Programación Web

Miguel Hernández Bejarano

Luis Eduardo Baquero Rey

Universidad ECCI
https://www.ecci.edu.co/publicaciones/

ISBN 978-958-8817-40-8
http://dx.doi.org/10.18180/LIBROECCI.ISBN.978-958-8817-40-8
Editorial Universidad ECCI

Edición 1
Correctores de estilo: Cristhian García y Ginna Morera.

2020

Dedicatoria

Miguel Hernández: A dios, a mi esposa (Flor Ángela) e hijos (Jefferson, Julieth y Oscar).

Luis Baquero: A mis hijos (Luis Alejandro, Rosa Valentina, Sebastián, Cristal e Isabela).

Índice general

Índice de figuras

Tablas

Prólogo

La aparición de las tecnologías de la información y la comunicación, junto con la Internet, han influido en la forma en que se gestionan los roles en las empresas, independiente del sector productivo al que pertenezcan; y como consecuencia de ello, aparecieron nuevos escenarios de negocios, sirviendo a los consumidores como fuente de información de productos y servicios en los sitios web. Entonces, aparece la programación web como una herramienta que permite diseñar y desarrollar sitios web, donde las organizaciones amplían su presencia y aprovechan las bondades que el ciberespacio les brinda.

La programación web tiene principalmente una arquitectura conformada por un cliente, un servidor y una base de datos. En lo que respecta a la programación del lado del cliente, se ejecuta en el computador del usuario codificando las páginas web con HTML5, CSS3 y JavaScript. Para su ejecución, se puede hacer uso de un navegador web como Chrome, Mozilla, etc; siendo una aplicación capaz de interpretar las órdenes recibidas en forma de código HTML fundamentalmente, y convertirlas en las páginas web que son vistas por los usuarios en la pantalla del computador.

Con respecto a HTML5, es un lenguaje de marcas, que, mediante el navegador, visualiza la estructura de una página web, la cual puede incluir: texto, imágenes, videos, audios, entre otros.

Para dar formato a un sitio web con márgenes, tipos de letra, fondos y colores, se utiliza CSS3, que, si en algún momento se requiere cambiar algún elemento del estilo, automáticamente se actualizará todas las páginas vinculadas del sitio web. Por otro lado, si gusta ejecutar los controles de los valores del formulario y enviar alertas al navegador del usuario, se utiliza JavaScript.

El otro elemento de la programación web corresponde a la programación del lado del servidor tecnológico que permite el procesamiento de una petición de un usuario mediante la interpretación de un script en el servidor web para generar páginas HTML. Para este proceso, se hace uso del "apache" como servidor web; y como lenguaje de programación, se utilizó PHP, para la creación de páginas web dinámicas que aprovechan la gestión de bases de datos.

Así pues, esta obra se compone de seis capítulos, los cuales cada uno de ellos empieza con mención a la temática a desarrollar, por ejemplo: una introducción, un mapa mental haciendo alusión a la temática central, el desarrollo de los temas y subtemas; finalmente, unas lecturas recomendadas, unas preguntas de revisión de conceptos, ejercicios propuestos y unas referencias bibliográficas. Su distribución es la siguiente:

- **Capítulo 1:** Fundamentos de la programación web.
- **Capítulo 2:** Fundamentos de HTML5.
- **Capítulo 3:** Fundamentos de CSS3.
- **Capítulo 4:** Fundamentos de JavaScript.
- **Capítulo 5:** Programación del lado del servidor.
- **Capítulo 6:** Programación con capa de persistencia (Base de datos).

Este texto está conformado por la programación de lado del cliente y del servidor. Se incluyen elementos de buenas prácticas, ejemplos; también, se plantean actividades que permiten complementar el proceso de aprendizaje de la programación web.

De esta manera, el estudiante interesado en la temática recibirá toda la fundamentación necesaria que le permitirá comprender y abordar de mejor manera un proyecto relacionado con la programación web con herramientas de software pertinentes y posibilidad de escalamiento.

Miguel Hernández Bejarano

Ingeniero de sistemas de la Universidad Autónoma de Colombia, con especialización en diseño y soluciones telemáticas de la misma Universidad, Magíster en comercio electrónico del Instituto

Tecnológico de Monterrey (México), Magister en seguridad informática y Magíster en Ingeniería de Software de la Universidad Internacional de La Rioja (España), cursando estudios de Doctorado en Ingeniería de Sistemas e Informática. Profesor Investigador y Coinvestigador de varios proyectos de investigación; con reconocimiento de la Sociedad Latinoamericana de Ciencia y Tecnología en el 2015-2019.

Luis Eduardo Baquero Rey

Graduado como Ingeniero de Sistemas de la Universidad Autónoma de Colombia, con maestría en Auditoría de Sistemas y Computación de la Universidad Santo Tomás (Colombia), y en Seguridad Informática de la Universidad Internacional de La Rioja (España). Actualmente cursa su segundo año del doctorado en Ingeniería de Sistemas e Informática en la Universidad de Zaragoza (España), Profesor investigador y evaluador reconocido por Minciencias.

Capítulo 1

1. Fundamentos de la Programación Web

1.1. Temática a desarrollar

- TCP/IP
- Ambientes para el desarrollo de aplicaciones web

1.2. Introducción

En el contexto de la programación web, es conveniente abordar los los siguientes temas:

- Fundamentos de TCP/IP (siglas de Protocolo de Control de Transmisión/Protocolo de Internet) correspondientes al sistema de protocolos que hacen posible los servicios entre computadores de diferentes redes.
- El protocolo de transferencia de hipertexto (HTTP), es un protocolo de red utilizado para publicar páginas web.
- Los navegadores web como, por ejemplo, Internet Explorer (el cual fue reemplazado por Microsoft Edge), Google Chrome, Mozilla Firefox, Opera, Safari, los cuales son programas de software que permiten presentar textos, imágenes, videos, sonidos en páginas web, que se almacenan en la Internet o en una red local.

TCP/IP es el nombre del conjunto de protocolos de comunicaciones que se usan para gestionar los componentes que están conectados a una red de computadoras y son los protocolos base de la Internet.

Dentro de este conjunto de protocolos están HTTP, sobre la cual está fundamentada la WWW, y el protocolo HTTPS garantizando que

el navegador está usando un esquema seguro para proteger la información que está siendo transferida. Este protocolo es utilizado en todo tipo de transacciones en Internet.

Por su parte, el navegador web o browser es un programa que permite visualizar la información contenida en una página web, ya sea que esté en la Internet o en una red local, siendo su trabajo visualizar código HTML.

En un browser se pueden observar: imágenes, vídeos; y trabajar con hipertextos, los que en su mayoría están en lenguaje HTML, el cual es el lenguaje de visualización por excelencia. La figura *-1-* muestra de manera detallada (mediante un mapa mental), los aspectos relacionados con los fundamentos de la programación del lado del cliente, y todo lo que ello implica.

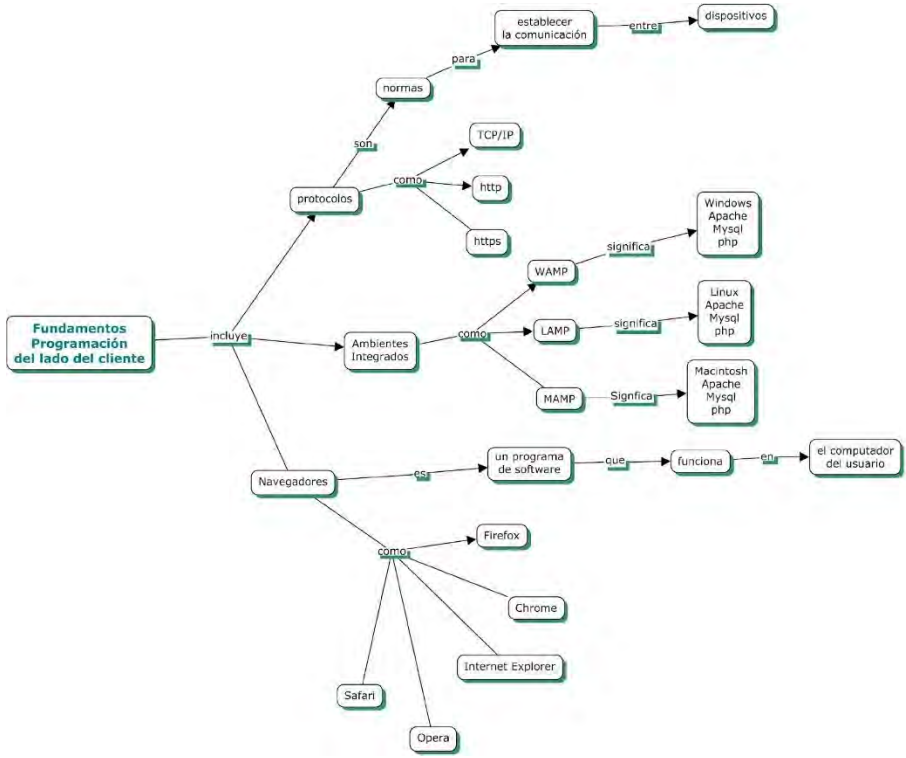


Ilustración 1: Mapa mental- Programación del lado del cliente.

1.3. Fundamentos de TCP/IP

Un protocolo de comunicaciones es un conjunto de normas que utilizan los equipos y dispositivos de informática (computadoras, teléfonos, enrutadores, switches, entre otros), para gestionar el intercambio de información, el formato, sincronización, secuencia y control de errores. Es por esto que los equipos de diferentes marcas se pueden comunicar sin problemas cuando hacen uso del mismo protocolo de comunicaciones.

TCP/IP es la denominación utilizada para identificar al grupo de protocolos de comunicaciones utilizados para gestionar los componentes conectados en red y que respaldan a la Internet, haciendo posible la transferencia de datos entre redes de computadores. TCP/IP hace referencia a los dos protocolos más trascendentes de este grupo: el conocido como Protocolo de Control de Transmisión (TCP) y el llamado Protocolo de Internet (presentado con la sigla IP). El protocolo más usado en Internet es el TCP/IP.

TCP, o Transmission Control Protocol, el cual se asegura de que los paquetes lleguen correctamente a su destino. Si TCP determina que un paquete no ha sido recibido, intentará volver a enviarlo hasta que sea recibido correctamente.

IP (Internet Protocol), es la especificación que determina hacia dónde son encaminados los paquetes, en función de su dirección de destino.

1.3.1. Protocolos HTTP y HTPPS

HTTP son las siglas en inglés de HyperText Transfer Protocol (protocolo de transferencia de hipertexto). Es un protocolo de red para publicar páginas web o HTML. HTTP es la base sobre la cual está fundamentado Internet o la WWW.

El protocolo HTTP funciona a través de solicitudes y respuestas entre un cliente, el cual puede ser un navegador de Internet y un servidor, que es la computadora donde residen páginas web. A una secuencia de estas solicitudes se le conoce como "sesión de HTTP".

HTTPS en lugar de HTTP, significa que el navegador está usando un esquema seguro para proteger la información que está siendo

transferida. Este esquema HTTPS es el que toda transacción comercial en Internet debe usar dadas las consideraciones de seguridad que se deben tener. A este esquema se le conoce como SSL, el cual, según las siglas en Inglés, se denomina Secure Socket Layer (capa de conexión segura), y es un protocolo criptográfico empleado para realizar conexiones seguras.

1.4. Web browser

El navegador puede considerarse como una interfaz de usuario universal. Dentro de sus funciones están la petición de páginas web, la representación adecuada de sus contenidos y la gestión de los posibles errores que se puedan producir. Un navegador web es un programa de software que permite visualizar la información que contiene una página web, la cual puede estar alojada en un servidor dentro de la World Wide Web, o en un equipo local.

La función de un navegador web es permitir la visualización de documentos de texto, los cuales pueden incluir imágenes, sonido, videos, hipervínculos, entre otros.

1.5. Web server

Un servidor de páginas web es un programa que permite acceder a páginas web alojadas en un computador. Presta servicios web, como: visualizar páginas web y servir de intermediario entre el cliente y el servidor. Ejemplos:

- **Internet Information Server (IIS):** Es el servidor de páginas web de la plataforma Windows.
- **Apache Server:** Es el servidor de páginas web más utilizado, de distribución libre y de código abierto.

1.6. Response and Request

La necesidad de realizar páginas web que respondan a estímulos de usuario se logra por medio de solicitudes al servidor web; en este sentido, el cliente (browser) inicia la comunicación por medio de una petición (request), donde el servidor responde (response), enviando los datos pedidos por el cliente. Los métodos para poder enviar información son:

- **GET:** Enviar una petición al servidor utilizando la URL.

- **POST:** Enviar datos al servidor utilizando un vector con las variables que se quieren transmitir, por medio de la comunicación interna entre las máquinas.

Los tipos de datos que envía el servidor dentro de un response como respuesta a una petición pueden ser muy variados, pero entre los más conocidos están: text/html, text/plain, image/gif, text/xml.

Para obtener el contenido de una página web, la comunicación cliente/servidor funciona por "ciclos" REQUEST-RESPONSE.

1.7. Ambientes integrados para el desarrollo de aplicaciones web

Un entorno de desarrollo integrado (IDE- Integrated Development Environment) es una aplicación de software que ofrece servicios integrales a los programadores de aplicaciones de software. Un IDE generalmente está integrado por todas o la mayoría de las siguientes herramientas:

- Un editor de texto.
- Un compilador.
- Un intérprete.
- Herramientas de automatización.
- Un depurador.
- Posibilidad de ofrecer un sistema de control de versiones.
- Factibilidad para ayudar en la construcción de interfaces gráficas de usuario. Acceso a la documentación del lenguaje de programación, ya sea interna o en web.

Algunos de los ambientes integrados más utilizados son:

WAMP

Es el acrónimo usado para describir un sistema de infraestructura de internet que usa las siguientes herramientas:

- Windows, como sistema operativo.
- Apache, como servidor web.
- MySQL, como gestor de bases de datos.
- PHP (generalmente), Perl, o Python, como lenguajes de

programación.

El uso del integrado WAMP permite servir páginas HTML a internet, además de poder gestionar datos en ellas, dadas las posibilidades de trabajo con MySQL y un lenguaje de programación.

LAMP

Esta aplicación es de código abierto, libre para desarrollar e implementar páginas web con persistencia en una base de datos. Está formada por un único servidor web, un servidor de base de datos, un servidor de archivos, una puerta de enlace para el acceso de redes externas y una puerta de enlace de entrada para el acceso de los usuarios. Permite integrar:

- Linux, como sistema operativo.
- Apache, como servidor web.
- MySQL, como gestor de bases de datos.
- PHP (generalmente), Perl, o Python, como lenguajes de programación.

MAMP

Aplicación de software que brinda a los desarrolladores los cuatro elementos necesarios para un servidor web:

- El sistema operativo (Macintosh),
- El manejador de base de datos (MySQL),
- El software para servidor web (Apache),
- El software de programación script web (PHP, Python o PERL).

FAMP

Entorno que brinda a los desarrolladores los cuatro elementos necesarios para un servidor web:

- Un sistema operativo (FreeBSD),
- Un manejador de base de datos (MySQL),
- Un software para servidor web (Apache),
- Un software de programación script web (PHP, Python o PERL).

El diseño y desarrollo de aplicaciones web consiste en implementar las necesidades, objetivos o ideas en Internet, utilizando las tecnologías más pertinentes según el proyecto. Las aplicaciones web pueden ser de acceso público como: tiendas virtuales, diarios digitales, portales de Internet, o de acceso restringido; como son las intranets utilizadas para mejorar la gestión interna de las empresas, como el reporte de horas del personal, gestión de proyectos y gestores documentales; o el uso de extranets para aumentar y mejorar el servicio con sus distribuidores, clientes, proveedores, comerciales y colaboradores externos.

1.8. Navegadores y herramientas para desarrolladores

Internet Explorer

Internet Explorer o IE es un navegador web gratuito desarrollado por Microsoft. Funciona en el sistema operativo Windows, aunque ya está terminando su vida útil y se va a reemplazar con EDGE, el cual tiene las mismas características de IE. Las principales características de Internet Explorer son:

- Navegación por pestañas.
- Capacidad de búsqueda en diferentes motores.
- Aceleradores que permiten, mediante un menú contextual, ejecutar acciones repetitivas de una forma rápida.
- Web Slices.
- Modo de compatibilidad.

Chrome

Es el navegador desarrollado por Google. Es un software gratuito para navegar por las páginas web, actualmente está disponible sólo para Windows; también existe una versión disponible para dispositivos móviles, sistema operativo Linux. Entre sus características están:

- Tiene pocos botones, barras o menús. De hecho, la barra de herramientas se encuentra en lo más alto de la pantalla y así deja a la vista casi todo el contenido de la página que se visita.
- Tiene un corrector de ortografía nativo y de un español bastante adelantado frente al resto de navegadores.
- Cuenta con un sistema de pestañas que permiten tener varias páginas disponibles en una sola ventana.

Mozilla Firefox

Navegador de código abierto desarrollado por la fundación Mozilla. En sus comienzos, tuvo varios nombres, tales como Phoenix y Firebird. Ya con su nombre actual, Firefox es abreviado en forma oficial como FF, así como también, Fx o fx2. Es reconocible por su logo de un zorro. Las características principales son:

- Es un proyecto "Open Source", en español, "Código Abierto", lo que quiere decir que todo desarrollador puede modificar el código y así poder mejorarlo.
- Es multiplataforma, es decir, que se encuentra disponible en diferentes versiones y así es posible utilizarlo con Microsoft Windows, Mac OS X, y también con GNU/Linux.
- Cuenta con una protección integrada contra algunos virus y la aparición de ventanas pop-up.
- Mozilla integra una serie de aplicaciones como correo electrónico (email), foros de noticias (newsgroups), charla (chat), y un medio para la elaboración de páginas web, o sea un editor HTML (llamado Composer).

Opera

Es un navegador desarrollado por Opera Software. Puede ser instalado en Mac, Windows, FreeBSD, Linux y Solaris. Opera está disponible en varios idiomas. Como características se tienen:

- Es una suite de Internet que integra principalmente un navegador web moderno y respetuoso con los estándares web de la W3C. Incluye numerosas funcionalidades, como el bloqueo de ventanas emergentes y el reconocimiento de gestor del ratón.
- Desde la versión 8.5, Opera se distribuye gratuitamente y sin publicidad alguna.

Safari

Es un navegador web de código cerrado desarrollado por Apple Inc. El cual está disponible para Mac OS X, iOS (sistema usado por el iPhone, el iPod Touch y el iPad) y Windows. Entre sus características están:

- Incluye navegación por pestañas.
- Corrector ortográfico.
- Incluye búsqueda progresiva.
- Desde Safari puedes twittear páginas web, subirlas a Facebook o compartirlas vía Mail o Mensajes. Viene con muchas más funcionalidades adicionales, así que navegar solo es el principio.
- Al escribir una dirección web, Safari lo lleva directamente a la página correspondiente, e incluso rellena toda la URL.

Avant Browser

Está basado en el motor de Internet Explorer, tiene compatibilidad con la mayoría de las páginas visitadas, ofrece bloqueo de ventanas emergentes y un filtro anti-publicidad. También cuenta con la ayuda de la navegación por pestañas. Otra interesante opción (para seguridad de los usuarios), es la posibilidad de deshabilitar todo tipo de elementos como:

- ActiveX.
- Scripts.
- Applets Java.
- Imágenes.
- Sonidos.
- Animaciones Flash.
- Videos.

Crazy Browser

Es un navegador web de apariencia similar a Internet Explorer, pero con algunas funciones avanzadas. La característica más importante de este programa es la posibilidad de realizar diferentes acciones con sólo combinar clics y movimientos de ratón. Una utilidad de interés es la existencia de una lupa con la que es posible aumentar el tamaño del texto y de los elementos presentes en una determinada página.

Plugins navegadores

Los plugins o complementos (add-ins o plug-ins) para los navegadores añaden funciones que facilitan el desarrollo de las páginas

web, como consultar las propiedades CSS que tiene un elemento de la página web, la utilización de algún servicio en Internet, como la validación del código HTML o CSS de una página, con un clic del mouse.

1.9. Lecturas recomendadas

- Los protocolos de TCP/IP que existen en sus diferentes capas y la aplicación de estos.
- Las características y la aplicación del servidor Apache.
- Las herramientas ActiveX y Scripts, y entornos de desarrollo.

1.10. Preguntas de revisión de conceptos

- ¿Qué es la WWW?
- ¿Qué es un entorno integrado de desarrollo?
- ¿Qué es un browser?
- Diferencia entre http y https.
- ¿Cuáles son las formas que se utilizan para realizar la comunicación entre dos máquinas a nivel de web?

1.11. Ejercicios propuestos

1) Realizar una comparación resaltando las principales características de los siguientes servidores web:

IIS	Apache	Nginx	Tomcat

Tabla 1:Ejercicios características de los servidores web.

Browser	Características	Versión actual	Observación
FireFox			
Internet Explorer			
Safari			
Opera			
Google Chrome			
Netscape			
Spacetime			
FineBrowser			
The World Browser			
Maxthon			
SmartBro			
SeaMonkey			
Browzar			
Aurus			
Avant Browser			
Green Browser			

Tabla 2: Ejercicios navegadores web.

2) Completar la siguiente tabla para los navegadores web mencionados, incluyendo en el espacio de observación aspectos relacionados como el tipo de licenciamiento:

3) Establecer las principales características, semejanzas y diferencias de los entornos de desarrollo integrado que a continuación se relacionan.

Entorno	Características	Semejanzas	Diferencias
MAMP			
WAMP			
LAMP			
FAMP			

Tabla 3: Ejercicios semejanzas y diferencias.

4) Consultar características relevantes de los lenguajes de programación PHP, Python y PERL y sus aplicaciones en la web.

Sitios web de interés

- http://www.w3schools.com/html/html5_intro.asp
- http://www.w3.org/TR/2011/WD-html5-20110525/

Bibliografía

Capítulo 2

2. Fundamentos de HTML5

2.1. Temática a desarrollar

- HTML
- HTML5

2.2. Introducción

En este capítulo se abordan los fundamentos de HTML5, como lenguaje de marcas que incluyen una serie de etiquetas semánticas para brindar mayor sentido a las páginas web como: Header, Footer, Aside, Article, Nav y Section. Otros temas que son tenidos en cuenta son los atributos globales, el uso del selector api, herramientas de desarrollo y depuración.

En el desarrollo del tema, se plantean actividades que permiten la integración de las marcas, también llamadas "TAG de HTML5" en la construcción de páginas web.

HTML es un lenguaje de marcas que está conformado por una serie de etiquetas que interpretan los navegadores (como los mencionados en el capítulo anterior) y que permiten visualizar la información de manera organizada con los atributos que se requieren frente al usuario final.

HTML5 es la actualización de HTML, y que al igual que su predecesor, permite agrupar las tecnologías de desarrollo de aplicaciones web como HTML básico, HTML5, hojas de estilo en cascada (CSS3) y Javascript.

HTML5 incluye nuevos elementos semánticos que describen el significado o propósito de la página al navegador y desarrollador dentro de los que están:

- **Article:** Contenido autónomo que puede existir sin el resto del contenido.
- **Aside:** Contenido que es enlazado levemente con el resto del contenido de la página. Figcaption: Identifica la descripción de una figura
- **Figure:** Representa una figura enlazada a un documento.
- **Footer:** Pie de página donde se ubica información legal y de autoría de la página.
- **Header:** Define la cabecera de una página o de una sección en la cual se deben generar cierto tipo de contenido.
- **Hgroup:** Define el header de una sección.
- **Mark:** Define referencias a secciones o a otros documentos.
- **Nav:** Define una sección con enlaces de navegación.
- **Section:** Define una sección dentro de un documento.
- **Time:** Define una variable de fecha y tiempo.

En la figura *-2-* se detalla el concepto de HTML5 de una manera más somera.

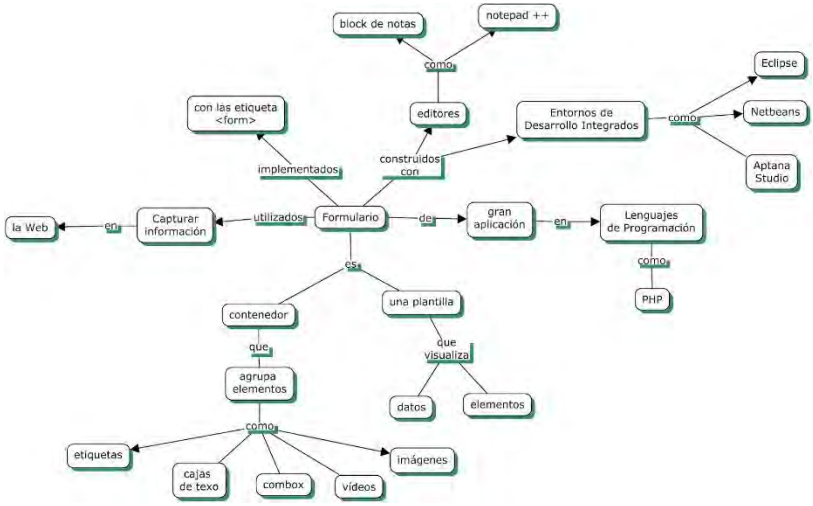


Ilustración 2: HTML5.

2.3. Aspectos generales de HTML5

HTML5 -HypertText Markup Language- (Lenguaje de Marcado de Hipertexto) versión 5, es la nueva revisión del lenguaje HTML. Esta revisión, como el lenguaje mismo, fue desarrollada por el Consorcio W3C, y aún está en modo experimental, aunque ya es bastante popular en la web.

