | |
| --- | --- |
| Nombre de la Actividad | Conceptos de diseño de interfaces web. |
| Objetivo de la actividad | Afianzar algunos de los conceptos más importantes del diseño web. |
| Tipo de actividad sugerida | Selección múltiple. |
| Archivo de la actividad  (Anexo donde se describe la actividad propuesta) | Anexos/ActividadDidactica1 |

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** |
| --- | --- | --- | --- |
| **1. HTML 5** | HTMLed.it. (2021). *Editor HTML gratuito en línea, limpiador y convertidor.* https://htmled.it/or/ | Página web | https://htmled.it/or/ |
| **3. JavaScript** | Cubic Factory. (2021). *Ejecutar JavaScript.* https://www.cubicfactory.com/jseditor/ | Página web | https://www.cubicfactory.com/jseditor/ |

1. **GLOSARIO:**

| **TÉRMINO** | **SIGNIFICADO** |
| --- | --- |
| **CMS** | sistema manejador de contenidos. |
| **CSS3** | hojas de estilo en cascada versión 3. |
| **ES6** | abreviación del estándar ECMA SCRIPT Versión 6. |
| **Frontend** | capa de presentación de una aplicación web. |
| **GET** | método de envío de datos por URL web. |
| **HTML** | lenguaje de marcado de hipertexto. |
| **JavaScript** | lenguaje de programación. |
| ***Landing Page*** | tipo de página web denominada de aterrizaje. |
| **LAN** | red de área local. |
| **LESS** | extensión de tipos de archivos CSS3. |
| **NODE.js** | servidor de aplicaciones. |
| **MAN** | red de área metropolitana |
| **Post** | método de envío de datos por formulario web. |
| **Plugin** | aplicación que extiende las funcionalidades de un componente. |
| **SEO** | posicionamiento de páginas web en los motores de búsqueda. |
| **WAN** | red de área mundial. |

1. **REFERENCIAS BIBLIOGRÁFICAS**

Acibeiro. M. (2021). Qué es *un gestor de contenidos y para qué se utiliza*. <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.*

Java desde 0. (2020). *Introducción a JavaScript*. <https://javadesde0.com/introduccion-a-javascript/>

Java Script en español. (2021). *Bucles e iteraciones*. <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*. <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. | Septiembre de 2021 |
| Deivis Eduard  Ramírez Martínez | Diseñador  Instruccional | Regional Distrito Capital - Centro para la Industria de la Comunicación Gráfica | Noviembre  2021 |
| Andrés Felipe Velandia Espitia | Revisor Metodológico y Pedagógico | Regional Distrito Capital - Centro de Diseño y Metrología | Noviembre  2021 |
| Sandra Patricia Hoyos Sepúlveda | Revisión y corrección de estilo | Centro para la Industria de la Comunicación Gráfica - Distrito capital | Noviembre 2021 |
| David Eduardo Lozada Ceron | Experto Temático | Regional-Cauca Centro de teleinformática y producción industrial. | Mayo de 2022 |
| Paula Andrea Taborda Ortiz | Diseñadora Instruccional | Regional Norte de Santander - Centro de la Industria, la Empresa y Los Servicios CIES | Mayo de 2022 |
| Carolina Coca Salazar | Asesora metodológica | Regional distrito capital - Centro de Diseño y metrología. | Mayo de 2022 |
| Rafael Neftalí  Lizcano reyes | Responsable Equipo de Desarrollo Curricular | Regional Santander - Centro Industrial de Diseño y la Manufactura | Mayo de 2022 |
|  | José Gabriel Ortiz Abella | Corrector de estilo | Regional Distrito Capital - Centro de Diseño y metrología. | Octubre del 2022. |

1. **CONTROL DE CAMBIOS**

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