HTML5 (el lenguaje de marcado con el que se crea la estructura y el contenido de las páginas web), es una nueva versión que intenta aumentar la potencia del lenguaje para conseguir aplicaciones web cada vez más complejas. Como su nombre indica, HTML5 es el sucesor de HTML4, el cual inició a finales de 2003 con un grupo de trabajo que se propuso hacer un lenguaje que llegara con un conjunto de tecnologías que permiten construir la nueva web. No fue sino hasta 2007 que el HTML5 del W3C aceptó la visión mediante la incorporación en ella del grupo de trabajo.

Esta versión de HTML conduce a una fusión entre JavaScript (como lenguaje de programación), HTML (como modelo semántico) y CSS3 (que es la evolución del CSS como el lenguaje de los estilos), que se dedica a dar un mejor aspecto gráfico a los proyectos. Con este fin, se añaden etiquetas para reproducir multimedia (video y audio), nuevas etiquetas de marcado semántico, nuevos tipos de campos de entrada de datos en formularios y también se define el DOM (Document Object Model – Modelo de Objeto de documento) con nuevas funcionalidades y sus respectivas API para añadir características tales como: geoposicionamiento, dibujo, arrastrar y soltar, entre otros.

HTML5 no pertenece a una empresa o un navegador específico. Se ha desarrollado por una comunidad de usuarios interesados en la evolución de la web y de un consorcio de líderes tecnológicos que incluye Google, Microsoft, Apple, Mozilla, Facebook, IBM, HP, Adobe, y muchos otros.

2.4. Etiquetas HTML5

<! DOCTYPE>

Esto es lo primero que se encuentra en un documento web; luego,

es necesario agregar la declaración <!DOCTYPE html> a los documentos HTML, para que el navegador identifique qué tipo de documento debe esperar. El DOCTYPE se activa en HTML5 en todos los navegadores que tienen un modo estándar y le informa al navegador la versión de HTML a utilizar.

Aunque gran parte de la atención que se tiene sobre HTML5 gira en torno a los nuevos elementos y etiquetas semánticas que se pueden utilizar en páginas estáticas, a continuación, se presentan algunas formas de utilización de estas.

Una estructura html5 básica (que es lo mínimo que se requiere para tener una correcta base para iniciar cualquier proyecto), es como la siguiente:

```
<!DOCTYPE html>
        <html lang="es">
        <head>
            <meta charset="utf-8"/>
                <meta name="description" content="html5" />
                <title>Mi primera página en html5</title>
        </head>
        <body>
         <!--aqui se despliega todo el contenido-->
        </body>
</html>
```

En la primera línea está la etiqueta DOCTYPE que permite definir un lenguaje HTML como base para el navegador. En la segunda línea aparece la etiqueta HTML con un atributo lang = "es" que indica el lenguaje de la página web construida, especificando que se va a trabajar con español. La etiqueta meta permite definir metadatos y para este caso, plantea un conjunto de caracteres utf-8, con contenido HTML5 y "description" como nombre.

En los siguientes ejemplos se muestran programas donde se involucra el uso de etiquetas HTML5.

Ejemplo 1: Primera página.

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Título Mi primera página</title>
    </head>

    <body>
        Mi primera pagina
    </body>
</html>
```

Ejemplo 2: Codificación de caracteres.

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Codificación de caracteres</title>
    </head>
    <body>
        <p> Si necesitamos escribir caracteres que utilizan el
        lenguaje HTML para definir etiquetas (&lt; &gt; &amp; &quot;
          &apos;) no se pueden utilizar libremente.</p>
    </body>
</html>
```

Ejemplo 3: Etiquetas de listas no ordenadas ul.

```
<!doctype html>
	<html>
		<head>
			<meta charset="utf-8">
			<title>Página con etiquetas</title>
		</head>
		<body>
			<h1>Titular nivel 1 </h1>
			<p>Párrafo de texto. <strong>negrita</strong> y
			<em>cursiva</em></p>
			<ul>
				<li>Elemento de la lista 1</li>
				<li>Elemento de la lista 2</li>
				<li>Elemento de la lista 3</li>
			</ul>
		</body>
	</html>
```

Ejemplo 4: Etiquetas de listas ordenadas y no ordenadas.

```
<!doctype html>
<html>
	<head>
		<meta charset="utf-8">
		<title>Listas</title>
	</head>
	<body>
		<h1>Listas</h1>
		<ul>
			<li>Elemento 1 </li>
			<li>Elemento 2</li>
		</ul>
		<ol>
			<li>Elemento ordenado 1</li>
			<li>Elemento ordenado 2</li>
		</ol>
	</body>
</html>
```

Ejemplo 5: Etiquetas de maracas de texto.

```
<!doctype html>
	<html>
	<head>
		<meta charset="utf-8">
		<title>Marcas de texto</title>
	</head>
	<body>
		<h1>Titular nivel 1</h1>
		<p>En este parrafo hay texto marcado como <em>importante</em>
		y otro texto marcado como <strong> muy importante.
		</strong></p>
		<p>Este es el segundo párrafo.</p>
	</body>
	</html>
```

Ejemplo 6: Etiqueta blockquote.

```
<!doctype html>
	<html>
	<head>
		<meta charset="utf-8">
		<title>Marcas de texto</title>
	</head>
	<body>
		<p>Ejemplo de la etiqueta <strong>blockquote</strong>
		incluyendo el atributo <em>cite</em>. </p> <blockquote
		cite='http://www.google.com...def-BLOCKQUOTE'> El elemento
		<strong>BLOCKQUOTE</strong>, se usa para escribir una cita
		textual o un párrafo exacto y que este se diferencie del
		resto del texto.
		El atributo <em>cite</em>, define la URL del documento o
		mensaje original, da información sobre la fuente donde se
		extrajo la cita.
		</blockquote>
	</body>
	</html>
```

Ejemplo 7: Etiqueta de preformateo de texto pre.

```
<!doctype html>
	<html>
	<head>
		<meta charset="utf-8">
		<title>Marcas de texto</title>
	</head>
	<body>
		<p>Ejemplo de texto preformateado:</p>
		<pre>
			dato	78	55
			dato	23	63
			dato	44	 6
		</pre>
	</body>
	</html>
```

Ejemplo 8: Etiqueta de abreviaturas.

-

```
<!doctype html>
	<html>
	<head>
		<meta charset="utf-8">
		<title>Marcas de texto ABBR</title>
	</head>
	<body>
		Bienvenido al uso de las abreviaturas de la <abbr lang="es"
		title="Departamento de Ingeniería de Sistemas
		">DIT</abbr>.
	</body>
	</html>
```

Ejemplo 9: Etiquetas de cabecera h1…h6 y salto de línea con br; <h1>...</h1>, <h6>…</h6>...Etiqueta titulares, las cuales son utilizadas para dividir el texto en secciones. Se pueden definir seis niveles de titulares, siendo <h1>el de mayor tamaño y el <h6> el de menor tamaño. El texto que es "título" se ubicará entre los elementos <h1> Título </h1>.

... : Párrafos. Etiqueta de párrafo la cual agrupa texto y sentencias.

```
<!doctype html>
      <html>
      <head>
            <meta charset="utf-8">
            <title>br, saltos de li¬nea</title>
          </head>
         <body>
            <p>
                  <h1>Lenguajes de Programación</h1> <br />
                  <h2> Lenguaje de programación C </h2> <br />
                  <h3> Lenguaje de programación Java </h3> <br />
                  <h4> Lenguaje de programación C# </h2> <br />
                  <h5> Lenguaje de programación C </h2> <br />
            <h3> Lenguaje C </h2> <br />
            </p>
            <p><strong>Lenguajes más utilizados</strong></p>
      </body>
      </html>
```

Ejemplo 10: Etiquetas de subíndice y superíndice.

```
<!doctype html>
      <html>
      <head>
            <meta charset="utf-8">
            <title>Superindices y subindices</title>
      </head>
      <body>
            <p>La fórmula del agua es H<sub>2</sub>O</p>
            <p>La fórmula de Albert Einstein para la equivalencia entre
            masa y energía es E=mc<sup>2</sup>. Llamada teoría de la
            relatividad </p>
      </body>
      </html>
```

Ejemplo 11: Etiquetas para el manejo de tablas.

```
<!doctype html>
	<html>
	<head>
		<meta charset="utf-8">
		<title>Tabla</title>
	</head>
	<body>
		<table>
			<tr>
				<th> </th>
				<th>Poblacion</th>
				<th>Hombres</th>
				<th>Mujeres</th>
			</tr>
			<tr>
				<th>Alemania</th>
				<td>82.020.578</td>
				<td>40.346.853</td>
				<td>41.673.725</td>
			</tr>
			<tr>
				<th>Francia</th>
				<td>65.578.819</td>
				<td>31.764.615</td>
				<td>33.814.204</td>
			</tr>
			<tr>
				<th>Inglaterra</th>
				<td>63.896.071</td>
				<td>31.423.339</td>
				<td>32.472.732</td>
			</tr>
		</table>
	</body>
	</html>
```

Ejemplo 12: Etiqueta de referencia y comentarios <!–>.

```
<!DOCTYPE html> <!--Indica HTML5-->
<html>
  <head><title>Ejemplo</title></head>
  <body>
    <h1>Ejemplo 1</h1>
      Ejemplo con un hiperenlace a
      <a href='http://google.com'>Google</a>
  </body>
</html>
Ejemplo 13. Etiquetas de cabeceras e id
<html>
  <head>
    <meta charset="utf-8">
    <title>Ejemplo</title>
  </head>
  <body>
    <h1>Ejemplo 2</h1>
    <h3 id='seccion1'>Seccion 1</h3>
    ... texto de la Seccion 1  ...
    <h3 id='seccion2'> Seccion 2</h3>
    ... texto de la seccion 2  ...
    <h3 id='seccion3'>Seccion 3</h3>
    ... texto de la seccion 3  ...
  </body>
</html>
```

2.5. Etiquetas de marcado semántico

Los elementos semánticos describen su significado o propósito claramente al navegador y al desarrollador. Por ejemplo, la etiqueta define una división o una sección de un documento HTML, pero no nos dice nada acerca de su contenido o no tiene sentido alguno el hacer dicha definición. Algunas de las más importantes etiquetas introducidas en HTML5 son las que añaden valor semántico y estructural, es decir, indican de forma inequívoca las distintas partes de las que consta una página, una cabecera, un pie de página, la navegación, etc. Algunos de los elementos semánticos nuevos en HTML5 son:

Header: La etiqueta <header> se usa para marcar un grupo de elementos de introducción o de navegación dentro de una sección o documento. Por lo general, se usa para incluir los encabezados, (<h1>–<h6> o <hgroup>), pero no es obligatorio. También puede contener otros elementos como el índice de una sección, un formulario de búsqueda y logos de importancia. El uso de la etiqueta <header> no es exclusivo por documento, sino que se puede hacer un header por cada sección del mismo, de esta forma:

```
<header>
      <h1>Título del documento</h1>
      <p>Más información</p>
</header>

<article>
      <header>
            <h1>Título del artículo</h1>
            <p>Autor del artículo</p>
      </header>
      <p>Texto del artículo</p>
</article>
```

La etiqueta <header> se recomienda utilizarla para marcar la cabecera de una página que contiene el logotipo del sitio, una imagen, un cuadro de búsqueda, entre otros elementos.

Footer: La etiqueta <footer>se utiliza para indicar el pie de una sección o de un documento. Un pie de página contiene información general acerca del autor, enlaces a documentos relacionados, datos de derechos de autor, licencias, términos de uso, entre otros.

```
<footer>
        Footer - Copyleft 2019
</footer>
```

Article. Esta etiqueta se utiliza para marcar un contenido que es independiente, el cual tendría sentido fuera del contexto de la página actual, como por ejemplo una noticia, un artículo en un blog o un comentario. Cuando se anidan las etiquetas <article>, los artículos internos están relacionados con el contenido del artículo exterior. Por ejemplo, una entrada de blog en un sitio que acepta comentarios.

Normalmente un <article> tiene, además de su propio contenido, una cabecera <header>y posiblemente un pie (<footer>).

Ejemplo:

```
<article>
        <header>
                <h1>Título del artículo</h1>
        </header>
                <p>Texto correspondiente al artículo</p>
        <footer>
                Footer - Copyleft 2019
        </footer>
</article>
```

Nav: La etiqueta <nav>representa una parte de una página que enlaza a otras páginas o partes dentro de la página. Es utilizada para marcar una sección del documento cuya función es la navegación por la página web.

Se recomienda que esta etiqueta no sea utilizada para marcar todos los grupos de enlaces, únicamente los bloques principales de navegación de la página. Por ejemplo, los típicos enlaces que hay en el pie de página no se deben marcar con <nav>.

```
<nav>
  <ul>
    <li><a href="#">inicio</a></li>
    <li><a href="#">blog</a></li>
    <li><a href="#">Galería</a></li>
    <li><a href="#">Ayuda</a></li>
  </ul>
</nav>
```

li: El tag "li" define un elemento dentro de una lista, ya sea ordenada (ol) o desordenada (ul). En HTML5, los tags li son los elementos que contiene el contenido del índice. A continuación, se muestra un ejemplo:

Lugares de Bogotá:

- Chapinero
- Fontibón
- Suba

Section: La etiqueta <section> es utilizada para marcar una sección genérica de un documento o aplicación. Una sección, en este contexto, es una agrupación temática del contenido, típicamente con un encabezado.

El uso de <section> se puede presentar en los capítulos, las pestañas, en un menú tabulado o, en la página principal de un sitio web, la introducción, la lista de noticias e información de contacto. Sólo se debería usar <section> para contenido independiente al que se pondría en un encabezado y que no sea susceptible de ir marcado con <article>, <aside>o <nav>.

Ejemplo:

```
<h1>Título de la página</h1>
<section id="news-list">
<h2>Noticias</h2>
<article>
        <h3>Noticia 1</h3>
        <p>Desarrollo de la noticia 1</p>
</article>
<article>
        <h3>Noticia 2</h3>
        <p>Desarrollo de la noticia 2</p>
</article>
</section>
```

Una página de inicio de un sitio web puede estar dividida en secciones, por ejemplo; la introducción, las noticias, la información de contacto, entre otros.

Aside: La etiqueta <aside> representa una sección de una página que consiste en el contenido, pero que no es parte del mismo, aunque sí está relacionado con el contenido que le rodea. Estas secciones son a menudo representadas como barras laterales o como inserciones, y contienen una explicación al margen como una definición de glosario, enlaces a páginas relacionadas (como publicidad, la biografía del autor), o en aplicaciones web, la información de perfil a blogs relacionados.

```
<aside>
    <h3>Lugares turísticos de Bogotá</h3>
    <p>Monserrate</p>
    <p>La Candelaria</p>
</aside>
```

Hgroup: La etiqueta <hgroup> se usa para agrupar un conjunto de uno o más elementos de encabezado (<h1>–<h6>). El uso más común de la etiqueta <hgroup> es para agrupar el título de la página con su eslogan.

Ejemplo:

```
<hgroup>
        <h1>Nombre de la compañía</h1>
        <h2>Eslogan de la compañía</h2>
</hgroup>
```

```
<header>
  <hgroup>
    <h1>Mi Programación Web</h1>
    <h2>La programación de lado del cliente. </h2>
  </hgroup>
  …
</header>
```

Figcaption: Es un tag complementario a <figure> que nos permite asignar un título a la imagen que va dentro de este elemento. Podemos situarlo sobre la imagen o bajo ella.

Figure: El tag <figure> permite delimitar un contenedor para imágenes, gráficas, entre otros. Se complementa con la marca <figcaption>, se le puede añadir un título a la imagen o gráfica mostrada. Un uso habitual de este elemento se da en el caso cuando se requiere una imagen con título.

Ejemplo:

```
<header>
 <h1><!-- El titulo de la web --></h1>
  <figure>
        <!-- Aqui va logo -->
        <figcaption><!-- Titulo del logo --></figcaption>
    </figure>
</header>
```

Los comentarios se utilizan para documentar de manera donde se explica qué se está haciendo con el código en las páginas web. Los

comentarios pueden ser de una sola línea.

```
<!-- Comentario en una página HTML5 -->
```

Un comentario puede tener varias líneas, teniendo en cuenta donde inicia y donde termina con los caracteres que lo delimitan.

```
<!--
     Comentario
     en varias
     líneas
-->
```

2.6. Estructura HTML5

La estructura básica para realizar un sitio web en HTML5 es la siguiente:

DOCTYPE: Etiqueta para indicar la versión, seguida de "html", la cual tiene el atributo lang para el idioma del sitio.

```
<!DOCTYPE html>
   <html lang="es">
   <head>
```

Head: Etiqueta <head>se utiliza para agrupar información sobre ella, como puede ser el título. Está conformada por las etiquetas <head> y </head>.

Título: El título está construido por dos etiquetas: <title> Título de la web </title>.

Meta: La etiqueta META es donde se le indica al navegador cual es el tipo de codificado del documento html, incluyendo el idioma para que acepte cosas como: las tildes, la letra ñ, etc; y con mucha más compatibilidad, se utiliza el UTF-8. <meta charset=utf-8/>.

Body: La etiqueta body es el espacio donde se define el contenido principal o cuerpo del documento; es decir, se visualiza el texto de la página, las imágenes, los formularios, entre otros. Estos elementos pueden ubicarse entre las etiquetas <body> y </body>, que van debajo de las sentencias </head>.

Ejemplo:

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="utf-8" />
    <title>Estructura basica de una pagina web en HTML5</title>
</head>

<body>
        <header>
            <nav>
                 referencias

            </nav>
        </header>
        <!-- Cuerpo de la página -->
        <footer>
            <p>contactos </p>
        </footer>

</body>
</html>
```

2.7. Maquetado HTML5

Maquetado es el término asociado al proceso de construir páginas web haciendo uso de HTML5.

Section: También conocido como "contenido". Se incluye un "h2" de título y el contenido en etiquetas p dentro de una etiqueta llamada "article" para los artículos.

```
<section>
      <article>
            <h2>Titulo de contenido</h2>
            <p> Contenido (texto, imágenes, citas, videos, otros.) </p>
      </article>
</section>
```

Aside: Es la etiqueta que corresponde a la barra lateral, muy fácil de implementar con "h3" de título y p de contenido dentro de ella.

```
<aside>
      <h3>Titulo de contenido</h3>
      <p>Contenido</p>
</aside>
```

<h3></h3>: Es una etiqueta que presenta el texto de un título de menor tamaño que la marca <h1>.

Footer: Es otra de las etiquetas estructurales de HTML5. Puede ser el pie de un documento o de una sección o artículo.

```
<footer>
      <p>Contactos </p>
</footer>
```

Ejemplo de una página web en HTML5 que integra las etiquetas, representando como se realiza el proceso de maquetado:

```
<!DOCTYPE html>

<html lang="es">

<head>
  <title>Titulo de la web</title>
  <meta charset="utf-8" />
  <link rel="stylesheet" href="estilos.css" />
</head>

<body>
    <header>
       <h1>Mi Pagina web</h1>
       <p>Mi Pagina web creada en html5</p>
    </header>
    <section>
       <article>
           <h2>Titilo de contenido<h2>
           <p> Contenido (texto, imágenes, citas, videos entre
otros.) </p>
       </article>
    </section>
    <aside>
       <h3>Titulo de contenido</h3>
           <p>contenido</p>
    </aside>
    <footer>
        Creado para una comunidad
    </footer>
</body>
</html>
```

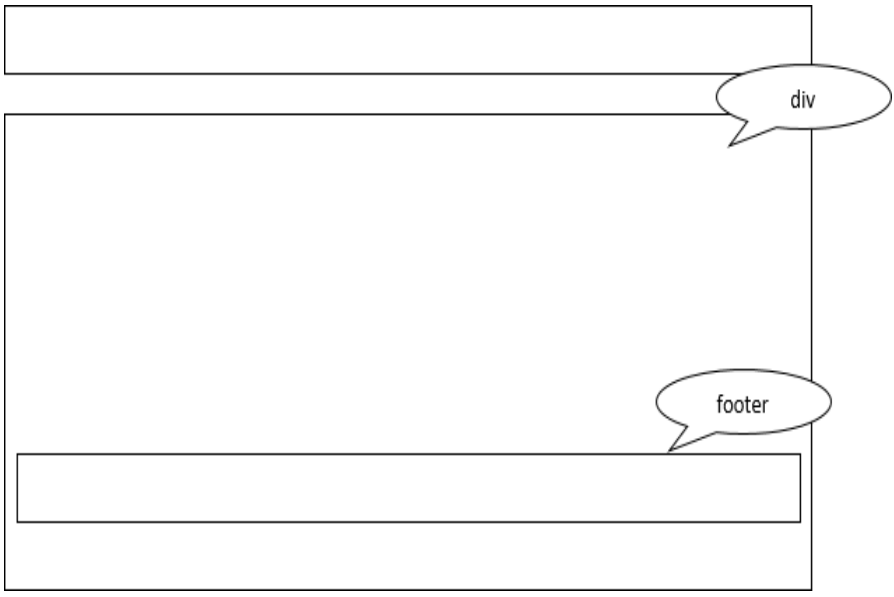
Otros ejemplos:

1. Página que tiene la siguiente representación.

```
<!DOCTYPE html>
<!-- programación web del lado del cliente -->
<!-- Maquetado Ejemplo 1-->
<html>
	<head>
		<title>Documento sin estilos</title>
	</head>
	<body>
		<header>
			<h1>Introduccion de HTML5</h1>
			<h2>header</h2>
			<h3>como le parece</h3>
		</header>
		 <div id="MyMainContainer">
			<h1>Aqui va el contenido de la div</h1>
		 </div>
	</body>
</html>
```

El <!DOCTYPE> no es una etiqueta, sino, una instrucción que se usa para especificar a los navegadores la versión de HTML en la que está escrito un documento.

2. Página web que tiene con la siguiente maquetación:



```
<!DOCTYPE html>
<!-- PROGRAMACION WEB DEL LADO DEL CLIENTE-->
<!-- MAQUETADO EJEMPLO 2-->
<html lang="es">
      <head>
            <meta charset="utf-8" />
            <title>Maquetado de una página web en HTML5</title>
      </head>
      <body>
      <header>
            <hgroup>
                  <h1>HTML5</h1>
                  <h2>Pagina Web </h2>
                  <h3>Maquetando</h3>
            </hgroup>
      </header>
       <div id="MyMainContainer">
                  <footer>
                        <h2>Gracias por su visita! </h2>
                  </footer>
        </div>
      </body>
</html>
```

3. Página web que integra las marcas div, header, hgroup, nav, article, section, aside y footer en la maquetación de la página:

```
<!DOCTYPE html>
<html lang="es">
    <head>
        <meta charset="utf-8" />
        <title>Titulo</title>
    </head>
    <body>
      <div id="centrado">
        <header>
            <hgroup>
                <h1> Título de la pagina </h1>
                <h2> descripción de la página web </h2>
            </hgroup>
            <div id="logotipo"> logotipo </div>
        </header>
        <id ="contenido">
            <nav>
            <ul>
              <li><a href="#">inicio</a></li>
              <li><a href="#">blog</a></li>
              <li><a href="#">Galería</a></li>
              <li><a href="#">Ayuda</a></li>
            </ul>
            </nav>
            <aside>
                 Información galería
            </aside>
            <section id="izquierda">
                <h2>Últimas Novedades<h2>
                <article> novedad uno  </article>
                <article> novedad dos </article>
                <article> novedad tres </article>
            </section>
            <section id="centro">
                <h2>Temas Importantes<h2>
                <article> Tema uno </article>
                <article>Tema dos</article>
                <article>Tema tres</article>
            </section>
          </div>
          <footer>
            <section  id="copyleft">  información  del  Copyletf
    </section>
            <section  id="contacto">  Información  de  contacto
    </section>
            <section  id="enlaces">  Mapa  sitio  Web,  Política
    </section>
          </footer>
        </div>
    </body>
</html>
```

2.8. Atributos Globales

Los atributos de HTML dan sentido y contexto a los elementos; en este sentido, permite generar mayor funcionalidad a las páginas a partir de la utilización de comunicaciones y de métodos de visualización.

Accesskey: Especifica una tecla de acceso directo para activar/hacer focus en un elemento.

Class: Especifica uno o más nombres de las clases de un elemento (se refiere a una clase en una hoja de estilo).

Contenteditable: Especifica si el contenido de un elemento es editable o no.

Contextmenu: Especifica un menú contextual para un elemento. El menú contextual aparece cuando un usuario hace clic en el elemento.

Dir: Especifica la dirección del texto para el contenido de un elemento.

Draggable: Especifica si un elemento es arrastrable o no.

Dropzone: Especifica si los datos arrastrados son copiados, movidos o vinculados, cuando se dejan caer sobre un elemento.

Hidden: Especifica que un elemento no es visible aún, o no es relevante.

Id: Especifica un único "id" a un elemento.

Lang: Especifica el idioma del contenido del elemento.

Spellcheck: Especifica si el elemento debe tener su ortografía y gramática comprobada o no.

Style: Especifica una línea de estilo CSS para un elemento.

Tabindex: Especifica el orden de tabulación de un elemento.

Title: Especifica información adicional acerca de un elemento.

2.9. Marcas o etiquetas para elementos audiovisuales

<audio> -Define un flujo de sonido, música o audio.

<video> -Define un recurso de video.

<source> -Se utiliza para especificar los recursos de medios múltiples para los elementos audiovisuales, como <video>y <audio>. También, permite especificar video alternativo / archivos de audio que el navegador puede elegir, en función de su tipo de medio o soporte de códec.

<embed> -Define un área donde se incluyen aplicaciones externas o contenido interactivo.

<track> -Específica pistas de texto para los elementos audiovisuales. En la actualidad, no es compatible con ninguno de los navegadores más importantes.

Uso lector API

Para el uso de las API, se requiere que los navegadores estén actualizados o haber descargado las últimas versiones que soporten la funcionalidad que se quiere utilizar. Algunos de los API más populares de HTML5 soportados en los navegadores de forma nativa son:

ClassList: El objeto classList brinda a los desarrolladores métodos para:

- **Add:** Añade un elemento.
  Agregar clase CSS:
  unDiv.classList.add('unaClase');

- **Remove:** Elimina el elemento.
  Remover clase:
  unDiv.classList. remove('unaClase');

- **Toggle:** Invierte el elemento (añade/elimina)
  Conmutar (toggle) clase:
  unDiv.classList.toggle('unaClase'); //se agrega la clase

```
unDiv.classList.toggle('unaClase'); //se remueve la clase
```

- **Contains:** Comprueba que exista el elemento.
  Chequear si un elemento contiene una clase CSS: unDiv.classList. contains('unaClase');

Geolocalización: El Polyfill Geolocation Shim usa el servicio de IP-Geocoding de Google como alternativa para los navegadores que no soportan esta API. El otro polyfill se llama "geolocation-javascript" el cual no usa el servicio del buscador, pero utiliza las API específicas de BlackBerry, WebOS y Google Gears. En la mayoría de los casos, no se deberían exponer características de geolocalización en la aplicación, si esta característica no está presente nativamente.

Historial (History): La API de historia provee a Javascript una forma de cambiar la URL mostrada en el navegador sin recargar la página. El plugin History.js suaviza algunas diferencias de implementación entre los navegadores y provee un hashchange alternativo para navegadores con HTML4.

LocalStorage y SessionStorage: Son una muy buena forma de guardar datos sin tener que utilizar cookies. Incluso Internet Explorer 8 soporta ambas.

WebSockets: Permite la construcción de aplicaciones en tiempo real, ya que brindan la posibilidad para agregar comunicación bidireccional en una conexión persistente para una aplicación, permitiendo incrementar la interactividad y el compromiso con el usuario.

2.10. Herramientas de desarrollo y depuración

La versión de HTML5 está acompañada de una nueva versión de las hojas de estilo CSS3, la cual se trata de nuevas posibilidades de formato, como, por ejemplo: La implementación de sombras, bordes redondeados, entre otros.

Muchas de las cosas que hoy en día hacemos, como la inserción de imágenes, en el futuro podrán realizarse con códigos. Esto no solo se traduce en una mejora de la velocidad y performance de un sitio; sino también, en nuevas e ilimitadas opciones de diseño.

Una diferencia fundamental entre las aplicaciones de escritorio y web era la necesidad de procesar información y generar consultas en bases de datos. Se puede almacenar y procesar información en el cliente, convirtiendo a una aplicación web en una mucho más parecida a una de escritorio para generar los volcados de pila hacia un sistema persistente como una base de datos.

Dentro de las herramientas de desarrollo que se pueden utilizar para HTML5, están los entornos de desarrollo integrado, tales como: Notepad++, DreamWaver, Komodo 7, Aptana Studio, entre muchos otros.

2.11. HTML5 vs Flash

Una de las comparaciones necesarias dentro del marco de trabajo de las aplicaciones web, se enmarca entre estas dos posibilidades de manejo de multimedia, conllevando a análisis, como los siguientes:

Html5	Flash
Es un lenguaje para crear páginas web estándar creado por la W3C.	Desarrollado por Macromedia y ahora en continuidad con Adobe, el cual permite generar contenido animado entre otros. Requiere de una conexión rápida para ejecutarse adecuadamente.
Potencia	
En cuanto a la potencia actual para la animación 2D, se pueden hacer las mismas cosas con HTML5 y Flash.	La animación 3D es superior en el Flash, que el HTML5.
SEO	
Los sitios desarrollados en HTML5 tienen buen posicionamiento en buscadores, ya que es asimilado completamente por los buscadores.	Los sitios desarrollados en Flash tienen limitaciones significativas para posicionarse en motores de búsqueda, ya que el buscador no es capaz de leer su contenido.
Móviles	
En HTML5, se ve en los dispositivos móviles, siempre y cuando se cuente con un navegador actualizado.	Flash no se ve en algunos dispositivos móviles, excepto que se tenga montado el respectivo plugin.
Documentación	
La documentación para HTML5 y Canvas no es tan extensa.	La documentación de Actionscript 3 es extensa.
Seguridad	
La seguridad de HTML5 es nativa, y tiene definido las buenas prácticas, junto con el mejoramiento de la seguridad frente a bugs y vulnerabilidades que sus predecesores tenían.	Existen problemas graves de seguridad a tal punto que muchos de los navegadores deciden no permitir la instalación nativa del plugin dentro del mismo.

Tabla 4: HTML5 vs Flash.

Para crear un sitio web que tiene elementos de animación importantes, es recomendable utilizar HTML5, ya que hacerlo en Flash le dará una vida útil menor. Para los sitios web que tienen utilidades Flash, es recomendable hacer un plan de migración de las aplicaciones a HTML5, pues ya muchos de los navegadores no soportan dicho lenguaje y, por ende, la migración se hace necesaria.

2.12. IDE para desarrollo

Los Entornos de Desarrollo Integrado (IDE) para HTML5, son herramientas que permiten la edición de páginas web y la verificación de la sintaxis, como, por ejemplo: Notepad++, DreamWaver, Komodo 7, MS VS MVC3, Kompozer, NVU, Aptana studio 3, Web Expression Microsoft y Netbeans.

Los formularios son interfaces construidas con objetos que permiten integrar etiquetas, cajas de texto, botones entre otros; con el objeto de poder capturar información de un usuario y poder realizar algún tipo de procesamiento.

Los IDE para desarrollo de páginas web están integrados por el editor de texto que permite la edición (adicionar, actualizar, borrar sentencias de la página web) de código de los archivos HTML5, CSS3 y Javascript. Algunos IDE presentan los errores de sintaxis del código en la misma interfaz del editor, e incluyen o permiten que se les anexen plugins los cuales posibilitan la mejora o ejecución de ciertas tareas específicas.

Entre los entornos de desarrollo integrado están:
Notepad++: Es un programa para editar código fuente de cualquier lenguaje de programación. Este mismo tiene soporte para una gran cantidad de lenguajes, siendo de gran interés no sólo para los desarrolladores de webs, sino en general (Comunidad de Programadores). El sitio de descarga es: http://notepad-plus-plus.org/

Características que ofrece Notepad++

- Coloreado de código para una serie de lenguajes de programación diferentes, entre los que se incluyen todos los que un desarrollador web podría necesitar, como: HTML,

Javascript, ASP, SQL, PHP, CSS, Python, Ruby, entre otros.
- Genera la impresión a color de los códigos.
- Autocompletado de código, también configurable por el usuario o extensible por medio de descargas bajo demanda del programador.
- Permite realizar las acciones de Buscar / Reemplazar.

DreamWaver: Es el software más usado en el sector de diseño y programación web. Esta aplicación permite crear sitios de forma totalmente gráfica, y dispone de funciones para acceder al código HTML generado. Además, permite la conexión a un servidor, a base de datos, soporte para programación en ASP, PHP, Javascript, cliente FTP integrado, etc.

Komodo 7: Está en capacidad de operar con varios lenguajes de programación, ofreciendo a los programadores web la posibilidad de operar con HTML, Ajax, CSS, Javascript, XML, Perl, PHP, Python, Ruby y todos aquellos relacionados con el desarrollo de proyectos web. El sitio oficial de descarga es http://komodo-edit.uptodown.com/

Características:
- Permite trabajar por pestañas e incluye las herramientas de edición habituales para el trabajo con texto, como son: cortar, copiar, pegar, las de búsqueda y reemplazo de caracteres.
- Komodo Edit tiene un validador de sintaxis, el cual le permite verificar las aseveraciones propuestas en el código, e incluye la función de autocompletado agilizando el proceso de digitación de un programa al poder insertar directamente las expresiones más repetidas.

MS VS MVC3

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE) para el sistema operativo Windows, que integra un conjunto de herramientas de desarrollo para la generación de aplicaciones web ASP.NET, Servicios Web XML, aplicaciones de escritorio y aplicaciones móviles. Este software tiene soporte para Visual Basic, Visual C++, Visual C#, entre otros lenguajes, los cuales hacen uso de las funciones de .NET Framework, que ofrece acceso a diferentes

paquetes y tecnologías, fundamentales para agilizar el desarrollo de aplicaciones Web ASP y Servicios Web XML y demás.

Kompozer: Es una aplicación de software libre y de código abierto. El módulo de autoría web de "Mozilla Application" es utilizado para crear y editar páginas web, e-mail, y documentos de texto con mayor facilidad. Es compatible con los sistemas operativos Windows, Mac OS X y Linux. La aplicación de Kompozer permite a los programadores ver, escribir y editar código HTML.

NVU: Es una aplicación de software que sirve de editor de HTML, donde el desarrollador puede editar y verificar el resultado tal y como se mostrará en el navegador.

Aptana Studio 3: Es un entorno de desarrollo integrado (IDE) del mundo de código abierto para el desarrollo de aplicaciones web, desarrollado por Aptana, Inc. Es multiplataforma, es decir, que puede funcionar bajo Windows, Mac y Linux, con soporte para las especificaciones de última tecnología de los exploradores, como: HTML5, CSS3, JavaScript, Ruby, Rails, PHP y Python.

Web Expression Microsoft: Es una aplicación de software que permite la edición de HTML para el diseño y construcción de páginas web.

Características:

- Expression Web es compatible con hojas de estilo CSS y con Microsoft Visual Studio. Asimismo, incluye la capacidad de procesar archivos XML mediante Javascript.
- La versión Expression Web tiene soporte completo para PHP y Silverlight.

Netbeans: Es un entorno de desarrollo Integrado. Una herramienta que permite escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para cualquier otro lenguaje de programación como: C, C++, PHP, HTML y Ruby. Es una aplicación libre y gratuita sin restricciones de uso.

2.13. Formularios

Es un sistema que permite ofrecer al usuario final una interfaz para registrar datos, los cuales son utilizados para conocer las opiniones, dudas o simplemente capturar información y los pedidos a través de la red, entre otros.

Los formularios son instrumentos en HTML5 para enviar información desde una página web a algún programa u otro recurso en un computador remoto, haciendo uso de las bondades de la tecnología cliente - servidor.

Un formulario puede estar integrado por etiquetas, campos de texto, menús desplegables y botones, etc.

La etiqueta <form> actúa como una especie de contenedor para almacenar elementos que permiten al usuario seleccionar o introducir datos. Todos los datos se enviarán a la dirección "url" indicada en el atributo "action de la etiqueta form", por medio del método indicado en el atributo method.

La figura *-3-* muestra detalladamente un mapa mental de este concepto.

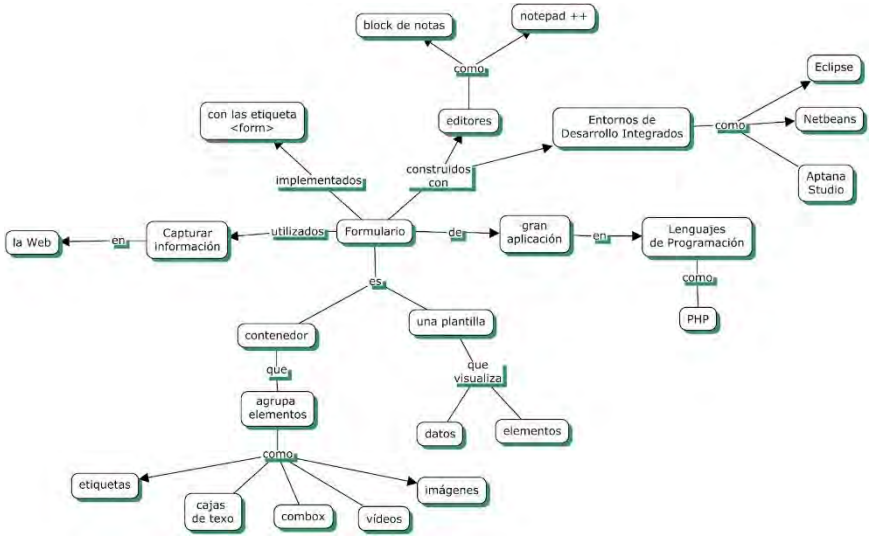


Ilustración 3: Mapa mental sobre formulario.

2.13.1. Etiquetas de formulario (Form tags)

<datalist>-Etiqueta que especifica una lista de opciones predefinidas para los controles de entrada. Se usa junto a <input>.

Ejemplo: Lista de países.

```
<label>Selecciona tu país:</label>
<input type="text" name="pais" id="pais" list="paises"/>
<datalist id="paises">
      <option value="Colombia" />
      <option value="Venezuela " />
      <option value="Ecuador" />
      <option value="Perú" />
      <option value="Bolivia" />
      <option value="Otro pais" />
</datalist>
```

<keygen>-Especifica un campo generador para claves con un formulario. Cuando se envía el formulario, la clave privada se almacena localmente y la clave pública se envía al servidor.

```
<form method="get">
      <label>Usuario:</label>
      <input type="text" name="usuario">
      <label>Clave encriptada:</label>
      <keygen name="clavepub" challenge="serie" />
      <input type="submit" name="security" value="Enviar" />
</form>
```

<output>-Representa el resultado de un cálculo o la salida de algún tipo de procesamiento, al igual que el realizado por una función.

<input>-Se utiliza para permitir al usuario introducir datos.

Tipos de entrada de atributos:

- **<input type="tel"/>-**Se utiliza para los números de teléfono. Por ejemplo <input type="tel" name="mitelefono">.

- **<input type="search" />-**Incluye un campo de texto utilizado para la búsqueda.
  Por ejemplo <input type= "search" name="buscador">.

- **<input type="url" />-**Indica un campo de texto para las direcciones URL.
  Por ejemplo <input type= "url" name= "url">.

- **<input type="email"/>-**Se utiliza para introducir direcciones de correo electrónico.
  Por ejemplo <input type="email" name= "email">.

- **<input type="date"/>-**Define un campo de fecha con selector.

- **<input type="month"/>-**Define un campo de fecha con el selector de fechas por meses.

- **<input type="week"/>-**Define un campo de fecha con el selector de fechas para la semana.

- **<input type="time"/>-**Define un campo de fecha con la hora, minutos, segundos y fracciones de segundos.

- **<input type="datetime"/>-**Define un campo con selector de fecha y hora.
  Por ejemplo <input type="datetime" name="fechayhora">.

- **<input type="datetime-local"/>-**Define un campo con la fecha y el selector de tiempo, en la zona horaria local.

- **<input type="number"/>-**Define un campo de número con controlador.
  Por ejemplo <input type=range name="numero" min=20, max=999>.

- **<input type="range"/>-**Un rango de números con un control deslizante.

Por ejemplo <input type=range name=”deslizador” min=1, max=100>

- **<input type=color/>-**Un selector de color.

Atributos

Placeholder: Visualiza información de ejemplo destinada a ayudar al usuario con la entrada de datos, sentencia utilizada con los elementos “input” y “textarea”. Información de apoyo al usuario que desaparece cuando se ingresan datos.

Required: Indica que ese campo no puede ser enviado vacío, y debe cumplir las especificaciones del tipo de campo que sea. Por ejemplo, en un campo tipo “email” debemos introducir una dirección de correo válida.

Autofocus: Hace que el cursor se coloque automáticamente en ese campo (caja de texto, área de texto) sin necesidad de ningún script.

Pattern: Expresión regular que debe coincidir con el valor correcto que queremos que introduzca el usuario.

Maxlength: En los textarea, se puede utilizar este atributo para limitar el número máximo de caracteres introducidos.

Ejemplos:
1) Formulario de contactos.

```
<!DOCTYPE html>
<html lang="es">
<head>
	<meta charset="utf-8">
	<title>HTML5 Contact Form</title>
</head>
<body>
	<form method="post" action="#">
		<label>Name</label>
		<input name="name" placeholder="Digite Nombre"> <br />
		<label>Email</label>
		<input name="email" type="email"
		placeholder="ejemplo@dominio.com"> <br />
		<label>Mensaje</label>
		<textarea name="message" placeholder="Digite
		 mensaje"></textarea> <br /><br />
		<button type="submit" name="enviar">Enviar </button>

	</form>
</body>
</html>
```

2) Página web que implementa un formulario de pago por la red.

```
<!DOCTYPE html>
<html lang="es">
	<head>
		<meta charset="utf-8">
		<title>HTML5 Contact Form</title>
		<link rel = "stylesheet" href = "css/estilos.css" />
	</head>
	<body>
		<form id="pago">
			<fieldset>
			<legend>Datos personales</legend>
				<ol>
					<li>
						<label for="nombre">Nombre</label>
						<input id="nombre" name="nombre"
						type=text placeholder="Escribe tu
					nombre
						completo" required autofocus>
					</li>
					<li>
						<label for="email">Email</label>
						<input id="email" name="email"
						type="email" placeholder =
						"datos@dominio.com" required>
					</li>
					<li>
						<label for="telefono">Telefono</label>
						<input id="telefono" name="telefono"
						type="tel" placeholder="Ej. 3102789520"
						required>
					</li>
					<li>
						<label for="pais">País</label>
						<input id="pais" name="pais"
type="text"
						required>
					</li>
				</ol>
			</fieldset>
			<fieldset>
				<legend>Datos de tarjeta de crédito</legend>
				<ol>
					<li>
						<label for="numtarjeta">Numero</label>
						<input id="numtarjeta"
name="numtarjeta"
						type="number" placeholder="Ej. 555-444-
						333-222" required>
```

```
					</li>
					<li>
						<label for="nomtarjeta">Nombre</label>
						<input id="nomtarjeta"
name="nomtarjeta"
						type="text" placeholder="Nombre exacto
					del propietario" required>
					</li>
				</ol>
			</fieldset>
				<fieldset>
				<button type="submit">Realizar compra</button>
			</fieldset>
		</form>
	</body>
</html>
```

Se recomienda digitar el anterior código de HTML5, grabarlo con el nombre de formularioPago.html y ejecutarlo en algún navegador,

salida.

2.13.2. Lecturas recomendadas

- Versiones de HTML.
- Visitar https://www.w3.org/ y revisar las temáticas de este sitio web.
- HTML API.
- HTML Geolocation.
- HTML Drag/Drop.
- HTML Web Storage.
- HTML Web Workers.
- HTML SSE.

2.13.3. Pregunta de revisión de conceptos

- ¿Qué es HTML?
- ¿Para qué sirve HTML?
- ¿Cuál es la estructura de HTML?
- ¿Qué es un estándar web?
- ¿Cuáles son los estándares relacionados con HTML?

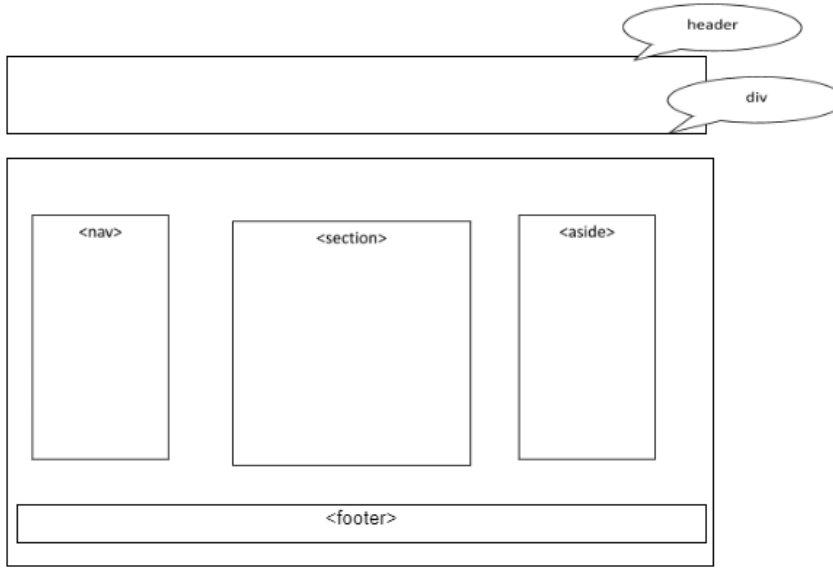
2.14. Ejercicios propuestos

1) Consultar y completar la información de la siguiente tabla:

IDE	Características	Sitio oficial de descarga
Notepad++		
DreamWaver		
Komodo 7		
Aptana Studio		
Kompozer		
Netbeans		

Tabla 5: Ejercicios IDE.

2) En grupo de dos o tres estudiantes realizar la maquetación de la página web con base al siguiente requerimiento:



3) Construir un formulario que permita ingresar la información de un usuario, conformada por: identidad, nombre, apellidos, dirección, teléfono, email, fecha de nacimiento, edad y género.

4) Marca utilizada para agrupar los elementos de introducción o de navegación dentro de una sección o documento. Por lo general, se usa para incluir los encabezados.

A. Hgroup
B. Aside
C. Section
D. Header

5) Etiqueta utilizada para marcar un contenido independiente, el cual tendría sentido fuera del contexto de la página actual, como, por ejemplo: una noticia, un artículo en blog o un comentario.

A. Article
B. Aside
C. Section
D. Header

6) Etiqueta utilizada para secciones que son a menudo representadas como barras laterales o como inserciones; y contienen una explicación al margen, como una definición de glosario, enlaces a páginas relacionadas con publicidad, la biografía del autor; o en aplicaciones web, la información de perfil o enlaces a blogs relacionados.

A. Article
B. Aside
C. Section
D. Header

7) La etiqueta que permite agrupar un conjunto de uno o más elementos de encabezado (<h1>–<h6>).

A. Hgroup
B. Aside
C. Footer
D. Header

8) Editor de texto y código fuente optimizado para el trabajo entorno a Windows; también con básicos que no incluyen ningún compilador utilizado por una amplia comunidad de desarrolladores.

A. Notepad++
B. DreamWaver.
C. Komodo 7
D. Kompozer.

9) Marca que se utiliza para indicar al usuario qué tipo de información ingresar.

A. Input
B. Meta
C. Placeholder
D. Form

10) Marca utilizada como plantilla para integrar elementos de entrada de información de los usuarios.

A. Input
B. Meta
C. Placeholder
D. Form

11) Etiqueta donde se le indica al navegador cuál es el tipo de codificado del documento HTML para el idioma y compatibilidad, entre otros.

A. Input
B. Meta
C. Placeholder
D. Form

Capitulo 3

Fundamentos de CSS3

3.1. Temática a desarrollar

- Estructuras de las sentencias de CSS.
- Selectores.
- Bordes Sombras.
- Transformaciones.

3.2. Introducción

En este capítulo, se abordarán los fundamentos de CSS (Cascading Style Sheets), los cuales son hojas de estilo en cascada, y tratan sobre un mecanismo utilizado para dar formato a un documento HTML5 o página web. Las hojas de estilo en cascada fueron creadas para separar el contenido del documento con la presentación de la página, pudiendo construir un lenguaje con su propia sintaxis para el manejo de los selectores, los comentarios y demás funcionalidades gráficas.

Las hojas de estilo se pueden incluir de manera interna en los archivos de las páginas web HTML5, o mediante archivos externos que tienen extensión CSS. CSS es una especificación desarrollada por el W3C (World Wide Web Consortium) para permitir la separación desde contenidos en los documentos escritos con HTML5, hasta la presentación del documento con las hojas de estilo (incluyendo elementos de presentación y diseño como los colores, los fondos, las márgenes, bordes, tipos de letra); lo cual permite mejorar la apariencia de una página web de una forma más sencilla.

W3C: Es una organización fundada y dirigida por Tim Berners-Lee (el creador de URL, HTTP, entre otros); los cuales son los elementos base

de la web actualmente, siendo la entidad que se encarga de definir los estándares adecuados para la red.

En este capítulo, se abordan conceptos de CSS3, los cuales permiten mejorar la presentación de los elementos de HTML5 en aspectos como: el poder aplicar bordes redondeados, añadir sombras a las cajas, utilizar una imagen como fondo, la rotación y transiciones.

CSS3 es uno de los avances de gran importancia en el diseño web, ya que tiene el control sobre todos los elementos del HTML, presentando interesantes efectos visuales y animaciones. La figura *-4-* detalla mediante un mapa mental, los fundamentos de CSS3. CITA

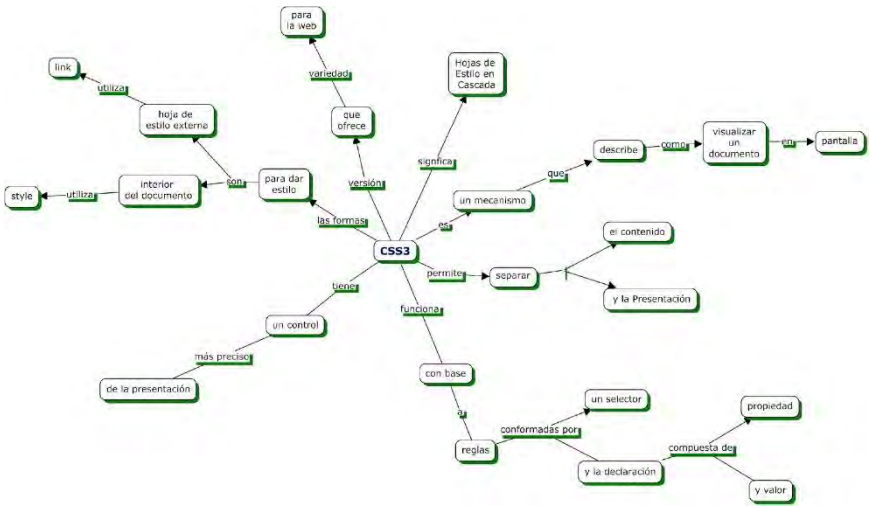


Ilustración 4: Fundamentos de CSS3.

3.3. Las hojas de estilo en cascada

CSS (Cascading Style Sheets), o también "hojas de estilo en cascada". La primera versión de CSS fue publicada a finales del año 1996. A su vez, fue logrando popularidad y aceptación hasta llegar a la versión 2.1, estándar actual que brinda compatibilidad con la mayoría de los navegadores existentes en el mercado.

A partir del año 2005, se comenzó a definir el sucesor de esta versión, al cual se le conoce como CSS3 o Cascading Style Sheets Level

3. Actualmente, en definición, esta versión brinda una gran variedad de opciones requeridas en el diseño web, como el sombreado y redondeado, hasta funciones avanzadas de movimiento y transformación.

CSS es un lenguaje de estilo que define la presentación de los documentos HTML5.Por ejemplo, CSS comprende aspectos relativos a fuentes, colores, márgenes, líneas, altura, anchura, imágenes de fondo, posicionamiento, animación, entre otros.

El lenguaje de las hojas de estilo está definido en las especificaciones del "World Wide Web Consortium" (W3C). Es, por tanto, un estándar internacional.

La asociación de estilos se realiza por medio de un archivo externo, lo cual genera que:

- El método más potente es una serie de especificaciones en un documento HTML5 en un archivo externo, el cual es uno de los más utilizados.
- Al tener un archivo de hojas de estilo construido, se puede aplicar a múltiples documentos HTML5.
- Pueden crearse distintos estilos (en otros tantos archivos CSS), y asociarlos con facilidad a varios documentos HTML5.
- Modificando el archivo CSS, se puede obtener un cambio radical en los documentos HTML5, sin necesidad de modificar las páginas web.
- Es una ventaja, ya que resulta mucho más organizado para el desarrollo web donde gestiona en forma separada el contenido y la apariencia.

3.4. Estructuras de las sentencias de estilo

Las sentencias de estilo se pueden vincular de varias maneras al código HTML5:

- Al digitar directamente en la etiqueta HTML5 (Online), aparece una serie de instrucciones en el área de cabecera del documento (embedded). La forma más sencilla de añadir estilos a las páginas web, es utilizando las etiquetas <style></style> en un documento HTML5 cualquiera.

- Otra manera es mediante la implementación de un archivo externo (extensión .css). Esto, por razones de claridad y cohesión, es recomendable separar los estilos de la página HTML5 en un archivo diferente; la hoja de estilo será importada desde la página principal.

3.4.1 Comentarios

CSS3 permite incluir comentarios entre sus reglas y estilos. Los comentarios son contenidos de texto que se incluyen en el archivo para su propia información y utilidad. Los navegadores ignoran por completo cualquier comentario de los archivos CSS3, por lo que es común utilizarlos para estructurar de forma clara los archivos CSS3 complejos.

El comienzo de un comentario se indica mediante los caracteres /*; y el final del comentario se indica mediante */.

Ejemplo: /* Presentación color del fondo y de la letra para los textos

```
h1 {
        background-color: #c0c0c0;
        color: #000033;
}
```

que están en la marca h1 */

3.5. Formas de escritura

Las hojas de estilo en cascada ofrecen propiedades para ampliar el lenguaje HTML5 en la representación visual de la página web. Las sentencias CSS3 utilizadas no son sensibles entre mayúsculas y minúsculas; igualmente, el uso que se recomienda para la codificación de hojas de estilo, es emplear por lo general minúsculas.

Para vincular hojas de estilo en archivos de HTML5, se puede realizar de las siguientes formas:

- Incluyendo las instrucciones de estilo en el encabezado de un documento HTML5.

Código html

```
<!DOCTYPE html>
        <head>
      <style>
            p {
                  color:blue;
                  text-align:center;
            }
      </style>
      </head>
      <html>
      <body>
            <h1>Un título </h1>
            <p>un párrafo</p>
      </body>
</html>
```

Salida

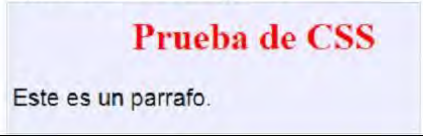
Está página genera la siguiente salida:

Codigo html

```
<!DOCTYPE html>
<html>
<head>
<style>
      body {
        background-color: lightblue;
      }

      h1 {
        color: red;
        text-align: center;
      }
      p {
        font-family: Arial;
        font-size: 20px;
      }
</style>
</head>
<body>
<h1>Prueba de CSS</h1>
<p>Este es un parrafo.</p>
</body>
</html>
```

Salida

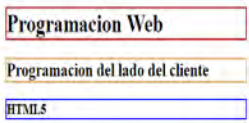
Al ejecutar el código html la salida es la siguiente:



Código Html

```
<!DOCTYPE html>
<html>
<head>
<style>
    h1 {
       border: 2px solid FireBrick;
    }
    h2{
       border: 2px solid BurlyWood;
    }
    h3{
      border: 2px solid Blue;
    }

</style>
</head>
<body>
<h1>Programacion Web</h1>
<h2>Programacion del lado del cliente</h2>
<h3>HTML5</h3>
</body>
</html>
```

Salida



Background: La propiedad "background" se utiliza para definir los valores individuales del fondo, en un lenguaje CSS. Se puede usar

Código Html

```
<!DOCTYPE html>
<html>
  <head>
  <style>
    h1 {
      background-color: green;
    }

    div {
      background-color: CornflowerBlue;
    }

    p {
      background-color: MediumAquaMarine;
    }
  </style>
  </head>
<body>
  <h1>Programacion Web</h1>
  <div>
    Programacion de lado del cliente.
  <p>HTML5 y CSS3</p>
    Pagina web
  </div>
</body>
</html>
```

Salida

background para definir los valores de una o de todas las propiedades, como: color, imagen, positivo, entre otras.

El archivo de una hoja de estilo externo tiene la extensión css, siendo un archivo anexo al código HTML5, lo cual permite separar la estructura de contenidos de la página con los estilos y presentaciones.

Para vincular una hoja de estilo en una página HTML5, se escribe el siguiente código:

```
<link rel="stylesheet" TYPE="text/css" HREF="estilos.css" />
```

- Para vincular una hoja de estilo a un documento, es necesario insertar la etiqueta <link> en él, entre las etiquetas <head> y </head>. Esta etiqueta no tiene etiqueta de cierre.
- La etiqueta "href" específica la hoja de estilo que se va a vincular al documento.
- Con el atributo "rel", se debe especificar la vinculación a una hoja de estilo, por lo que su valor ha de ser el nombre de la stylesheet.
- Por medio del atributo "type", se debe especificar que el archivo es de texto, con sintaxis CSS, por lo que su valor ha de ser text/css.

3.6 Los selectores

Permiten establecer la vinculación entre las instrucciones css y los elementos de la página a los que hay que dar formato. Se define entre las llaves que abre y la llave que finaliza { }. Los selectores permiten

```
selector {
        Propiedad1:valor1;
        Propiedad2:valor2;
        ...
        PropiedadN:valorN;
}
```

determinar a qué objeto o grupo de elementos se aplica un determinado estilo. La sintaxis básica de las reglas CSS es la siguiente:
Ejemplo de regla:

```
h1 {
        color: blue;
        font-family: garamond, serif;
        font-size: 12pt;
}
```

3.6.1. Selector universal

El selector universal representado por el carácter (*) es utilizado para seleccionar cada elemento del documento y vincularlo con una instrucción determinada.

Ejemplo:

El margen y el padding son 2 aspectos los cuales corresponden al espacio que hay entre el contenido y los bordes del contenedor.

```
* {
        margin: 0;
        padding: 0;
 }
```

Con (*) se da el valor cero a las distancias de todos los elementos predefinidos por el navegador.

Código html

```
<!DOCTYPE html>
<html>
  <head>
    <style>
    div {
      border: 1px solid black;
      margin-top: 100px;
      margin-bottom: 150px;
      margin-right: 150px;
      margin-left: 80px;
      background-color: LightSteelBlue;
    }
    </style>
  </head>
<body>
  <h2>Propiedades de los marges</h2>
  <div>Este elemento div tiene un margen superior de 100 px, un margen derecho de 150 px, un margen inferior de 150 px y un margen izquierdo de 80 px. </div>
</body>
</html>
```

Salida

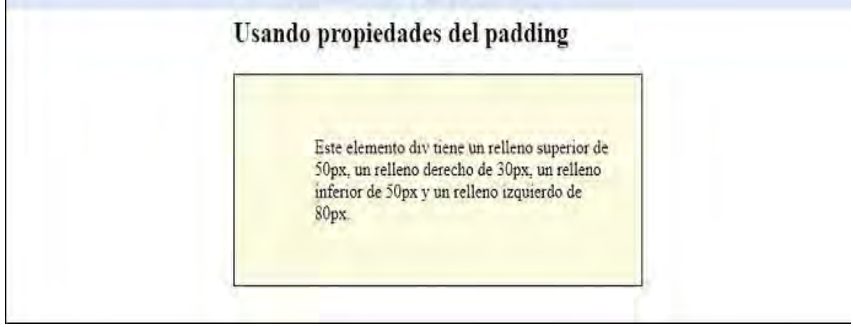
Propiedades de las márgenes

Este elemento div tiene un margen superior de 100 px, un margen derecho de 150 px, un margen inferior de 150 px y un margen izquierdo de 80 px.

Código html

```
<!DOCTYPE html>
<html>
<head>
      <style>
      div {
        border: 1px solid black;
        background-color: Beige;
        padding-top: 50px;
        padding-right: 30px;
        padding-bottom: 50px;
        padding-left: 80px;
      }
      </style>
      </head>
<body>
      <h2>Usando propiedades del padding</h2>
      <div>
        Este elemento div tiene un relleno superior de 50px, un relleno derech
      de 30px, un relleno inferior de 50px y un relleno izquierdo de 80px.
      </div>
</body>
</html>
```

Salida



3.6.2. Selector de elemento o de tipo

Para asignar a varios elementos las mismas propiedades, se puede formar un grupo con los selectores de elemento correspondientes:

```
h1, h2, h3 {
        color: #000333;
 }
```

Se selecciona todos los párrafos de la página:

```
p {
          sentencias
 }
```

El selector de tipo o etiqueta permite seleccionar todos los elementos de la página cuya etiqueta HTML5 coincida con el valor del selector.

Ejemplo: Página web HTML5 con los elementos article, headers y footers; e implementación de hojas de estilo, aplicando el selector universal y selector de tipo.

```
<!DOCTYPE html>
<html lang="es">
<head>
        <!-- Código de ejemplo HTML5 del curso de programación del lado del
cliente -->
    <meta charset="utf-8" />
    <title>Article, con Headers y Footers</title>
    <link rel = "stylesheet" href = "css/estilos.css" />
</head>
<body>

<header>
  <h1>Titulo dentro del header de la Página</h1>
</header>

<article>
    <header>
    <h2>Titulo dentro del header del artículo</h2>
    </header>

    <p>Contenidos del articulo principal de esta página</p>

    <footer>Footer clase fundamentos de programación web</footer>
</article>

<footer> <small>http://www.universidad.edu</small> </footer>

</body>
</html>
```

```
@charset "utf-8";
/* CSS Document */

*{
  margin:0;
  padding:0;
  border:0;
  outline:0;
  font-size:100%;
  font:inherit;
  vertical-align:baseline;
}
/* elementos de HTML5 */
article,footer,header{
  display:block;
}
body{
  line-height:1;
}
/*-------------------------------------*\
  Elementos del body
\*-------------------------------------*/
h1{
        font-size:3em;
```

```
}
h2{
	font-size:2em;
}
header, footer,article,p{
	padding:4em;
	}
header{
	background-color:#FFE4C4;
	}
footer{
	background-color:#ADD8E6;
	}
article{
	background-color:#FAF0E6;
	border:8px solid #E6E6FA;
}
```

3.6.3. Selector de clase

El selector de clase permite brindar un mayor control y precisión sobre la apariencia de los elementos HTML.

El nombre de una clase está conformado por letras, es decir, no puede empezar con números, caracteres especiales o espacios en blanco. Una regla importante para que el navegador no confunda este selector con los otros tipos de selectores, se prefija el valor del atributo class con un punto (.)

Ejemplo:
1) Creaciòn de selector de clase:

El código CSS es:

```
.importante{color:red;}
```

El código HTML es:

```
<body>
  <p class="importante">HTML5</p>
  <p> css3 </p>
</body>
}
```

2) Página web con los elementos de un formulario en HTML, e implementación de hojas de estilo aplicando el selector de clase.

Código archivo index.html:

```
<!DOCTYPE html>
<html lang="es">
<head>
      <meta charset="utf-8">
      <title>HTML5 Contact Form</title>
      <link  rel = "stylesheet" href = "css/estilos.css" />
</head>
<body>
<form class="contact_form" action="#" method="post">
    <ul>
        <li>
             <h2>Contáctos</h2>
        </li>
        <li>
            <label for="name">Nombre:</label>
            <input type="text"  placeholder="Miguel Hernandez" required />
        </li>
        <li>
            <label for="email">Email:</label>
            <input type="email" name="email"
            placeholder="mhernandez@ejemplo.com.co" required />
        </li>
        <li>
            <label for="website">Sitio Web:</label>
            <input type="url" name="web"
placeholder="http://www.universidad.edu.co"
            required />
        </li>
        <li>
            <label for="Mensaje">Mensaje:</label>
            <textarea name="Mensaje" cols="40" rows="6" required ></textarea>
        </li>
        <li>
             <button class="submit" type="submit">Enviar</button>
        </li>
    </ul>
</form>
</body>
</html>
```

Código archivo estilos CSS:

```
.contact_form h2 {
    background: none repeat scroll 0 0 #F3F3F3;
    border-radius: 5px;
    color: #5CD053;
    display: block;
    font-family: sans-serif;
    font-size: 20px;
    padding: 5px;
    text-shadow: 1px 1px 1px #CCCCCC;
    width: 433px;
}
/*--- estilos para los ul y li del formulario ---*/
.contact_form ul {
    width:750px;
    list-style-type:none;
      margin:0px;
      padding:0px;
}
.contact_form li{
      padding:12px;
      border-bottom:1px solid #eee;
      position:relative;
}
.contact_form label {
    color: #555555;
    display: inline-block;
    float: left;
    font-family: sans-serif;
    font-size: 13px;
    font-weight: bold;
    margin-top: 3px;
    padding: 3px;
    width: 90px;
}
.contact_form input {
      height:20px;
      width:220px;
      padding:5px 8px;
}
.contact_form textarea {
      padding:8px;
      width:300px;
}
.contact_form button {
      margin-left:90px;
}
```

3.6.4. Selectores ID

En ocasiones, es necesario aplicar estilos CSS a un único elemento de la página. Aunque, puede utilizarse un selector de clase para aplicar estilos a un único elemento, existe otro selector más eficiente en este caso.

El selector de ID permite escoger un elemento de la página a través del valor de su atributo ID. Este tipo de selectores sólo seleccionan un elemento de la página porque el valor del atributo ID no se puede

repetir en dos elementos diferentes de una misma página.

La sintaxis de los selectores de ID es muy parecida a los selectores de clase, salvo que se utiliza el símbolo de la almohadilla (#) en vez del punto (.) como prefijo del nombre de la regla CSS:

```
#destacado { color: red; }
```

Ejemplos: En el ejemplo, el selector "#" destacado, solamente selecciona el segundo párrafo (cuyo atributo ID es igual a destacado).

```
<p>Primer párrafo</p>
<p id="destacado">Segundo párrafo</p>
<p>Tercer párrafo</p>
```

1) Página web e implementación de hojas de estilos aplicando el selector ID. Archivo index.html:

```
<!DOCTYPE html>
<!-- FUNDAMENTOS DE CSS Selectores -->
<html>
	<head>
	<meta charset="utf-8">
	<title>Ejemplo Selector ID</title>
	<link  rel = "stylesheet" href = "css/estilos.css" />


	</head>
	<body>
		<header>
		<h3> Ejemplo Selector ID</h3>
		</header>

		<div class="columna">
		    <p id="primero">Parrafo uno.</p>
		    <p id="segundo">Parrafo dos.</p>
		</div>
	</body>
</html>
```

Archivo estilos .css:

```
#primero {
  color: blue;
}

#segundo {
  color: red;
}
```

3.6.5. Selectores de atributo

En CSS2, se tienen cuatro selectores de atributo por presencia o por valor [atributo]. Después, se selecciona un elemento con el atributo "atributo", cualquiera que sea su valor [atributo=valor]. Luego, se selecciona un elemento con el atributo "atributo", cuando su valor es exactamente igual a "valor".

[atributo =valor]: Se selecciona un elemento con el atributo "atributo", cuando su valor es una lista de valores separados por espacios, uno de los cuales es exactamente igual a "valor".

[atributo|=val]: Se selecciona un elemento con el atributo "atributo", cuando su valor es exactamente igual a "valor", o comienza por "valor", seguido de un guión "-".

En CSS3, se incorporan tres selectores que permiten seleccionar subcadenas dentro del valor del atributo:

Elemento [atributo =^ "valor"]: Se selecciona un elemento con el atributo, cuando su valor comienza con el prefijo de la cadena "valor".

Elemento [atributo $= "valor"]: Se selecciona un elemento con el atributo, cuando su valor del texto termina con el sufijo "valor".

Elemento [atributo *= "valor"]: Se selecciona un elemento con el atributo, cuando su valor contiene alguna instancia de "valor".

Ejemplos:

```
/* Se seleccionan todos los enlaces o referencias que lleven a una página web
que contenga la palabra java */
a [href *= "java"] { }
```

```
/* Se seleccionan todos los enlaces o referencias que incluyan una dirección
de correo */
a [href *= "mailto:"] { }
```

3.6.6. Gestión de las propiedades en CSS3

La figura *-4-* ilustra, mediante un mapa mental, el concepto de gestión de propiedades en las hojas de estilo como: transiciones, fuentes, fondos, bordes y sombras.

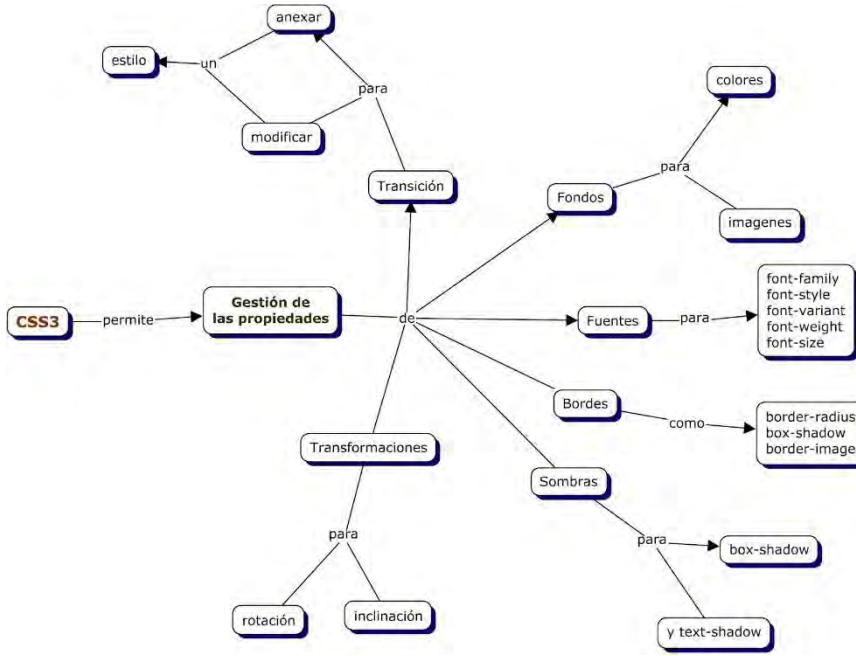


Ilustración 5: Gestión de las propiedades en CSS3.

3.7. Bordes

La propiedad que permite colocar las esquinas redondeadas en CSS3, es border-radius. Esta característica tiene la sintaxis "border-X-radius", donde X puede ser top-right, top-left, bottom-right y bottom-left. Es decir, se declara cada esquina por separado.

Hay tres tipos de bordes que son:

- Border- radius
- Box- shadow
- Border- image

3.7.1. Border- radius

Permite definir bordes redondeados en las esquinas, especificando las medidas del radio que deben darse a la curva de las esquinas; por ejemplo, border-radius: 5px 10px.

El valor de la x (5px 10px) puede estar compuesto por dos longitudes o porcentajes. Estas dos medidas especifican los radios de una elipse, la primera el radio horizontal y la segunda el vertical, de tal forma que el borde se dibuja con base a esa elipse.

Ilustración 6:Redondeo con elipse.

Para este caso "border-radius: 5px" con único valor, se entiende que el segundo es igual y, por tanto, la elipse se convierte en un círculo.

Los navegadores basados en WebKit, como Google Chrome o Safari; también soportan las esquinas redondeadas de CSS3, pero el atributo "border-radius" no funciona directamente, como en el caso de Firefox, sino que hay que utilizar un "alias": -webkit- border-radius.

Ejemplo: Página que permite la implemenación de bordes.

```
<!DOCTYPE html>
  <html>
     <head>
      <title>HTML5 Contact Form</title>
      <link rel = "stylesheet" href = "css/estilos.css" />
     </head>

     <body>
     <h1> HTML5 </h1>
     <p class="uno">Ejemplo de bordes CSS3</p>
</html>
```

Archivo **estilos.css**, el cual implementa bordes redondeados y bordes cuadrados.

```
h1{
          border:2px solid #a1a1a1;
          padding: 10px 40px;
          background:#dddddd;
          width:300px;
          border-radius:25px;
}
p.uno {
       border-style:solid;
       padding: 10px 40px;
       background:#dddddd;
       width:300px;
       border-width:5px;
}
```

3.7.2. Box-shadows

Esta sentencia realiza sombra a las cajas. El atributo box-shadow requiere varios valores para especificar las características de la sombra, como decolorado, separación de la sombra y la propia caja o color.

Ejemplo: Página que permite la implementación de cajas con sombra.

```
<!DOCTYPE html>
<html>
      <head>
        <title>HTML5 Contact Form</title>
        <link  rel = "stylesheet" href = "css/estilos.css" />
      </head>

      <body>
        <div class="Cuadrado">
      </div>
</html>


.Cuadrado {
      border-radius: 25px;
      padding: 10px 40px;
      width:  300px;
      height: 200px;
      background: #5e8b77;
      // tamaño y color de la sombra
      box-shadow: 10px 10px 5px #888888;
}
```

3.7.3. Border-image

Esta propiedad agrega un borde gráfico con el soporte de una imagen a cualquier tipo de elemento. Esta característica permite recortar de esa imagen la parte que se necesite, visualizándola a modo de background, mediante diversos tipos de repetición.

3.8. Sombras

CSS3 brinda la capacidad de crear y configurar el efecto de sombra con código CSS, lo cual permite elegir el color, tamaño, desenfoque y desplazamiento de la sombra (drop shadow).

3.8.1. Propiedad text-shadow y box-shadow

Para generar un código compatible con los principales navegadores, se debe tener en cuenta las siguientes reglas:

- moz- /* Firefox para su versión 5 en adelante */
- ms- /* para IE en su versión 9 en adelante */

- -o- /* Opera */
- -webkit- /* tanto para Chrome como para Safari */

Para implementar estos atributos mencionados en la propiedad text-shadow y la propiedad box-shadow, sería lo siguiente:

- moz-text-shadow:
- ms-text-shadow:
- -o-text-shadow:
- -webkit-text-shadow:

Ejemplo: Página web que implementa sombras para las marcas de h1, mediante el archivo de hojas de estilo externo.

Código HTML5:

```
<!DOCTYPE html>
   <html lang="es">
        <head>
          <meta charset="utf-8" />
          <link rel="stylesheet" type="text/css" href="estilos.css">
          <title>Ejemplo sombras</title>
        </head>
        <body>
          <header>
                  <hgroup>
                      <h1>Empresa abc</h1>
                      <h2>Informacion general</h2>
                  </hgroup>

           </header>
        </body>
</html>
```

Código de CSS:

```
h1{
  color:#7788cc;
  text-shadow:5px 0px 10px #f90;
  -moz-shadow:5px 0px 10px #f90;
  -ms-shadow:5px 0px 10px #f90;
  -o-shadow:5px 0px 10px #f90F;
  -webkit-shadow:5px 0px 10px #f90;
}


h2 {
     color:blue;
     text-shadow: 0 4px 0 #ccc;
    -moz-shadow:5px 0px 10px #f90;
    -ms-shadow:5px 0px 10px #f90;
    -o-shadow:5px 0px 10px #f90F;
    -webkit-shadow:5px 0px 10px #f90;
```

3.9. Transformaciones

Esta propiedad permite realizar transformaciones 3D y 2D a los elementos de HTML5, debido a la posibilidad de rotar y mover, sin necesidad de utilizar programas de edición de imágenes. La sintaxis es transform: función (parámetros); y en los navegadores, se aplica de la siguiente manera:

- moz-transform: función(parámetros); // en Mozilla
- webkit-transform: función(parámetros); // en Safari y Chrome
- -o-transform: función (parámetros); // en Opera

-Transform: función (cantidad): La función o tipo puede tomar los valores (rotate, skew, scale, translate), y cada uno de ellos realiza una función diferente.

-Rotate: Propiedad que permite girar los elementos un número de

grados. La sintaxis es transform: rotate(valor en radianes).

Ejemplo:

```
transform: rotate(19deg);
```

-Skew: Permite inclinar un elemento, tanto para coordenadas X, como Y. El valor se expresa en grados y la sintaxis es transform: skew(gradosX, gradosY);

Ejemplo:

```
transform: skew(18deg, 8deg);
```

-Scale: Con esta función, se puede escalar el elemento de HTML5, tanto en X, como en Y. La sintaxis es transform: scale(escalaX, escalaY);

Ejemplo:

```
transform: scale (2.4, 0.8);
```

-Translate: Permite desplazar el elemento, tanto en X, como en Y. Su sintaxis es transform: translate(desplazamientoX, desplazamientoY);

Ejemplo:

```
transform:translate(14px, 23px);
```

En una aplicación web, se pueden utilizar diferentes transformaciones en un mismo elemento de HTML5, solo hay que ingresarlas de manera consecutiva de esta forma:

```
transform: scale(2.8) skew(12deg) translate(8px) rotate(16deg);
```

Ejemplo:

- **Translate3d:** Traslada (o desplaza) el elemento a una posición 3D definida, en los ejes x, y, z.
- **Scale3d:** Especifica una operación de escala 3D, en los ejes x, y, z.
- **RotateX:** Especifica una rotación hacia la derecha por el ángulo dado, alrededor del eje x.
- **RotateY:** Especifica una rotación hacia la derecha por el ángulo dado, alrededor del eje y.
- **RotateZ:** Especifica una rotación hacia la derecha por el ángulo dado, alrededor del eje z.

Archivo *estilos.css*

```
.cuadrado{
	display:block;
	margin-bottom:25px;
	width:105px;
	height:65px;
	background:#6495ED ;
	border:2px solid #800000;
	-webkit-transition: all 2s ease-in-out;
	-moz-transition: all 2s ease-in-out;
	-o-transition: all 1s ease-in-out;
	-ms-transition: all 1s ease-in-out;
	transition: all 2s ease-in-out;
}
.cuadrado_transladar3d:hover {
	transform: translate3d(75%, -25%, 0);
	-ms-transform: translate3d(75%, -25%, 0); /* IE 9 */
	-webkit-transform: translate3d(75%, -25%, 0); /* Safari y Chrome */
	-o-transform: translate3d(75%, -25%, 0); /* Opera */
	-moz-transform: translate3d(75%, -25%, 0); /* Firefox */
}
.cuadrado_escala3d:hover {
	transform: scale3d(0.5, 1.25,1);
	-ms-transform: scale3d(0.5, 1.25,1); /* IE 9 */
	-webkit-transform: scale3d(0.5, 1.25,1); /* Safari y Chrome */
	-o-transform: scale3d(0.5, 1.25,1); /* Opera */
	-moz-transform: scale3d(0.5, 1.25,1); /* Firefox */
}
.cuadrado_rotar_x:hover{
	transform: rotateX(105deg);
	-ms-transform: rotateX(105deg); /* IE 9 */
	-webkit-transform: rotateX(105deg); /* Safari y Chrome */
	-o-transform: rotateX(105deg); /* Opera */
	-moz-transform: rotateX(105deg); /* Firefox */
}
```

Página HTML5

```
<!DOCTYPE html>
<!-- FUNDAMENTOS DE PROGRAMACION WEB -->
<!-- MAQUETADO con translated-->
<html>
<head>
    <meta charset="utf-8" />
    <title>Maquetado de una pagina web en HTML5</title>
      <link  rel = "stylesheet" href = "css/estilos.css" />
</head>
<body>
<header>
<h3>Ejemplos de Translated </h3>
</header>
    <p>Trasladar 3d:</p>
    <div class="cuadrado cuadrado_transladar3d"></div>
      <p>Escalar 3d:</p>
    <div class="cuadrado cuadrado_escala3d"></div>
      <p>Desplazar en el eje x:</p>
    <div class="cuadrado cuadrado_rotar_y"></div>
      <p>Rotar en el eje y:</p>
    <div class="cuadrado cuadrado_rotar_z"></div>
      <p>Rotar en el eje z:</p>
    <div class="cuadrado cuadrado_rotar_x"></div>
    <footer>
        <h3>Rotaciones en 3D</h3>
    </footer>

</body>
</html>
```

3.10. Imágenes

Para la gestión de imágenes, se utilizan las propiedades width y height, las cuales permiten visualizar las imágenes con cualquier altura o anchura, independientemente de sus dimensiones reales.

```
#imagen {
    width: 150px;
    height: 280px;
}
```

```
<img id="imagen" src="imagen.png" />
```

Los Fondos en CSS3, son una propiedad que se puede realizar de las siguientes maneras:

- **Background-origin & background-clip:** Permite ubicar las posiciones del fondo con respecto a las cajas, con tres modos diferentes.
- **Background-size:** Propiedad que permite definir el tamaño de la imagen de fondo. En píxel o porcentaje.
- **Multiple backgrounds:** Propiedad para el manejo de varias imágenes de fondo en el mismo elemento, lo cual se hace separando con comas cada imagen.

3.11. Transiciones

Permiten cambiar una propiedad (o un grupo de ellas) en un período de tiempo determinado. Una ventaja sobre Javascript es que, si esta propiedad no es soportada por el browser, la animación simplemente no es mostrada.

Ejemplo 1: El siguiente código conformado por un archivo HTML y otro CSS; permite visualizar el aumento del ancho de un en un segundo, cuando el mouse se posiciona sobre él. Luego, cuando el mouse sale del , su ancho retorna a la posición inicial en tres segundos:

Archivo html

```
<!DOCTYPE html>

<html>
  <head>
      <title>HTML5 Transiciones </title>
      <link  rel = "stylesheet" href = "css/estilos.css" />
  </head>
     <body>
        <p class="prueba">Ejemplo de transiciones CSS3</p>
     </body>
</html>
```

Archivo estilos .css:

```
.prueba {
    margin:10px;
    width:200px;
    height:200px;
    background:#eeeddd;
    border:1px solid #abcdf0;
    -webkit-transition: all 1s ease; /* Safari-Chrome */
}
.prueba:hover{
      width:400px;
     -webkit-transition: all 3s ease; /* Safari-Chrome */
}
```

Ejemplo 2: Página web que integra el maquetado de una página HTML5, como el header, las marcas nav, section, aside; presentadas entre columnas con bordes redondeados y de diferentes colores.

Codigo CSS3:

```
*{
      margin: 0;
      padding: 0;
}

/* 16px = 12pt = 100% = 1em M*/

body {
      background:  #212121 url(../imagen/fondo.jpg);
      color:#FFF;
      font-family: Helvetica, Verdana;
      font-size: 2m;
}

h1 {
      color: #F60;
      margin:0.25em auto;
      text-align:center;
      text-shadow: 5px 5px 10px rgba(255,255,255,0.5);
}

h3 {
      color: #A60;
      margin:0.25em auto;
      text-align:center;
      text-shadow: 5px 5px 10px rgba(255,255,255,0.5);
}


header{
      background: #1E6381;
      border-radius: 0.5em;
      margin: 0.5em auto;
      max-width: 560px;
      padding:0.25em;
      text-align:center;
}

div
{
-moz-column-count:3; /* Firefox */
-webkit-column-count:3; /* Safari and Chrome */
column-count:3;
}

nav {
      background: #28108A;
      border-radius: 0.5em;
      margin: 0.5em auto;
      max-width: 560px;
      padding:6em;
```

```
      text-align:center;
      width:15%;
}



aside {
      background: #0D351F;
      border-radius: 0.5em;
      margin: 0.5em auto;
      max-width: 560px;
      padding:6em;
      text-align: center;
      width:15%;
}

section {
    background: #4E240D;
      border-radius: 0.5em;
      margin: 0.5em auto;
      max-width: 700px;
      padding: 6em;
      text-align: center;
      width:50%;
}

section#principal {
      background: #1E6381;
      border-radius: 0.5em;
      display:inline-block;
      margin: 0.25em auto;
      max-width: 560px;
      min-height: 100px;
      padding: 0.25em;
      text-align:center;
      vertical-align:top;
      width:65%;
}
```

En el body de la URL, se define con referencia a la hoja de estilo, no al documento vinculado a ella; por lo tanto, si quiere integrar una imagen de fondo en su página, en el archivo CSS aparecerá, por ejemplo:

```
body {
background-image: url (../imágenes/imagen.gif);
/* otras declaraciones */
}
```

Página HTML5

```
<!DOCTYPE html>
<!-- FUNDAMENETOS DE PROGRAMACION WEB -->
<!-- MAQUETADO PASO 3-->
<html>
<head>
    <meta charset="utf-8" />
    <title>Maquetado de una pagina web en HTML5</title>
      <link  rel = "stylesheet" href = "css/estilos.css" />
</head>
<body>
<header>
<h1>Etiqueta header </h1>
</header>
 <div id="MyMainContainer">
        <nav>
             <h3>Enlaces</h3>
        </nav>
        <section>
        <h1> section</h1>
        </section>
        <aside>
        <h1> aside</h1>
        </aside>
</div>
        <footer>
        <h1>Estas en la etiqueta header footer</h1>
        </footer>

</body>
</html>
```

3.12. Lecturas recomendadas

- @font-face
- Gradiente lineal
- Gradiente radial
- RGBA
- HSLA
- Outline
- Border-image
- Transform y transition

3.13. Preguntas de revisión de conceptos

- W3C (World Wide Web Consortium)
- ¿Qué son las hojas de estilo?

- Utilidad de las hojas de estilo en la programaciónweb
- Elementos de seguridad según la W3C

3.14. Ejercicios propuestos

1) Elemento de CSS3 que permite establecer a qué elemento o grupo de elementos se aplica un determinado estilo:

a) CSS

b) Selector

c) Propiedad

d) Comentarios

2) Elementos de CSS3, donde los contenidos de texto que se incluyen en el archivo son para su propia información y de utilidad para los programadores, pero son ignorados por los navegadores:

a) CSS

b) Selector

c) Propiedad

d) Comentarios

3) El selector representado por el carácter (*) es utilizado para seleccionar cada elemento de la página web y vincularlo con una instrucción determinada:

a) Selector de ID

b) Selector de Clase

c) Selector de Atributo

d) Selector Adyacente

4) Selector que aplica estilos CSS a un único elemento de la página:

a) Selector de ID

b) Selector de Clase

c) Selector de Atributo

d) Selector Adyacente

5) Sentencia que permite cambiar una o más propiedades en un período de tiempo determinado:

a) Sombras

b) Transformaciones.

c) Imágenes

d) Transiciones

6) Las propiedades width y height, permiten visualizar las imágenes con cualquier altura y/o anchura. Cuál de las siguientes marcas las implementa:

a) Sombras

b) Transformaciones.

c) Imágenes

d) Transiciones

7) Esta propiedad permite la ejecución en 3D y 2D a los elementos de HTML5, debido a la posibilidad de rotar o mover sin necesidad de utilizar programas de edición de imágenes:

a) Sombras

b) Transformaciones.

c) Imágenes

d) Transiciones

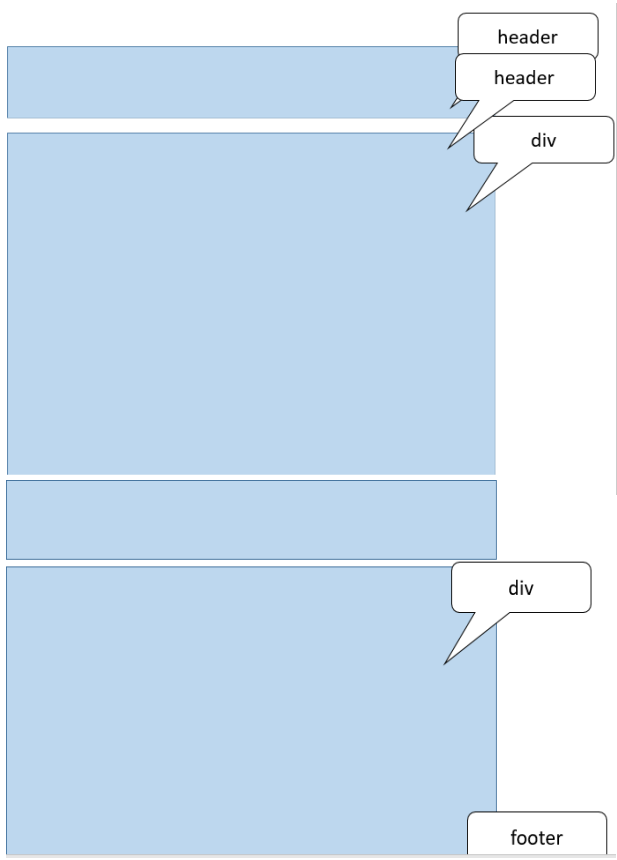
8) Completar la información de la siguiente tabla:

Tipo de Selector	Patrón	Descripción
Universal	*	
Tipo	E	
Clase	.info	
ID	#footer	
Descendiente	E F	
Hijo	E > F	
Adyacente	E + F	
Atributo	E[att]	
Atributo	E[att=val]	
Atributo	E[att~=val]	
Atributo	E[att|=val]	
La pseudo-clase :first-child	E:first-child	
Tipo de Selector	Patrón	Descripción
Las pseudo-clases link	E:link E:visited	
Las pseudo-clases dinámicas	E:active E:hover E:focus	
La pseudo-clase :language	E:lang(c)	
El pseudo-elemento :first-line	E:first-line	
El pseudo-elemento :first-letter	E:first-letter	
Los pseudo-elementos :before y :after	E:before E:after	

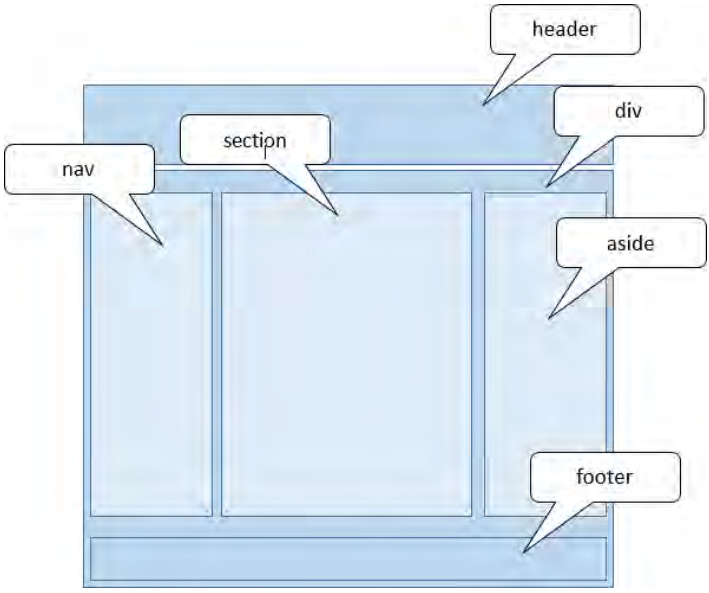
Tabla 6: Ejercicio tipo de selector.

9) Construir una página web para registrar reclamos de los clientes a partir de una identificación, teniendo en cuenta el nombre y la novedad, de manera que se pueda ejecutar en cualquier navegador, sin alterar la presentación establecida en el archivo de estilos.

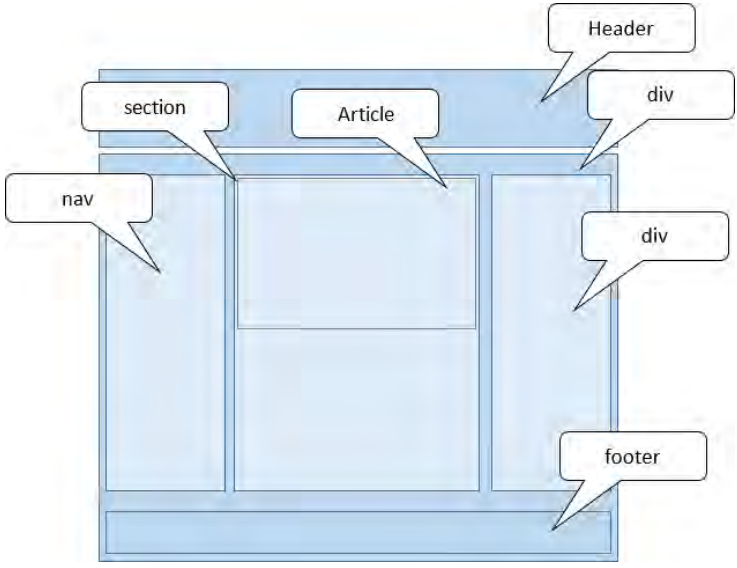
10) Desarrollar la página HTML y la CSS correspondiente a la siguiente salida.



11) Desarrollar la página HTML y la CSS correspondiente a la siguiente salida.



12) Desarrollar la página HTML y la CSS correspondiente a la siguiente salida.



Sitios web de interés

- http://www.w3.org/TR/2009/PR-css3-selectors-20091215/#selectors
- http://www.w3schools.com/css/default.asp
- http://www.w3.org/People/Bos/

Bibliografía

Capitulo 4

4. Fundamentos de JavaScript

4.1. Temática a desarrollar

- Variables
- Condicionales
- Ciclos
- Arreglos

4.2. Introducción

En la programación web, se utiliza JavaScript como un lenguaje de programación similar a otros de la especie, como PHP; claro está, con varias diferencias importantes. Este lenguaje se utiliza principalmente del lado del cliente, es decir, se ejecuta en nuestro computador, no en el servidor, permitiendo crear efectos atractivos y dinámicos en las páginas web.

En este capítulo se abordarán los siguientes temas:

- JavaScript como un lenguaje de programación orientado al desarrollo de software del lado del cliente, el cual se integra con HTML5.
- JavaScript requiere para el almacenamiento de datos en memoria, el manejo de variables.
- Para efectos de realizar comparaciones, evaluar datos, repetir sentencias; se hace uso de las estructuras de control.
- Las funciones son partes del código que realizan tareas específicas y pueden ser invocadas las veces que sea necesario, en lugar de digitar esas mismas sentencias en diferentes partes del programa.

JavaScript permite la construcción de programas conformados por funciones, dentro de las cuales se desarrolla la lógica de actividades específicas, utilizando las sentencias condicionales para evaluar o comparar expresiones y variables, por medio del uso de sentencias de control para la ejecución y verificación de condiciones, ya sean de una sola pasada o dentro de un determinado número de veces, mientras se cumple con la condición específica, según sea el caso.

Dado que JavaScript dispone de fuertes capacidades de programación orientada a objetos, en este capítulo también se abordará el fundamento de este paradigma de la programación como lenguaje interpretado basado en objetos, ya que no posee clases donde los scripts manipulen los objetos propios del lenguaje, como: Math, String, Date, Los Arrays, entre otros; y los que le proporciona el navegador. Otros elementos a tener en cuenta, son los eventos que suceden cuando el usuario pasa el mouse por una imagen, o al hacer clic en un botón, dichos eventos son los que sirven para que se ejecuten las instrucciones de JavaScript de una página o un programa.

Javascript no tiene todas las características de los lenguajes orientados a objetos como Java o C++, pero si está en capacidad de manejar objetos e inclusive crearlos.

Javascript tiene algunas clases predefinidas u objetos intrínsecos, como son: Array, Boolean, Date, Function, Global, Math, Number, Object, RegExp y String.

En ambientes de desarrollo, se pueden crear objetos nuevos con métodos y propiedades adaptados a las necesidades concretas de la aplicación a desarrollar. La figura -7- detalla, mediante un mapa mental, este concepto de JavaScript. La figura -7- amplía varios conceptos inherentes a este lenguaje de programación para la web.

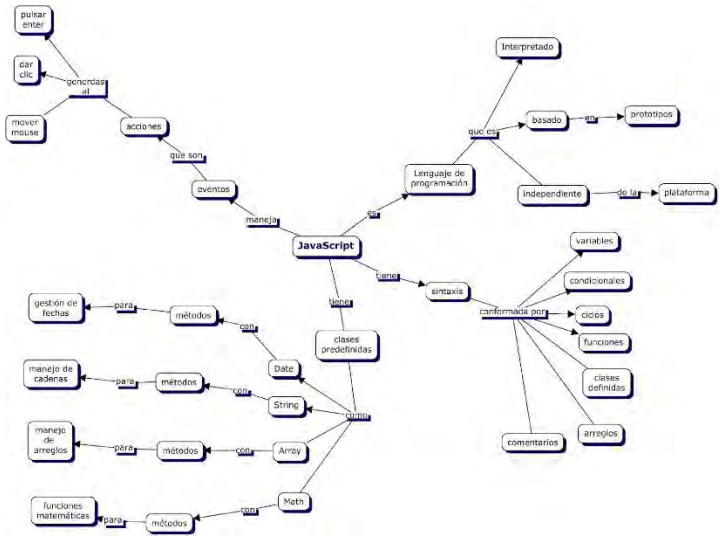


Ilustración 7: JavaScript.

4.3. Fundamentos de JavaScript

JavaScript es un lenguaje interpretado, basado en objetos que permite incluir macros en páginas web. Estas macros se ejecutan en el computador, en páginas web del cliente, y no en el servidor (algo muy interesante, porque los servidores web están sobrecargados con el procesamiento propio de la aplicación, mientras que los computadores de los usuarios no suelen estarlo, pues solo realizan las peticiones y esperan las respuestas del lado del servidor).

4.4. Ingreso de códio JavaScript en las páginas web

Para añadir un código JavaScript en una página HTML5, debe incluirse el código entre las etiquetas <head> y </head> de la página, que es la forma correcta de registrar la funcionalidad, aunque también puede ir en el cuerpo de la misma.

El tag para adelantar esta labor es:

```
<script>
 <!-- Programa JavaScript //--
>
```

4.5. Comentarios en JavaScript

Los comentarios se utilizan en los lenguajes de programación para realizar explicaciones del código, para que una o varias líneas no sean tenidas en cuenta cuando el intérprete de comandos las encuentre. En JavaScript los comentarios pueden ser:

- Comentario de una línea //
- Comentario de varias líneas, se usan los caracteres /* (inicio de comentario) y */ (fin de comentario):

Los bloques de código que integran una unidad, están encerrados entre las llaves (y) (por ejemplo, el código de una función, las sentencias incluidas dentro de un condicional o ciclo). Por otra parte, JavaScript hace diferencia entre mayúsculas y minúsculas para los nombres de variables y funciones.

4.6. Variables

Por la naturaleza de JavaScript, las variables permiten almacenar información en memoria y pueden pasar de contener un tipo de dato a otro tipo sin necesidad de estar definiendo el cambio o haciendo la conversión.

JavaScript cuenta con siete tipos de datos que son:

- **Números:** Corresponde a valores numéricos, con o sin fracción decimal, ya sean positivos o negativos.

Ejemplos:

```
<script>
      var dato = 40;
</script>
```

- **Cadenas de caracteres:** Se encuentran delimitados por comillas simples o dobles. Además, pueden incluir caracteres especiales como:
- \' (comilla)
- \" (comilla doble)

- \n Nueva línea
- \t Tabulador
- \r Retorno de carro
- \f Alimentación de formulario
- \b Retroceso de un espacio

Ejemplo:

```
<script>
        var cadena = "Hola Mundo";
</script>
```

Las cadenas de texto en JavaScript pueden ir entre comillas dobles o simples; lo único a tener en cuenta, es utilizar el mismo tipo de comillas en la apertura y cierre de la cadena de texto.

Adicionalmente a esto, se debe tener en cuenta el uso de las tildes y los caracteres especiales, pues por defecto, trabaja con el juego de caracteres en inglés.

- **Booleanos:** Permiten solo el manejo de dos valores: verdadero (true) y falso (false). Se digitan sin comillas.

Ejemplo:

```
<script>
    var semaforo = true;
</script>
```

- **Objetos:** Corresponden a una agrupación de variables y funciones definidas en JavaScript o por el usuario, como, por ejemplo: Image y Array, que son objetos predefinidos. Para definir un objeto, se hace uso del operador new y los paréntesis redondos.

Ejemplo:

```
<script>
      var obj = new Image();
</script>
```

- **Nulos:** Son datos vacíos que se generan cuando se ha definido una variable como null para borrarla.

Ejemplo:

```
<script>
    var valorNulo = null;
</script>
```

- **Indefinidas:** Son variables que aún no se les ha asignado algún valor.

Ejemplo:

```
<script>
    var x1, x2, x3;
</script>
```

- **No numérico:** Son valores generados cuando se realiza una operación determinada, porque los datos no son compatibles. Su valor es NaN (not a number):

Ejemplo:

```
<script>
    var noNumerico = 'x' * 2;
</script>
```

4.7. Entrada y salida de datos

Para la entrada de datos, se utiliza el método prompt () del objeto windows, conformado por los botones "Aceptar" y "Cancelar", junto a una caja de texto donde el usuario ingresa los datos.

Ejemplo:

```
<script>
    nombre = prompt('Digite nombre', '');
</script>
```

Para realizar la impresión o salida de datos, se utiliza el método write().

Ejemplo:

```
document.write("Texto") o bien  document.write('Texto')
```

4.8. Operadores

Los operadores permiten manejar el valor de las variables, realizar operaciones matemáticas con sus valores y comparar diferentes variables.

Se distinguen los siguientes tipos de operadores:

- **Operador de asignación**
  Este operador se utiliza para almacenar un valor específico en una variable. El símbolo utilizado es un igual (=).
- **Operadores relacionales**
  Los operadores relacionales definidos por JavaScript son similares a los que definen las matemáticas: mayor que (>), menor que (<), mayor o igual (>=), menor o igual (<=), igual que (==) y diferente (!=).

  No confundir con el operador de asignación (un igual =), el operador relacional (doble igual ==).
- **Operadores ariméticos**
  Los operadores aritméticos definidos por JavaScript son similares a los que definen las matemáticas: suma (+), resta (-), producto (*), división (/) y residuo o módulo de la división (%).
- **Operadores de incremento y decremento**
  Estos operadores actúan de la siguiente manera: solamente son válidos para las variables numéricas y se utilizan para incrementar doble más (++) o decrementar doble menos (–) en una unidad el valor de una variable.

Ejemplo:

```
var x = 5;
++x; // x = 4
var y = 9;
 --y ;  // y = 8
```

- **Operadores lógicos**
  Estos operadores actúan de la siguiente manera:

Operador	Descripción
&&	"Y "Devuelve true, si ambos operadores son true.
||	"O" Devuelve true, si uno de los operadores es true.
!	"No"Devuelve true, si la negación del operando es true.

Tabla 7: Operadores lógicos.

Ejemplo:

```
<!doctype-->
<html>
<head>
   <title>Ejemplo con operadores en JavaScript</title>
</head>
<body>
<script>
   a = 10;
   b = 5;
   c = 5;
   document.write( (a == b) && (c > b) );document.write("<br/>");
   document.write( (a == b) || (b == c) );document.write("<br/>");
   document.write( !(b <= c) );document.write("<br>");
</script>
</body>
</html>
```

4.9. Estructuras de control

Para efectos de controlar la ejecución de una acción o bloque de acciones (una o más veces), se hace uso de las estructuras de control, dentro de las que están:

4.9.1. Condicionales

El condicional if realiza un bloque de acciones si se cumple una condición y si no se realiza el bloque de acciones de la sentencia else.

4.9.2. Estructura de "if"

La estructura if (en JavaScript), se emplea para tomar decisiones en función de una condición. La sintaxis formal es:

```
if(condición) {
  ...
}
```

La condición es una expresión booleana (true y false), que si la condición es verdadera, se ejecutan todas las instrucciones que se encuentran dentro del if. Si la condición no se cumple (es decir, si su valor es false) no se ejecuta ninguna instrucción contenida en el if, y el programa continúa ejecutando el resto de las instrucciones del script.

Ejemplos:
1) Se evalúa si el valor de dos variables es igual:

```
var mostrarMensaje = true;
var x = 7;
var y = 7;
if(x == y) {
  alert("números iguales");
}
```

2) Página web, que haciendo uso de JavaScript, se evalúa si una persona es mayor de edad:

```
<!DOCTYPE html>
<!-- FUNDAMENTOS DE JavaScript -->
<html>
<head>
 <meta charset="utf-8" />
 <title>Documento sin estilos</title>
<h3> Introduccion javascript </h3>
</header>
<script language="JavaScript">
        var nombre;
        var edad;
        nombre=prompt('Ingrese nombre:','');
        edad=prompt('Ingrese la edad:','');
         if (edad>=18)
         {
           document.write(nombre+' eres mayor de edad '+edad);
         }

</script>
</head>
<body>
<header>
<h3> Introduccion javascript </h3>
</header>
</body>
</html>
```

3) Programa donde se involucre el manejo de variables:

```
<!DOCTYPE html>
<!—Fundamentos de JavaScript -->
<html>
       <head>
        <meta charset="utf-8" />
        <title>Documento sin estilos</title>
       <script language="JavaScript">
                var nombre="Pedro";
                var edad=19;
                var genero="M";
                var casado=false;
                document.write("Nombre: " ,nombre);
                document.write('<br />');
                document.write("Edad: ",edad);
                document.write('<br />');
                document.write("Genero: ",genero);
                document.write('<br />');
                document.write("Casado: ",casado);
       </script>

       </head>
       <body>
              <header>
              <h3> Introduccion javascript </h3>
              </header>
       </body>
</html>
```

4.9.3. Eestructura “if else”

Al evaluar la expresión “si la condición se cumple” (es decir, si su valor es verdadero), se ejecutan todas las instrucciones que se

encuentran dentro de las llaves del if. Si la condición no se cumple (es decir, si su valor es falso), se ejecutan todas las instrucciones contenidas en else . La sintaxis formal es:

```
if(condición) {
  ...
}
else {
  ...
}
```

Ejemplo: A partir del ejemplo anterior, se agrega la sentencia "else", y en caso de que la edad sea menor de 18, se ejecutarán las sentencias correspondientes.

```
<!DOCTYPE html>
<!-- Fundamentos de JavaScript -->
<html>
<head>
 <meta charset="utf-8" />
 <title>Documento sin estilos</title>
<h3> Introduccion javascript </h3>
</header>
<script language="JavaScript">
    var nombre;
    var edad;
    nombre=prompt('Ingrese nombre:','');
    edad=prompt('Ingrese la edad:','');
    if(edad >= 18) {
         alert(nombre+' eres mayor de edad, tienes '+edad);
    } else {
        alert(nombre+' eres menor de edad, tienes '+edad);
    }
</script>
</head>
<body>
<header>
<h3> Introduccion a javascript </h3>
</header>
</body>
</html>
```

4.9.4. Sentencia Switch

El condicional switch selecciona uno de muchos bloques de código para ejecutar, según una evaluación o un valor. La sintaxis formal es:

switch([evaluación | valor]) {
case 1:
ejecuta este bloque si el valor o la evaluación es igual a 1. break;
case 2:
ejecuta este bloque si el valor o la evaluación es igual a 2. break;
default:
ejecuta este bloque si no ingresa a ninguno de los çase.anteriores.
}

Ejemplo: Programa en JavaScript que a partir del número del día de la semana imprime el nombre.

```
<!DOCTYPE html>
<!—Fundamentos de JavaScript -->
<html>
<head>
 <meta charset="utf-8" />
 <title>Documento sin estilos</title>
<h3> Introduccion javascript </h3>
</header>
   <script language="JavaScript">
    var dia;
    dia=prompt('ingrese numero del dia de la semana:','');
     // estructura condicional de elección día
      dia= parseInt(dia);
     switch (dia)
     {
         case 0 : nomDia="domingo"; break;
         case 1 : nomDia="lunes"; break;
         case 2 : nomDia="martes"; break;
         case 3 : nomDia="miércoles"; break;
         case 4 : nomDia="jueves"; break;
         case 5 : nomDia="viernes"; break;
         case 6 : nomDia="sábado"; break;
         default : nomDia="un día extraño";
     }
     // emisión del mensaje de alerta
     alert("¡ Hoy es " + nomDia + " !");

</script>
</head>
<body>
<header>
<h3> Introduccion javascript </h3>
</header>
</body>
</html>
```

Se recomienda digitar el anterior código en una página web, ejecutarlo en un browser y observar el resultado a partir de los datos ingresados.

4.10. Ciclos o estructuras repetitivas

Son estructuras que permiten la ejecución de una o más sentencias en un determinado número de veces y tiene las siguientes características:

- La inicialización es el espacio donde se establecen los valores iniciales de las variables que controlan el ciclo.
- La condición es el único elemento que decide si continúa o se detiene el ciclo.
- La actualización puede ser el incremento o decremento de una variable que se asigna después de cada iteración a las variables que controlan el ciclo.

Las estructuras cíclicas en JavaScript son:

- While
- Do while
- For
- For in

4.10.1. La estructura while

La sintaxis general es while (condición), donde se ejecuta la expresión mientras la condición sea verdadera (while (true) {...}), y en caso contrario continúa con el resto del script.

Ejemplo: Programa que muestra los números del 1 hasta 10.

```
<!DOCTYPE html>
<!–Fundamentos de JavaScript -->
<html>
<head>
 <meta charset="utf-8" />
 <title>Documento sin estilos</title>
<script language="JavaScript";
        var x;
        x=1;
        while (x<=10)
        {
           document.write(x);
           document.write('<br>');
           x=x+1;
        }
```

```
  </script>
</head>
<body>
<header>
<h3> Introduccion a javascript </h3>
</header>
  </body>
</html>
```

4.10.2. La estructura do…while

El ciclo de tipo do...while es muy similar a la estructura while, con la diferencia que para este caso, se ejecutan las instrucciones del ciclo (al menos la primera vez). La sintaxis general es:

do {
...
} while(condición);

Ejemplo: Utilizando este bucle, calcular el factorial de un número:

```
var fact = 1;
var numero = 5;
 do {
  fact *= numero;  // fact =fact * numero
  numero- -;
} while(numero > 0);
document.write(fact);
```

4.10.3. La estructura for

Estructura repetitiva que tiene la siguiente forma:

for(inicialización; condición; actualización) {
...
}

La sentencia de inicialización se ejecuta una única vez al ingresar al ciclo; las iteraciones se realizan mientras la condición indicada sea

verdadera, repite las instrucciones definidas dentro del for. Además, en cada iteración se actualiza el valor de las variables que se utilizan en la condición de terminación.

Ejemplo:

```
for(var n = 0; n < 5; n++) {
        document.write ( n );
}
```

4.10.4. Estructura for...in

Una estructura de control derivada de for es la estructura for...in. Su definición exacta implica el uso de objetos, aplicada al uso de arrays. La sintaxis formal es:

```
for(indice in array) {
...
}
```

Si se quiere recorrer todos los elementos que forman un array, la estructura for...in es la forma más eficiente de hacerlo, como se muestra en los siguientes ejemplos:

```
var diasSemana = ["Lunes", "Martes", "Miércoles", "Jueves", "Viernes",
"Sábado", "Domingo"];
for(i in diasSemana) {
      document.write (diasSemana[i]);
}
```

```
var colores=["Gris", "Verde", "Azul", "Negro"];
for(i in colores){
      alert(colores[i]);
}
```

4.10.5. Funciones

Permiten agrupar sentencias una sola y única vez dentro de una función para realizar una tarea específica y que se pueda reutilizar fácilmente. Las funciones son la solución al proceso de repetir el mismo código en varias partes de un programa.

La estructura general de una función de JavaScript es:

```
function nombre_de_la_función(){
... enunciados a ejecutar...
}
```

Algunas funciones no necesitan ninguna información para generar resultados, sin embargo, una función puede recibir tantos parámetros como se requieran y para representarlo, se colocan los nombres de los parámetros separados por comas, dentro de los paréntesis.

```
function mi_funcion(parametro1, parametro2, parametro3) {

}
```

En caso de que se requiera retornar un valor, se hace uso de la sentencia return que permite devolver el resultado de una función.

Ejemplo: En el ejemplo que se verá a continuación, se muestra una función que se vuelve verdadero si un número es divisible por dos y falso en caso contrario.

```
function multipleReturn(numero) {
        var resto = numero % 2;
        if (resto == 0)
           return true;
        else
        return false;
}
```

Se sugiere construir una aplicación para el manejo de una contraseña de acceso a una página web (establecida con antelación) mediante el uso de sentencias repetitivas, donde el usuario tendrá tres oportunidades para ingresar.

4.10.6. Manejo de objetos

Un objeto es el concepto clave de la Programación Orientada a Objetos. La idea de objeto es similar a la del mundo real como, por ejemplo: un computador, un celular, una persona, una silla, un sofá; entonces, un objeto está conformado por un conjunto de atributos y

funciones (métodos) los cuales deben ser genéricos para toda la familia de dicha clase.

JavaScript es un lenguaje de programación que permite optimizar y mejorar las posibilidades de las páginas HTML5. El navegador se encarga de interpretar el código escrito y no hay necesidad de compilación.

En el lenguaje JavaScript no existen las clases para crear un nuevo objeto que tenga los mismos atributos que otro, estos deben ser copiados. A esta forma de trabajar se le llama "herencia", basada en prototipos. Los objetos que copian sus propiedades de otros objetos son llamados prototipos.

Para crear objetos en JavaScript, existen diferentes opciones. En el siguiente ejemplo, se hace por medio de una función:

```
var Estudiante = function()
{
      this.habla = true; //Atributo
      this.camina = function() //Método
      {
         alert ("un paso");
      }
};
```

Para instanciar un objeto de ese tipo, se utiliza la palabra clave new.
var unEstudiante = new Estudiante ();

Al pasar los parámetros a los objetos, se deben dar nombres cuando se crea el objeto con el constructor. Posteriormente, se indica el valor concreto cuando se solicite.

La definición de objeto, como instancia de una clase, no está definida hasta el momento en JavaScript. Con el siguiente caso, el programador puede utilizar una rutina como la siguiente:

```
var unEstudiante = function(nombre)
{
      this.lee = true; //Atributo
      this.nombre = nombre;
      this.camina = function() //Método
      {
            alert("un paso");
      }
};
var unEstudiante = new Gato("Francisco");
```

La otra forma de crear objetos es mediante el concepto de prototipo (prototype). Su utilidad es almacenar los métodos de los prototipos. A continuación, se presenta su uso:

```
//Creamos una función sin métodos
var Estudiante = function(nombre)
{
       this.lee = true; //Atributo
       this.nombre = nombre;
       };
       //Añadimos los métodos en su prototype
       Estudiante.prototype.camina = function()
       {
              alert ("un paso");
       }
};
```

El siguiente código fuente corresponde a un ejemplo donde se programa una página web que incluye la utilización de objetos en JavaScript:

```
<!DOCTYPE html>
<!-- FUNDAMENTOS DE PROGRAMACION WEB -->
<!-- Ejemplo JavaScript-->
<html>
<head>

    <script type="text/javascript">
       var persona = {
        nombre: 'Miguel',
        saludar: function(){
          alert("Hola me llamo " + this.nombre);
        }
       };
      //var persona = new persona();
      persona.saludar();
    </script>
    <meta charset="utf-8" />
    <title>Ejemplo de objeto javascript HTML5</title>

</head>
<body>
<header>
  Etiqueta header <br />
</header>

        <footer>
        Estas en la etiqueta header footer<br />
        </footer>

</body>
</html>
```

4.10.7. Clases predefinidas

JavaScript tiene una serie de clases predefinidas, entre las que están:

Clase Date: Permite el manejo de fechas y horas, para lo cual se debe definir un objeto de la "Clase Date".

Ejemplo:
Creación de un objeto de la clase:
fecha = new Date();

Un programa "web complete" sería:

```
<!DOCTYPE html>
<!-- Fundamentos de JavaScript -->
<html>
<head>
 <meta charset="utf-8" />
 <title>Ejemplo Clase Date</title>
<script language="JavaScript">
  function imprimirFechaActual()
  {
    var fecha=new Date();
    document.write('Hoy es ');
    document.write(fecha.getDate()+' / ');
    document.write((fecha.getMonth()+1)+' / ');
    document.write(2000+(fecha.getYear()-100));
    document.write('<br />');

  }
  //Llamada a la función
  imprimirFechaActual();
 </script>

</head>
<body>
<header>
<h3> Introduccion a la Clase Date </h3>
</header>
</body>
</html>
```

Clase Array: Un Array o arreglo, es una estructura que permite almacenar una lista finita de datos de un mismo tipo.

La creación de un array se puede realizar de la siguiente manera:
var nombr_array = new Array(tamaño_array)

El tipo de dato de la variable Array puede ser: cadenas, enteros, booleanos y su tamaño se establece según el requerimiento de la cantidad de datos a almacenar, como en el siguiente ejemplo:
var numeros = new Array(10);

La definición de arreglos se puede realizar de dos maneras diferentes:

- La primera forma define el arreglo con la información registrada en el array: var ciudades = new Array("Bogotá","Barranquilla",Cartagena","Villavicencio");
- La segunda, es asignando los valores a cada una de las posiciones del array:
  var ciudades = new Array (4);
  ciudades[0] = "Bogotá"; c
  iudades[1] = "Barranquilla";
  ciudades[2] = "Cartagena";
  ciudades[3] = "Villavicencio";

Un array puede verse como una lista de datos registrados bajo el mismo nombre, donde se accede a cada elemento individual del array por medio de un número entero denominado índice, 0 es el índice del primer elemento. Por otro lado, n-1 es el índice del último elemento; donde n, es el tamaño del array.

Con los arreglos, se pueden implementar una serie de propiedades o métodos como:

La propiedad "length" especifica cuántos elementos tiene el arreglo y se actualiza automáticamente para mantener su consistencia cuando se agregan nuevos elementos o se sobreescribe el arreglo.

Ejemplo:

```
var arreglo = new Array();
document.write('Longitud del arreglo es : ' + arreglo.length + '<br>');
arreglo= new Array(10); // aquí no se tiene en cuenta length
```

El método join() de un arreglo convierte todos los elementos de un arreglo a un string y los une, como en el siguiente caso:

```
var aNumeros = [1,2,3,4];
s = aNumeros.join();
document.write(s);
s = aNumeros.join(" ")
document.write('<br>');
document.write(s);
```

Lo anterior producirá la siguiente salida:
1,2,3,4
1 2 3 4

El método reverse() de un arreglo ordena al revés los elementos de un arreglo.

Por ejemplo:

```
s = aNumeros.reverse();
document.write('<br>');
document.write(s);
```

Salida:
4,3,2,1

El método sort() de un arreglo permite ordenar los elementos de un arreglo.

Por ejemplo:

```
var arregloNombres = ['Flor','Oscar','Miguel','Ángela','Julieth','Jefferson'];
arregloNombres = arregloNombres.sort();
document.write('<br>');
document.write(arreglosNombres);
```

Salida:
Ángela, Flor, Jefferson, Julieth, Miguel, Oscar

Los métodos push() y pop() de un arreglo, permiten anexar uno o más elementos al final del arreglo y regresa el último valor que anexó. El método pop() de un arreglo, elimina el último elemento de un arreglo y reduce la longitud del arreglo, retornando el valor que eliminó.

Clase Math: JavaScript tiene una serie de funciones asociadas a esta clase para realizar diferentes tipos de operaciones matemáticas.

Ejemplos:

- Math.exp(x) //Exponenciación.
- Math.sqrt(x) //Raíz cuadrada.
- Math.pow(x, y) //Potencia base (x) con exponente (y)
- Math.floor() //Redondea al número entero más cercano y menor.
- Math.ceil() //Redondea al número entero más cercano y mayor.
- Math.round() //Redondea al entero más próximo.
- Math.random() //Número aleatorio entre 0 y 1.
- Math.sin(x) //Seno, x expresada en radianes.
- Math.cos(x) //Coseno, x expresada en radianes.

Clase String: Esta clase se utiliza para la gestión de cadenas de caracteres, por ejemplo, cuando se requiere juntar o dividir cadenas, buscar texto o extraer parte de un texto. En JavaScript, todo texto debe estar encerrado entre comillas, dobles o simples. Por ejemplo, la cadena '37' no es el número tres y siete, sino la cadena formada por los caracteres 3 y 7.

Formularios y controles: Un formulario digital está diseñado con la finalidad que el usuario ingrese datos personales como: el nombre, apellidos, teléfono, correo, entre otros; con el propósito de ser almacenados y gestionados posteriormente. Los controles son cada uno de los diferentes componentes que se integran al formulario, como cajas de texto, etiquetas, link, imágenes, botones, etc; elementos con los que va a interactuar el usuario.

Eventos: Los eventos en JavaScript son una forma de controlar las

acciones de los usuarios, cuando estos realizan una visita a una página web. Por ejemplo, al pulsar clic en un botón, editar un campo de texto o cerrar la página.

El manejo de eventos permite el desarrollo de páginas interactivas, porque de esta manera, se puede responder a las acciones que los usuarios ejecutan.

Ejemplos:
1) Página web que permite ingresar los datos del nombre y la edad de una persona. También, al pulsar clic en el botón correspondiente, presenta los datos ingresados.

```
<!DOCTYPE html>
<!-- Fundamentos de JavaScript -->
<html>
<head>
 <meta charset="utf-8" />
 <title>Documento sin estilos</title>
 <script>
      function mostrar()
      {
        var nom=document.form1.nombre.value;
        var ed=document.form1.edad.value;
        alert('Ingreso el nombre:' + nom);
        alert('Y la edad:' + ed);
      }
</script>
</head>
<body>
      <header>
      <h3> Introduccion a Eventos </h3>
      </header>
      <form name="form1">
      <label for="email">Nombre:</label>
      <input type="text" name="nombre" /><br>
      <label for="email">Edad:</label>
      <input type="text" name="edad" /><br>
      <input type="button" value="Confirmar" onClick="mostrar()" />
      </form>
</body>
</html>
```

El evento "onclick" se ejecuta al pulsar clic al botón. Para incorporarlo en HTML, basta con utilizar el atributo "onclick" en un elemento HTML y especificar el código o función de JavaScript a ejecutar.

2) Página web que al pasar el mouse sobre los links, cambia de color el

fondo de la pantalla.

```
<!DOCTYPE html>
<!-- FUNDAMENTOS DE JavaScript -->
<html>
       <head>
        <meta charset="utf-8" />
        <title>Documento sin estilos</title>

       </head>
       <body>
       <header>
       <h3> Introduccion Eventos </h3>
       </header>

       <script language="JavaScript">
              function pintar(col)
              {
                document.bgColor=col;
              }
       </script>

       <a href="pagina1.html" onMouseOver="pintar('#87CEEB')"
           onMouseOut="pintar('#ffffff')">Azul</a>

       <a href="pagina1.html" onMouseOver="pintar('#FFA07A')"
            onMouseOut="pintar('#ffffff')">Salmon</a>

       <a href="pagina1.html" onMouseOver="pintar('#8FBC8F')"
           onMouseOut="pintar('#ffffff')">Verde</a>
       <br/>

        </body>
</html>
```

La descripción de los eventos utilizados para la página web del segundo ejemplo son los siguientes:

Evento	Descripción	Elementos para los que está definido
Onmouseout	El usuario retira el ratón del elemento	Todos los elementos
Onmouseout	El usuario posiciona el mouse sobre el elemento	Todos los elementos

Tabla 8: Eventos utilizados en la página web.

3) Página web que implementa eventos del teclado. Para este ejemplo, se utilizan dos tipos de archivos.

Código del archivo tipo CSS (estilos.css):

```
p {
  padding: 0.3em 1em;
  font-family: arial;
}

#info {
 width: 250px;float: left; border: 1px solid black;
 margin:20px;
}
```

Código del archivo tipo html:

```
<!DOCTYPE html>
<!-- FUNDAMENTOS DE JavaScript -->
<html>
<head>
 <meta charset="utf-8" />
     <title>Ejemplo Eventos teclado </title>
     <link  rel = "stylesheet" href = "css/estilos.css" />
<script>
window.onload = function() {
  document.onkeyup = imprimirInformacion;
  document.onkeypress = imprimirInformacion;
  document.onkeydown = imprimirInformacion;
}

function imprimirInformacion(elEvento) {
  var evento = window.event || elEvento;

  var texto = "Tipo de evento: " + evento.type + "<br>" +
              "Propiedad keyCode: " + evento.keyCode + "<br>" +
              "Propiedad charCode: " + evento.charCode + "<br>" +
              "Carácter pulsado: " +
String.fromCharCode(evento.charCode);
                          info.innerHTML +=
"<br>**************************<br>"
                         + texto;
}
</script>
</head>
<body>
<h1>Informacion del teclado</h1>
   <p>Pulsa una tecla</p>
   <div id="info"></div>
</body>
</html>
```

Para la aplicación del ejemplo anterior, existen tres eventos diferentes para las pulsaciones de las teclas (onkeyup, onkeypress y onkeydown), y dos tipos de teclas: las teclas normales (como letras, números y símbolos normales) y las teclas especiales (como ENTER, Alt, Shift, F1, F2, entre otras.)

Expresiones regulares: Una expresión regular, es un patrón que se crea utilizando caracteres del alfabeto con la ayuda de operaciones de unión, concatenación, etc. En JavaScript se emplean para validar información.

Ejemplos:

1) Página web que válida el ingreso de una cuenta de correo, teniendo en cuenta que solo debe tener caracteres, por ejemplo: alguien@gmail.com.

```
<!DOCTYPE html>
<!-- FUNDAMENTOS DE JavaScript -->
<html>
<head>
 <meta charset="utf-8" />
 <title>Ejemplo Expresion regular </title>
 <script>
    function validarCadenaExpReg() {
        // Expresion regular que representa una cuenta de correo válida
        cadena = "^[a-z]+@[a-z]+\.[a-z]{2,4}$";
        re = new RegExp(cadena);
        if (document.getElementById("textValidReg").value.match(re))
            alert("Cuenta de correo valida");
        else
            alert("Cuenta de correo errada");
    }
  </script>
  </head>
  <body>
  <header>
   <hgroup>
     <h3>Ejemplos expresion regular validar correo </h3>
     <h4>ejemplo miguel@gmail.com </h4>
   </hgroup>
  </header>
  <label>Email</label>
  <input id="textValidReg" type="text" value="" maxlength="65" /></br>
  <input onclick="validarCadenaExpReg()" type="button" value="Entrar"
/>
</body>
</html>
```

El significado de los símbolos utilizados para validar el correo en el ejercicio anterior es:

- **^:** Comienzo de la cadena.
- **$:** Final de la cadena.
- **\:** Permite usar caracteres especiales.
- **[a-z]:** Indica una opción de la letra "a" hasta la "z".

2) Programa que permite validar el formato de la fecha en dd/mm/aaaa.

```
<!DOCTYPE html>
<!-- FUNDAMENTOS DE JavaScript -->
<html>
<head>
 <meta charset="utf-8" />
 <title>Ejemplo Expresion regular </title>
<script>
  function validarFecha(){
     //var oracion=prompt('Ingrese una palabra','');
      var patron = /^\d{2}\/\d{2}\/\d{4}$/ ;
      re = new RegExp(patron);
     if (document.getElementById("textValidReg").value.match(re))
           alert("Fecha valida");
        else
           alert("fecha errada");
  }
</script>
</head>
<body>
  <label>Fecha</label>
  <input id="textValidReg" type="text" value="" maxlength="65" /></br>
  <input onclick="validarFecha()" type="button" value="Entrar" />
</body>
</html>
```

Se recomienda digitar los códigos de los ejemplos anteriores, ejecutarlos en un browser y observar los resultados generados.

4.11. Lecturas recomendadas

- Manejadores de eventos en línea en JavaScript.
- Otros tipos de manejadores de eventos.
- El método addEventListener().
- Consultar otras notaciones con la descripción y ejemplo de las expresiones regulares utilizadas en JavaScript para validar información.

4.12. Preguntas de revisión de conceptos

- ¿Qué es y para qué sirve JavaScript?
- ¿En qué casos es recomendable su aplicación?
- ¿Cómo se trabaja la herencia en JavaScript?
- ¿Cuáles son los demás principios de la POO que se pueden implementar en JavaScript?

4.13. Ejercicios propuestos

1) Corresponde a un texto adicional que se añade al código para explicar su funcionalidad y son una parte importante de la documentación de un programa.
a) Variables
b) Sentencias
c) Documentación
d) Estructuras de control

2) Es una orden que se le da al programa para realizar una tarea específica, esta puede ser: imprimir un mensaje en la pantalla, inicializar una variable, llamar a una función, entre otros.
a) Variables
b) Sentencias
c) Documentación
d) Estructuras de control

3) Es un nombre que se asocia con un espacio de la memoria del computador, en la que se guarda el valor asignado.
a) Variables
b) Sentencias
c) Documentación
d) Estructuras de control

4) Supongamos el siguiente fragmento de código:

```
var aux1=7;
var aux2=10;
var aux3=17; var aux4=20;
```

¿Cuál será el valor que se imprimirá al ejecutar la sentencia?:

```
alert((aux1<aux3)&&(aux2<aux4));
```

a) 7
b) 17
c) True
d) False

5) Valor que se imprimirá al ejecutar la sentencia: alert((aux1<aux3)||(aux2=aux4));
a) 7
b) 17
c) True
d) False

6) Valor que se imprimirá al ejecutar la sentencia: alert((aux1<aux3)||(aux2!=aux4));
a) 17
b) 20
c) True
d) False

7) Para el siguiente fragmento de código, el valor que se imprime es:

```
<script>      varn=2000;
var  x=1;
var a=10;
while (a<n) {
          x+=1;
          a*=10
             }

document.write(x);
</script>
```

a) 200.000
b) 200
c) 3
d) 4

8) Supongamos el siguiente fragmento de código, ¿cuál es el valor que se imprime?

```
<script>
var a=2;
var b=5;
var c=0;
while (b!=0) {
c+=a;
b-=1;
}
document.write(c);
</script>
```

a) 2
b) 10
c) 5
d) 4

9) Una forma de controlar las acciones de los usuarios cuando realizan una visita a una página web, es con la programación de:
a) Sentencias
b) Evento
c) Objetos
d) Expresión regular

10) Es un texto especial (con una sintaxis definida) para identificar un patrón de caracteres o un carácter en particular, y a su vez, permiten validar el contenido que ha sido ingresado.
a) Variaable
b) Evento
c) Arreglo
d) Expresión regular

11) Se puede observar como una lista de datos registrados bajo el mismo nombre.
a) Variaable
b) Evento
c) Arreglo
d) Expresión regular

12) Completar la información de la siguiente tabla correspondiente a

eventos en JavaScript:

Evento	Descripción	Elementos para los que está definido
Onblur	Al seleccionar el elemento o un componente, pierde el foco.	<button>, <input>, <label>, <select>, <textarea>, <body>
Onchange		
Onkeydown		
Onchange		
Onkeypress		
Onload		
Onselect		
Onsubmit		
Onunload		

Tabla 9: Eventos en JavaScript.

13) ¿Cuál es el objeto que permite calcular la potencia, a partir de la base y el exponente?
a) Math.exp( )
b) Math.sqrt( )
c) Math.pow( )
d) Math.floor( )

14) El método que elimina el último elemento de un arreglo:
a) Pop( )
b) Push( )
c) Remove( )
d) Delete( )

15) Completar la información de la siguiente tabla correspondiente a métodos asociados a la clase Date.

Método	Descripción
etDate()	
getDay()	
getHours()	
getMinutes()	

getMonth()	
getSeconds()	
getTime()	
getYear()	
setDate(día_mes)	
setDay(día_semana)	
setHours(horas)	
setMinutes(minutos)	
setMonth(mes)	
setSeconds(segundos)	
setTime(milisegundos)	
setYear(año)	
toGMTString()	

Tabla 10: Métodos asociados a la clase Date.

16) Elaborar una aplicación donde se implementen los siguientes aspectos:
a) Página que permita ingresar una cadena de cinco caracteres que comience y finalice con a (utilizar var patron=/â...a$/;)
b) Una función que permita ingresar tres dígitos seguido de tres caracteres (var patron=/^[0-9]3[a-z]3$/;)
c) Una función que permita validar el ingreso de por lo menos 5 caracteres (var patron=/?4?/;)

17) Genera un archivo HTML que represente una factura de la siguiente manera:

TIENDA DON PEPE				Factura No.
NIT.: 876.154.263.1				Fecha: (La del dia)
Cliente:		No. Cedula:		Teléfono:
Dirección:				
NO.	**ARTICULO**	**CANTIDAD**	**PRECIO UNITARIO**	**SUBTOTAL**
1	**	*		
2	**	*		
3	**	*		
4	**	*		
5	**	*		
6	**	*		
NETO				
IVA				
TOTAL				

Tabla 11: Ejemplo factura.

Instrucciones:

- Se debe colocar una lista de despliegue con los números del 1

al 10.

- Se debe colocar una lista de despliegue, donde deben colocar por lo menos 5 artículos de selección.
- En precio unitario, se debe colocar una caja de texto por cada fila, al igual que en subtotal y en el resto de los campos.
- SUBTOTAL: Debe ser calculado con java script con la siguiente fórmula: Cantidad * precio unitario.
- NETO= Suma de subtotales.
- IVA= Neto * 16%
- TOTAL = NETO + IVA

Bibliografía

Capitulo 5

5. Programación del lado del Servidor

5.1. Temática a desarrollar

- PHP
- Variables y constantes
- Operadores
- Estructuras de control
- Funciones
- Programación Orientada a objetos en PHP

5.2. Introudcción

La Programación del lado del servidor, es una tecnología que consiste en el procesamiento de una petición de un usuario mediante la interpretación de un script en el servidor web para generar páginas HTML dinámicas.

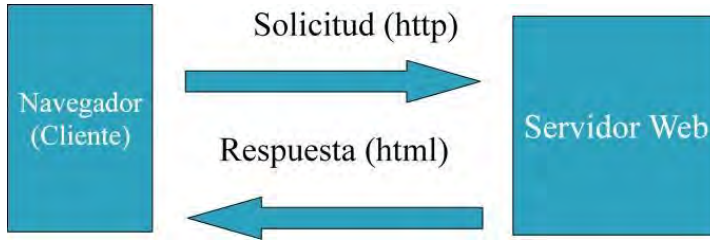


Ilustración 8: Programación del lado del servidor.

Todo lo que suceda dentro del servidor, es llamado procesamiento del lado del servidor, o serverside processing. Cuando la aplicación necesita interactuar con el servidor, ésta realiza una petición del lado del cliente (client-side request) desde el navegador a través de la red,

usando invocaciones remotas a métodos. La utilización de las diferentes aplicaciones o servicios de Internet se lleva a cabo respondiendo al llamado modelo "cliente-servidor".

Acrónimo recursivo de PHP: -Hypertext Preprocessor-, es un lenguaje de programación desarrollado por el programador de origen danés Rasmus Lerdorf, en 1994, con el propósito de facilitar el diseño de páginas web de carácter dinámico. Lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML.

A su misma vez, PHP es un lenguaje de programación que puede estar embebido en código HTML, como se puede evidenciar en el siguiente ejemplo:

```
<!DOCTYPE HTML>
<html>
    <head>
        <title>Ejemplo</title>
    </head>
    <body>

        <?php
            echo "Hola mundo . . .";
        ?>

    </body>
</html>
```

5.3. Variables

Las variables son útiles para almacenar valores con los que vayamos a trabajar en nuestro programa. Se definen anteponiendo el signo $.
Ejemplos:

```
$a = 5
$nombre = 'Miguel'
```

5.4. Funciones para trabajar con variables

- **Isset:** Saber si la variable ya existe.
- **Unset:** Para liberar la memoria.

- **Empty:** Saber si está vacía o es 0. Devuelve un valor "true" o "false" según corresponda.

Variables predefinidas:

- **$argc:** Número de argumentos pasados.
- **$argv:** Array con los argumentos.

5.5. Constantes

Las constantes son valores que no cambian durante la ejecución de un programa y se diferencian de las variables comunes, en que no utilizan el símbolo $ al declararlas o emplearlas. La visibilidad predeterminada de las constantes de clase es pública. Por buenas prácticas, se debe definir el nombre de las constantes con letra mayúscula sostenida.

Ejemplo:

```
<?php
class EjemploConstante
{
    const CONSTANTE = 'Esto es una Constante';

    function imprimirConstante() {
        echo self::CONSTANTE . "\n";
    }
}

echo EjemploConstante::CONSTANTE . "\n";

$nombreclase = "Ejemplo de Constante";
echo $nombreclase::CONSTANTE . "\n";

$clase = new EjemploConstante();
$clase->imprimirConstante();

echo $clase :: CONSTANTE."\n";
?>
```

5.6. Operadores aritméticos

Operadores aritméticos		
Ejemplo	**Nombre**	**Resultado**
$x + $y	Adición	Suma de *$a* y *$b*.
$x - $y	Sustracción	Diferencia de *$a* y *$b*.
$x * $y	Multiplicación	Producto de *$a* y *$b*.
$x / $b	División	Cociente de *$a* y *$b*.
$x % $y	Módulo	Resto de *$a* dividido *$b*.
$x ** $y	Exponenciación	Resultado de elevar *$x* la potencia *$y*.
+$x	Identidad	Conversión de $a a int o float según el caso.
-$x	Negación	Opuesto de $a.

Tabla 12: Operadores aritméticos.

5.7. Operadores de comparación

Permiten comparar dos valores. En la siguiente tabla se muestran estos operadores:

Operadores de comparación		
Ejemplo	**Nombre**	**Resultado**
$x == $y	Igual	**TRUE** si *$x* es igual a *$y* después de la manipulación de tipos.
$x === $y	Idéntico	**TRUE** si *$x* es igual a *$y*, y son del mismo tipo.
$x != $y	Diferente	**TRUE** si *$x* no es igual a *$y* después de la manipulación de tipos.
$x <> $y	Diferente	**TRUE** si *$x* no es igual a *$y* después de la manipulación de tipos.
$x !== $y	No idéntico	**TRUE** si *$x* no es igual a *$y*, o si no son del mismo tipo.
$x < $y	Menor que	**TRUE** si *$x* es estrictamente menor que *$y*.
$x > $y	Mayor que	**TRUE** si *$x* es estrictamente mayor que *$y*.
$x <= $y	Menor o igual que	**TRUE** si *$x* es menor o igual que *$y*.
$x >= $y	Mayor o igual que	**TRUE** si *$x* es mayor o igual que *$y*.

Tabla 13: Operadores de comparación.

A partir de la versión 7.0 de PHP, se incorporan los siguientes operadores:

Paso 1: $x <=>$y Un entero menor que, igual a, o mayor que cero cuando $x es respectivamente menor que, igual a, o mayor que $b.

Paso 2: $x ?? $y ?? $z El primer operando de izquierda a derecha que exista y no sea NULL. NULL si no hay valores definidos y no son NULL.

5.8. Operadores lógicos

Corresponde a los operadores and, or y not; como se observa en la siguiente tabla:

Operadores lógicos		
Ejemplo	**Nombre**	**Resultado**
$x and $y	And (y)	TRUE si tanto *$x* como *$y* son TRUE.
$x or $y	Or (o inclusivo)	TRUE si cualquiera de *$a* o *$y* es TRUE.
$x xor $y	Xor (o exclusivo)	TRUE si *$x* o *$y* es TRUE, pero no ambos.
! $x	Not (no)	TRUE si *$x* no es TRUE.
$x && $y	And (y)	TRUE si tanto *$x* como *$y* son TRUE.
$x || $y	Or (o inclusivo)	TRUE si cualquiera de *$x* o *$y* es TRUE.

Tabla 14: Operadores lógicos.

5.9. Estructuras de control

5.9.1. Selectivas

Determinan que parte del código se ejecuta en función del cumplimiento de una condición.

- **If:** Sentencia utilizada en lenguajes como C, Java, C#, PHP, etc. También, permite la ejecución condicional de fragmentos de código. Una forma de utilizarla es:

```
<?php
if (condición) {
  sentencias . . .
}
?>
```

Ejemplo:

```php
<?php
if ($x > $y) {
  echo "x es mayor que y";
}
?>
```

- **La sentencia else:** Se ejecuta en caso de que la expresión en la sentencia if se evalúe como falsa.

Ejemplo:

```php
<?php
if ($x < $y) {
  echo "x es menor que y";
} else {
  echo "x NO es menor que y";
}
?>
```

- **Elseif/else if:** Es una sentencia que combina if y else. Así como el else, se ejecuta cuando la expresión if original se evalúe como falsa. Sin embargo, a diferencia de else, esa expresión alternativa sólo se ejecutará si la expresión condicional del elseif se evalúa como verdadera.

Ejemplo:

```php
<?php
if ($x > $y) {
    echo "x es mayor que y";
} elseif ($x == $y) {
    echo "x es igual que y";
} else {
    echo "x es menor que y";
}
?>
```

- **Switch:** Es una forma de expresión de un anidamiento múltiple de instrucciones "if else".

```php
<?php
if ($x == 0) {
    echo "s es igual a 0";
} elseif ($x == 1) {
    echo "x es igual a 1";
} elseif ($i == 2) {
    echo "x es igual a 2";
}

switch ($x) {
    case 0:
        echo "x es igual a 0";
        break;
    case 1:
        echo "x es igual a 1";
        break;
    case 2:
        echo "x es igual a 2";
        break;

   Default:

      Echo "Es otro número";

      Break;
}
?>
```

Como se puede observar en el código anterior, se recorren todas las sentencias "case", comparando el valor que hay en cada una de ellas con el valor de la variable. Cuando estos valores coincidan, se ejecutará el fragmento de código correspondiente al case. En caso de que no concuerde ninguno, se ejecuta el código asociado a la sentencia default.

5.10. Ciclos o sentencias iterativas

Al igual que en muchos otros lenguajes de programación, se tienen varias formas de generar estructuras cíclicas. Para el caso de PHP, se tienen:

- While
- Do while
- For
- Foreach

- **While:** En PHP, el ciclo while ejecuta las sentencias anidadas. Cuando la expresión while se evalúe como verdadera, la ejecución no se detendrá hasta el final de la iteración. Es de tener en cuenta que si la expresión while se evalúa como false desde el inicio, las sentencias incluidas dentro del ciclo no se

ejecutarán.

Ejemplos:

```php
<?php

$x = 1;
while ($x <= 10) {
    echo $x;
    $x++;
}
?>
```

Otra forma de generar el while sin el uso de corchetes es:

- **Do while:** El ciclo do-while es muy similar al ciclo while, con la diferencia que la condición es verificada al final de cada iteración. La diferencia con el bucle while, es que en la primera iteración siempre se ejecutaría (por lo menos una vez) las sentencias internas del bucle, caso que no ocurre con la sentencia while.

Ejemplo:

```php
<?php
$x = 1;
do {
    echo $x;
    $x++;
} while ($x <= 10);
?>
```

- **Ciclo for:** El ciclo for tiene la siguiente estructura:

```php
<?php

   for (expres1; expres2; expres3) {
    sentencias. . .;
}
?>
```

La primera expresión **(expres1)** es ejecutada al inicio del bucle y corresponde a la asignación de los valores de inicio de la variable que dirige el ciclo.

La **expres2** corresponde a la condición de parada, la cual, si se evalúa como verdadera, el ciclo continúa y se ejecutan las sentencias que se encuentren dentro de los corchetes ({ }). Si se evalúa como falsa, finaliza la ejecución del ciclo.

La **expres3** permite realizar los incrementos o decrementos correspondientes a la variable de control, y se ejecuta una vez la expresión booleana sea evaluada. Es de vital importancia dicha expresión, pues hace que no se generen ciclos infinitos, los cuales pueden generar errores de lógica dentro del programa.

Ejemplo:

```php
<?php

   for ($i = 1; $i <= 10; $i++) {
    echo $i;
}
?>
```

5.11. Funciones en Php

Corresponde a un bloque de código que realiza una operación específica, que en ocasiones pueden o no retornar algún valor.

5.11.1. Funciones definidas por el usuario

Según sea el requerimiento en la implementación de una aplicación de software, el desarrollador construye sus propias funciones, dependiendo de las necesidades y de la solución que se le de a las mismas.

La estructura base para la definición de una función es la siguiente:

```php
<?php
function ejemplo ($parámetro 1, $parámetro _2, /* ..., */ $parámetro n)
{
    echo "Función de ejemplo.\n";
    return $valor_retornando;
}
?>
```

5.11.2. Funciones con parámetros o argumentos

Es una función que recibe una lista de expresiones delimitadas por "comas" como entradas para su procesamiento interno, las cuales son evaluadas de izquierda a derecha.

```php
<?php
function imprimir($entrada)
{
 echo "$entrada";
}
?>
```

5.11.3. Funciones y métodos predefinidos

Son una serie de funciones que ya están definidas en PHP y solo se debe hacer el respectivo llamado.

Ejemplo:
Abs: Retorna el valor absoluto de un número. Hay que tener en cuenta que algunas de estas funciones necesitan de la inclusión de archivos dentro de la ejecución del script, con el fin de que dicha función sea reconocida por el intérprete.

```php
<?php
      echo abs(-8.3); // 8.3 (double/float)
      echo abs(3);    // 3 (integer)
      echo abs(-15);   // 15 (integer)
?>
```

Cos: Retorna el valor del coseno de número.

```php
<?php
  echo cos(180); // -1
?>
```

5.12. Orientación a objetos en PHP

Actualmente, todos los proyectos de desarrollo de software están implementados por medio de la utilización de la Programación Orientada a Objetos (POO), paradigma que permite organizar conceptos, generar procesos de desarrollo sólido y realizar una estructuración bastante fuerte del proyecto desde el punto de vista de arquitectura. En este sentido, es necesario conocer a profundidad la forma de trabajo de la POO, como cimiento para la construcción de diferentes tipos de software.

Class: La definición básica de clases comienza con la palabra clave: class, seguido de un nombre de clase, continuado por un par de llaves que encierran las definiciones de las propiedades y métodos pertenecientes a la clase.

Ejemplo:

```php
class MyClass { }
```

New: Para crear una instancia de una clase, la palabra clave new debe ser usada. Un objeto siempre se creará a menos que el objeto tenga un constructor.
Ejemplo:

```
objeto -> new MyClass ( )
```

Las **variables** pertenecientes a clases se llaman "propiedades". También, se les puede llamar usando otros términos como "atributos" o "campos", pero para los propósitos de esta referencia, se va a utilizar el término "propiedades". Éstas se definen usando una de las palabras clave, ya sea: public, protected, o private, seguido de una declaración normal de variable. Dichas palabras claves permiten definir el nivel de permeabilidad o visibilidad que las propiedades tienen frente a otros objetos de otras clases. Por ejemplo: private $saldo;

Los **constructores** son métodos especiales para las clases, los cuales permiten inicializar las propiedades del objeto con valores que se puedan necesitar antes de ser usados dentro de un programa. Se definen mediante el término "lista de parámetros".

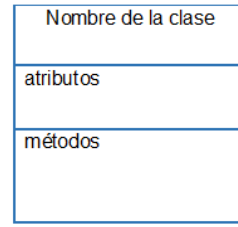
La **visibilidad** de una clase, propiedad o método se puede definir anteponiendo una de las palabras claves, ya sea: public, protected o private en la declaración. Miembros de clases declarados como "public" pueden ser accedidos por cualquier tipo de objeto. Para privados, sólo pueden ser accedidos por objetos de la misma clase y, en el caso de protegidos, sólo pueden ser accedidos por un objeto específico. A nivel del Lenguaje de Modelamiento Unificado (UML), se utilizan formas gráficas de visualizarlo, como las siguientes: (+) public; (-) private; (#) protected.

Operador: Llamado operador de referencia, el cual permite acceder a las propiedades y métodos de un objeto (un guión seguido por un signo mayor que, sin espacio entre ellos).

```
$unSaludo=new Saludo (" nombre "); // crea el objeto y utiliza su constructor
$unSaludo->imprimirMensaje(); // accede al método imprimirMensaje();
```

Métodos: Son los algoritmos o funcionalidades de las clases, y en PHP se definen a partir de las funciones.

Código PHP	Diagrama de clases

<?php
class NombreClase {
//atributos
var $variable1;
var $variable2;
 	
// métodos
function metodo1() {
 	
}
 	
function metodo2($param1, $param2) {
 	
}
}
?>
	

Para una mayor familiarización con la temática, se propone realizar las siguientes actividades:

Actividad 1

-Descargar Notepad++ e instalarlo.

-Descargar la versión de XAMPP para Windows e instalarlo. Comprobar dicha instalación.

Actividad 2

-Aplicación, que imprime un mensaje de un saludo y el nombre.

Diagrama de clases:

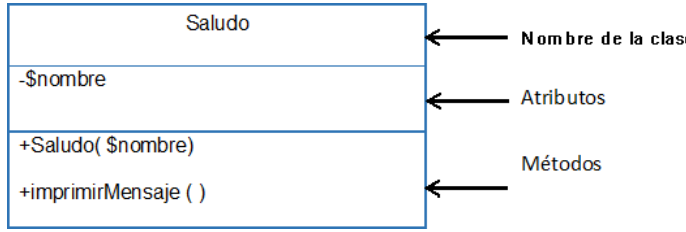


Tabla 15: Diagrama de clases.

-Digitar el código utilizando Notepad++ y grabarlo con el nombre de Saludo.php.

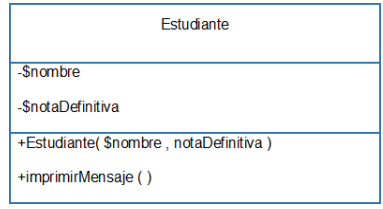
-Cambiar dónde está "nombre" por su nombre en el Saludo ("..."); guardar y ejecutar.

```php
<?php
class Saludo
{
    // Declaración de los atributos
    private $nombre;

   // Declaración del método

   public function Saludo($nombre) {
    $this->nombre=$nombre;
   }
    public function imprimirMensaje() {
        echo "Hola ".$this->nombre;
    }
}

$unSaludo=new Saludo (" nombre ");
$unSaludo->imprimirMensaje();
?>
```

Actividad 3

-Programa que imprime el nombre del estudiante y su nota definitiva.

Diagrama de clases:



-Digitar el siguiente código:

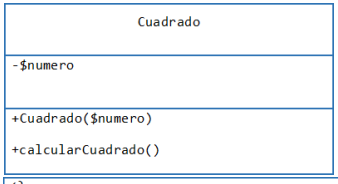
Cambiar dónde está "nombre" por su nombre en el Saludo (Çarlos..."); utilizar el notepad++ (grabarlo con el nombre de Estudiante.php

```php
<?
class Estudiante {
	private $nombre;
	private $notaDefinitiva;

	function Estudiante( $nombre , $notaDefinitiva )
	{
		$this->nombre = $nombre;
		$this->notaDefinitiva = $notaDefinitiva;
	}

	public function imprimirMensaje() {
		echo "Nombre: ".$this->nombre;
		echo "<br /> Nota: ".$this->notaDefinitiva;

	}

}// fin de la clase

$unEstudiante=new Estudiante (" Oscar ",4.5);
$unEstudiante->imprimirMensaje();
?>
```

Actividad 4

-Aplicación que eleva un número al cuadrado.

Diagrama de clases:




```php
<?
class Estudiante {
	private $nombre;
	private $notaDefinitiva;

	function Estudiante( $nombre , $notaDefinitiva )
	{
		$this->nombre = $nombre;
		$this->notaDefinitiva = $notaDefinitiva;
	}

	public function imprimirMensaje() {
		echo "Nombre: ".$this->nombre;
		echo "<br /> Nota: ".$this->notaDefinitiva;

	}

}// fin de la clase

$unEstudiante=new Estudiante (" Oscar ",4.5);
$unEstudiante->imprimirMensaje();
?>
```

Actividad 5

Aplicación que permite calcular el cuadrado de un número a partir de la construcción de un formulario para capturar un número en un archivo HTML5 y calcular el cuadrado de este en un archivo de PHP.

Digitar formulario (llamarlo formulario.html)

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="utf-8" />
    <title>Formulario Cuadrado </title>
</head>
<body>
    <header>
       <h2> Cuadrado de un numero</h2>
    </header>
      <article>
        <form method="post" action="Cuadrado.php">
             Numero:
 <input type="text" name="num" />
 <br >
<input type="submit" value="Enviar">
   </form>
</article>
    <footer>
        <h4> curso bases orientadas a la web</h4>
     </footer>
</body>
</html>
```

Digitar programa de php (grabarlo con el nombre de Cuadrado.php)

```
<?
class Cuadrado {
    private $resultado;
    function proceso($a) {
          $this->resultado=$a*$a;
    }

    function imprimir () {
         return $this->resultado;
    }
}
   $objeto= new Cuadrado;
   $nro= $_REQUEST['num'];
   $objeto->proceso($nro);
   echo $objeto->imprimir ();
 ?>
```

Actividad 6

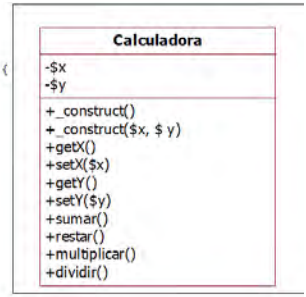
Crear el siguiente programa, el cual representa una calculadora. Tener en cuenta que, para este efecto, se están trabajando diferentes formas de constructor, el cual recibe o no parámetros de construcción. A esto se le denomina una sobrecarga de un método, lo cual obedece al polimorfismo a nivel de método.

Adelantar la digitación del archivo correspondiente denominado **calculadora.php** y del respectivo **operación.php**. Tener en cuenta

que se utiliza un formulario en html para poder enviar los datos a la clase principal.

-Clase: una calculadora cuya responsabilidad es realizar el cálculo de las operaciones básicas de suma, resta, multiplicación y división.

```php
<?php
class Calculadora{
      private $x;
      private $y;
      public function __construct()  {
            $this->x=0;
            $this->y=0;
      }
      public function SetX($x)  {
            $this->x=$x;
      }
      public function getX()  {
            return $this->x;
      }
      public function SetY($y)  {
            $this->y=$y;
      }
       public function getY()  {
            return $this->y;
      }
      public function sumar()  {
            return ($this->x+$this->y);
      }
      public function restar()  {
            return ($this->x-$this->y);
      }
      public function multiplicar()  {
            return ($this->x*$this->y);
      }
      public function dividir()
      {
            return ($this->x/$this->y);
      }
}
?>
```



Clase que tiene como responsabilidad el ingreso de los valores y el tipo de operación que muestra el resultado correspondiente:

```php
<?php
    include("Calculadora.php");
    class Operacion
    {
       function procesarDatos() {
           $caddio=new Calculadora();
          $caddio->setX($_POST['n1']);
           $caddio->setY($_POST['n2']);
           $x=$caddio->getX();
           $y=$caddio->getY();
           $suma = $caddio->sumar();
           $resta = $caddio->restar();
           $prod = $caddio->multiplicar();
           $divi = $caddio->dividir();
            echo "$x + $y = $suma <br />";
            echo "$x - $y = $resta <br />";
            echo "$x * $y = $prod <br />";
            echo "$x/ $y = $divi <br />";
       }
    }
    $obj = new Operacion();
    $obj ->procesarDatos();
    echo "<a href=\"FormCalculadora.html\"> Formulario </a>";
?>
```

Formulario html en el que se ingresan los valores y el tipo de operación a realizar:

```
<!DOCTYPE html>
<head><title>Calculadora </title></head>
<body>
<form method="post" name="formulario" action="Operacion.php">
    <table>
        <tr> <td><b>Numero Uno</b></td><td><input type="text" name="n1" size = "30"></td>
        </tr>
        <tr>
            <td><b>Numero Dos</b></td><td><input type="text" name="n2" size = "30"></td>
        </tr>
        <tr> <td><input type="submit" value="Calcular" name="anexar"></td><td>
            <input type="reset"value="Limpiar Datos" name="limpiar"></td> </tr>
    </table>
</form>
</body>
</html>
```

Actividad 7

Aplicación en el que a partir de un formulario en html, se reciben los datos de un cliente y se visualizan. Para tal efecto, se hace uso de dos clases en PHP.

Formulario html

```
<!DOCTYPE html>
<html lang="es">
<head>
        <meta charset="utf-8">
        <title>HTML5 Contact Form</title>
        <link  rel = "stylesheet" href = "css/estilos.css" />
</head>
<body>
<form class="contact_form" action="Control.php" method="post">
    <ul>
        <li>
             <h2>Formulario de datos personales</h2>
        </li>
                <li>
            <label for="name">Codigo:</label>
            <input type="text" name="codigo" placeholder="123456" required />
        </li>
        <li>
            <label for="name">Nombre:</label>
            <input type="text" name="nombre" placeholder="Miguel " required />
        </li>

                <li>
            <label for="name">Apellido:</label>
           <input type="text" name="apellido" placeholder="Hernandez" required/>
        </li>

        <li>
            <label for="email">Email:</label>
            <input type="email" name="email" placeholder="mhernandez@ejemplo.com.co" required />
        </li>
        <li>
            <label for="telefono">Telefono:</label>
            <input type="tel" name="telefono" placeholder="2916520" required />
        </li>
        <li>
            <label for="Mensaje">Mensaje:</label>
            <textarea name="mensaje" cols="40" rows="6" required ></textarea>
        </li>
        <li>

         <button class="submit" type="submit" name="anexar">Enviar </button>
             <button class="submit" type="submit" name="listar">Listar </button>
             <button class="submit" type="submit" name="borrar">Borrar  </button>
             <button class="submit" type="submit" name="limpiar">Limpiar </button>

        </li>
    </ul>
</form>
</body>
</html>
```

Archivo css

```
@CHARSET "ISO-8859-1";
.contact_form h2 {
    background: none repeat scroll 0 0 #F3F3F3;
    border-radius: 5px;
    color: #5CD053;
    display: block;
    font-family: sans-serif;
    font-size: 20px;
    padding: 5px;
    text-shadow: 1px 1px 1px #CCCCCC;
    width: 433px;
}
/*--- estilos para los ul y li del formulario ---*/
.contact_form ul {
    width:750px;
    list-style-type:none;
      margin:0px;
      padding:0px;
}
.contact_form li{
      padding:12px;
      border-bottom:1px solid #eee;
      position:relative;
}
.contact_form label {
    color: #555555;
    display: inline-block;
    float: left;
    font-family: sans-serif;
    font-size: 13px;
    font-weight: bold;
    margin-top: 3px;
    padding: 3px;
    width: 90px;
}
.contact_form input {
      height:20px;
      width:220px;
      padding:15px 12px;
}
.contact_form textarea {
      padding:8px;
      width:300px;
}
.contact_form button {
      margin-left:50px;
       padding:12px;
      width:110px;
}
```

Persona.php

```
<?php
 class Persona {
    private $codigo;
    private $nombre;
    private $apellido;
    private $email;
    private $telefono;
    private $mensaje;

     function Persona() {
        $this->codigo = "";
        $this->nombre = "";
        $this->apellido = "";
       $this->email = "";
       $this->telefono = "";
       $this->mensaje = "";
    }

    function Persona($codigo,$nombre, $apellido, $email, $telefono, $mensaje)
{
       $this->codigo = $codigo;
       $this->nombre = $nombre;
       $this->apellido = $apellido;
       $this->email = $email;
       $this->telefono = $telefono;
       $this->mensaje = $mensaje;
    }

      function getCodigo() {
        return $this->codigo;
    }

    function setCodigo($codigo) {
       $this->codigo = $codigo;
    }

    function getNombre() {
        return $this->nombre;
    }

    function setNombre($nombre) {
       $this->nombre = $nombre;
    }

      function getApellido() {
        return $this->apellido;
    }

    function setApellido($apellido) {
       $this->apellido = $apellido;
    }
```

```
    function getEmail() {
        return $this->email;
    }

    function setEmail($email) {
       $this->email = $email;
    }

    function getTelefono() {
        return $this->telefono;
```

```
    function setTelefono($telefono) {
       $this->telefono = $telefono;
    }

    function getMensaje() {
        return $this->mensaje;
    }

    function setMensaje($mensaje) {
       $this->mensaje = $mensaje;
    }
}
?>
```

Control.php

```
<?php
     include("Persona.php");
     class Control  {
              private $unaPersona;

              function Control( ) {
               $this->unaPersona = new Persona();
         }

             function registrarDatos() {
                    $this->unaPersona->setCodigo($_POST['codigo']);
                  $this->unaPersona->setNombre($_POST['nombre']);
                    $this->unaPersona->setApellido($_POST['apellido']);
            $this->unaPersona->setEmail($_POST['email']);
                    $this->unaPersona->setTelefono($_POST['telefono']);
                    $this->unaPersona->setMensaje($_POST['mensaje']);
             }

        function imprimirResultados() {
            $codigo=$this->unaPersona->getCodigo();
            $nombre=$this->unaPersona->getNombre();
             $apellido=$this->unaPersona->getApellido();
            $email = $this->unaPersona->getEmail();
            $telefono = $this->unaPersona->getTelefono();
            $mensaje = $this->unaPersona->getMensaje();
            echo "Codigo: $codigo  <br />";
            echo "Nombre: $nombre <br />";
                    echo "Apellido: $apellido <br />";
            echo "Email: $email <br />";
                    echo "Telefono $telefono <br />";
                    echo "Mensaje $mensaje <br />";

       }
    }
    $obj = new Control();
    $obj -> registrarDatos();
    $obj ->imprimirResultados();
    echo "<a href=\"index.html\"> Formulario </a>";
?>
```

5.13. Revisión de conceptos

Revisar los conceptos con sus respectivos ejemplos referentes a:

- Clases
- Atributos
- Objetos
- Métodos
- Métodos "set" y "get"

5.14. Ejercicios propuestos

1) A partir del siguiente diagrama de clases correspondiente a un triángulo, codificarlo en PHP:

Tabla 16: Ejercicio codificación en PHP.

2) Elaborar una aplicación que permita calcular el perímetro y área del Rectángulo.

3) Construir un proyecto que permita obtener la siguiente información de la circunferencia:

- **Longitud de la circunferencia**
- **Área de la circunferencia**
- **Diámetro de la circunferencia**

4) Construir una aplicación que permita capturar los datos de un estudiante, como el: código, nombres, apellidos, dirección, teléfono y cuenta de correo electrónico institucional en un formulario de HTML5. Crear la clase (Estudiante) con los mismos atributos del estudiante en "visibilidad privada", e implementar un método constructor y un método que imprima los datos enviados mediante el formulario.

Bibliografía

Capítulo 6

6. Programación con capa de persistencia (Base de datos)

6.1. Temática a desarrollar

- Base de datos.
- Paquetes de desarrollo.
- Creación de base de datos y tablas.
- CRUD para la base de datos con PHP.

6.2. Introducción

Las temáticas desarrolladas en los capítulos anteriores, incluyendo los ejercicios resueltos y propuestos, los datos arrojados se obtenían de carácter temporal o volátil; esto es, en memoria principal. Las aplicaciones que se acercan más a la realidad, necesariamente requieren de un almacenamiento permanente; esto es, en memoria auxiliar o secundaria. En este sentido, se debe incluir la gestión básica alrededor de lo que conocemos como base de datos.

De esta manera, la temática que se va a desarrollar dentro de este capítulo, tiene especial relación con las siguientes funcionalidades:

- La creación de una pequeña base de datos, la cual va a ser construida dentro de XAMPP y administrada por medio de PhpMyAdmin.
- La creación de funciones y librerías de PHP para el acceso a dichas tablas y a la base de datos.
- La generación de funcionalidad que permita las operaciones básicas sobre las tablas de la base de datos (CRUD), las cuales son: inserción, actualización y borrado de registros.

En la figura *-9-* se visualizan algunas consideraciones de PHP con bases de datos.

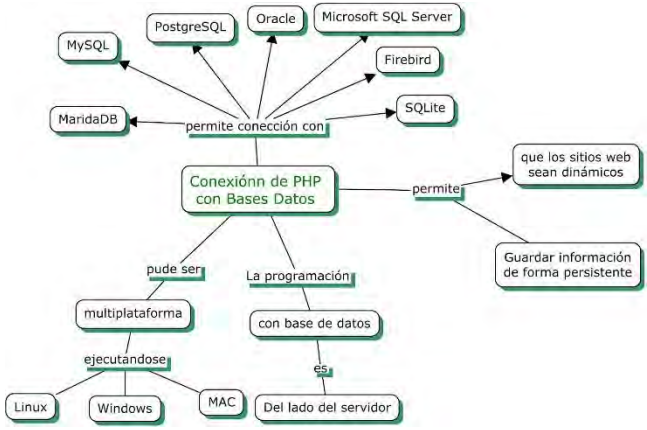


Ilustración 9: Mapa conexión PHP con bases de datos.

6.3. Base de datos

Dentro de los procesos computacionales, siempre ha existido la necesidad de recibir información y guardarla de manera persistente, de tal modo, que se pueda acceder a ella en el momento que se requiera, bajo unas consideraciones precisas de manejo de información. En la figura *-10-*, se visualizan algunas características de las bases de datos.

Ilustración 10: Mapa conexión PHP con bases de datos.

En este sentido, la solución que la ingeniería le dio a este problema se denomina "Base de Datos", la cual permite realizar ese almacenamiento de información cumpliendo ciertas condiciones específicas para dicho fin, por medio del establecimiento de políticas de trabajo, conectividad y relación entre la información que dentro de su estructura se genera.

De esta forma, la estructura que la base de datos tiene para su gestión es la siguiente:

SGBD: Sistema Gestor de base de datos, el cual permite realizar las acciones básicas sobre cualquier base de datos y genera la interfaz para crear la comunicación entre la información y el exterior.

Base de datos: Es el objeto macro dentro del cual se van a almacenar estructuras de información que van a permitir adelantar el registro, administración y gestión de la información correspondiente. Dicho sistema está demarcado gráficamente de la siguiente manera:

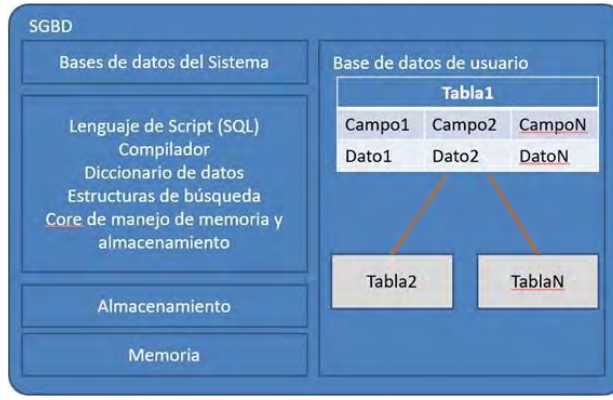


Ilustración 11: Mapa de conexiòn PHP con bases de datos.

En este gráfico, se puede observar cómo se definen innumerables bases de datos de usuario, dentro de la cual existen diferentes funcionalidades y estrategias de trabajo que permiten darle el poder de almacenar y generar estructuras de búsqueda de información lo suficientemente sólidas para las aplicaciones que se requieren montar.

El estudio de las bases de datos es una temática completamente

grande y se deja al lector su estudio minucioso, junto con el empoderamiento de estas temáticas.

6.4. Software para administración de bases de datos

Así como ya se tiene instalado una base de datos la cual, generalmente, se encuentra en infraestructura específica para x fin, el acceso a dicho sistema se hace por medio de clientes, los cuales permiten utilizar las funcionalidades de la base de datos para los fines pertinentes. En este sentido, existen clientes de diferentes bondades, tales como: clientes gráficos, web, línea de comando (o Shell) y conectores directos desde los lenguajes de programación.

6.5. Paquetes de desarrollo

Actualmente en el mercado, se tienen paquetes que instalan automáticamente los diferentes productos para adelantar un desarrollo web sin tener que configurar cada una de las aplicaciones, lo cual conllevaba mucho tiempo y conocimientos. De esta manera, los paquetes que se instalan son:

- Servidor de aplicaciones.
- Base de datos.
- Lenguaje de programación.
- Software para administración de la base de datos (opcional).

En el mercado, actualmente existen varios paquetes que cumplen con estos requerimientos y que ayudan a los desarrolladores a hacer despliegues de manera muy sencilla y fácil. Algunos de ellos son:

NO	NOMBRE	SISTEMA OPERATIVO	ENLACE
1	APPSERV	WINDOWS	https://www.appserv.org
2	FOXSERV	WINDOWS	http://sourceforge.net/projects/foxserv/
3	PHP TRIAD	WINDOWS	http://sourceforge.net/projects/phptriad/
4	EASYPHP	WINDOWS	http://www.easyphp.org
5	E-NOVATIVE WAMP	WINDOWS	http://www.e-novative.info/software/wamp.php
6	WAMP	WINDOWS	http://www.wampserver.com/en/
7	XAMP	WINDOWS	https://www.apachefriends.org/es/index.html
8	LAMP	LINUX	https://bitnami.com/stack/lamp/installer
9	MAMP	MAC	https://www.mamp.info/en/

Tabla 17: Paquetes.

Para adelantar el proyecto de este libro, se trabajará con la versión de APPSERV, la cual tiene la siguiente configuración:

- Apache 2.4.25
- PHP 5.6.30
- PHP 7.1.1
- MySQL 5.7.17
- PhpMyAdmin 4.6.6
- Soporte TLS, SSL or https para implementación de sitios seguros

Una vez descargada la aplicación, se pasa a su instalación (la cual es bastante sencilla). Después con su wizard, solo pide unos cuantos datos; y a partir de esto, todas las aplicaciones se instalan y quedan listas para ser utilizadas.

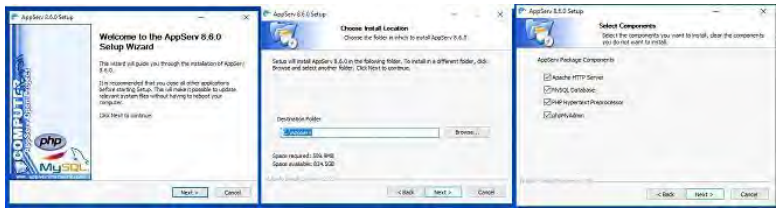


Ilustración 12: Proceso de instalación.

Ilustración 13: Proceso de instalación.

Para probar que todo salió bien dentro de la instalación, se debe abrir un web browser y en la url ejecutar http://localhost.

Ilustración 14: Prueba ejecución de la URL.

6.6. Creación de una base de datos y una tabla por medio de PhpMyAdmin

Una vez se tenga lista la instalación y funcionando todos los servicios (servidor de aplicaciones y base de datos), se pasa a trabajar sobre la base de datos. Para ejecutar dicha tarea, se debe ejecutar dentro del browser la siguiente URL: http://localhost/phpMyAdmin, lo cual abrirá una página parecida a la siguiente:

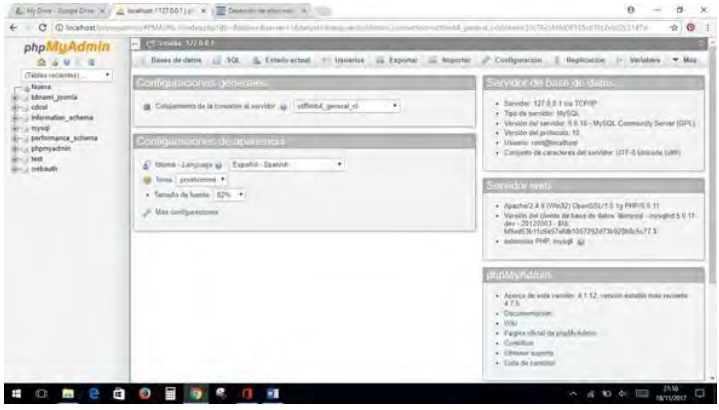


Ilustración 15: Entorno PhpMyAdmin.

Desde esta página, se puede realizar la gestión de las bases de datos que se han creado o crear las que se necesiten. Para efectos del ejercicio, se va a adelantar la creación de una base de datos llamada "ejercicio".

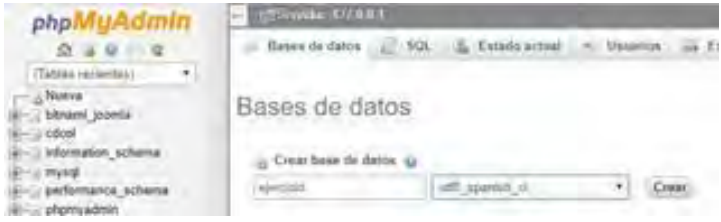


Ilustración 16: Creación base de datos.

Una vez creada la base de datos, se crea una tabla denominada "libros", en la cual se va a registrar la información dispuesta para tal fin.

Ilustración 17: Ejemplo creación base de datos.

Dicha tabla va a contener la siguiente estructura:

Tabla 18: Ejemplo estructura de tabla.

En el caso de necesitar el script de creación de la base de datos, se debe ejecutar lo siguiente:

```
//crear base de datos
CREATE DATABASE ejercicio DEFAULT CHARSET=utf8 COLLATE=utf8_spanish_ci;
//Usar la base de datos creada
use ejercicio;
//crear la tabla libro
CREATE TABLE IF NOT EXISTS `libro` (
  `idLibro` int(11) NOT NULL AUTO_INCREMENT,
  `nombre` varchar(50) COLLATE utf8_spanish_ci NOT NULL,
  `editorial` varchar(40) COLLATE utf8_spanish_ci DEFAULT NULL,
  `anio` varchar(4) COLLATE utf8_spanish_ci NOT NULL,
  `autor` varchar(100) COLLATE utf8_spanish_ci NOT NULL,
  PRIMARY KEY (`idLibro`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_spanish_ci
AUTO_INCREMENT=1 ;
```

6.7. Conectividad desde PHP a MySQL

Para poder adelantar la conectividad entre el lenguaje de aplicación (PHP en este caso) y la base de datos (MySQL), se tienen diferentes directivas. Las funciones concretas de MySQL que se van a utilizar para realizar estas operaciones son:

Conexión con el servidor de bases de datos:

- **mysqli_connect("servidor",usuario","password","bd"):** Esta directiva permite conectar el servidor con las credenciales definidas y a la base de datos discriminada con "bd". Este objeto devuelve una variable de tipo conexión.

Ejecutar un script o un enviar una instrucción SQL a la base de datos:

- **mysqli_query("conexión","script"):** Permite ejecutar un script dentro de la conexión definida. Devuelve la consulta dentro de una tabla "Hash" cuyos nombres de columna son los nombres de los campos de la tabla de base de datos.

Gestionar, obtener y procesar los resultados obtenidos después de una ejecución de script:

- **mysqli_fetch_array("vector"):** Permite extraer un registro de un vector bidimensional. Devuelve un arreglo unidimensional.

Cerrar la conexión con el servidor de bases de datos:

- **mysqli_close("conexión"):** Una vez se deje de utilizar una conexión, las buenas prácticas de programación orientan que siempre deben cerrarse para liberar recursos.

6.8. CRUD

Dentro de los procesos que se deben manejar en la programación de aplicaciones orientadas a web, se tiene que muchas de las funcionalidades están orientadas a una base inicial la cual se define dentro de las 4 operaciones básicas que se tienen dentro de las tablas de la base de datos, las cuales dan origen a los procesos CRUD, acrónimo de: Create, Read, Update and Delete (Crear, Leer, Actualizar y Borrar).

En este sentido, se va a adelantar un ejercicio en el cual se va a insertar, listar, actualizar y borrar datos de la tabla "libro", para lo cual se va a requerir dos archivos, un formulario donde se pueda registrar, listar y borrar; y en un archivo adicional denominado "editar.php", se hace la operación en mención.

6.9. Conexión a la base de datos

1) Crear la conexión con la base de datos.

```php
<?php
class Libro{
    private $servidor;
    private $usuario;
    private $clave;
    private $basedato;

    function Libro() {
        $this->servidor = "127.0.0.1";
        $this->usuario  = "root";
        $this->clave    = "";
        $this->basedato = "biblioteca";
    }
```

```php
        $link=mysqli_connect($this->servidor,$this->usuario ,"",$this->basedato);
        mysqli_select_db($link,$this->basedato);
```

2) Creación de formulario para inserción de datos.

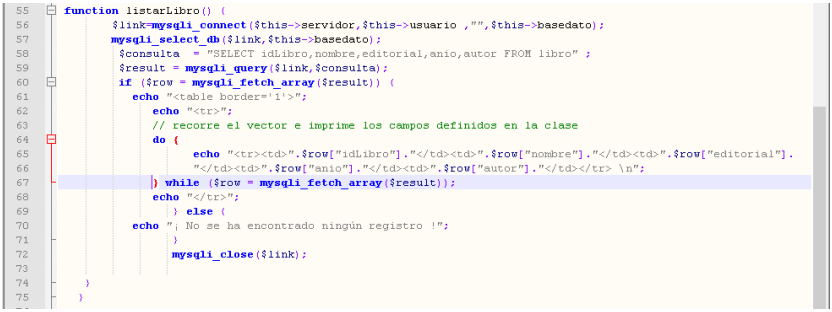
3) Código para insertar en la base de datos desde PHP.

```php
function anexarLibro($nombre,$editorial,$anio, $autor) {
    $link=mysqli_connect($this->servidor,$this->usuario ,"",$this->basedato);
    mysqli_select_db($link,$this->basedato);

    //crear script de inserción con la información dada.
    $grabar_libro = "INSERT INTO libro (nombre, editorial, anio,autor)
    values ('$nombre', '$editorial', '$anio', '$autor')";
    //ejecutar el script en la base de datos
    $guardar_libro=mysqli_query($link,$grabar_libro) or die('El registro de datos fall&oacute;: ' .
    mysqli_connect_error());//valido si guardo
    mysqli_close($link);
}
```

4) Creación de tabla para mostrar la información de la tabla.

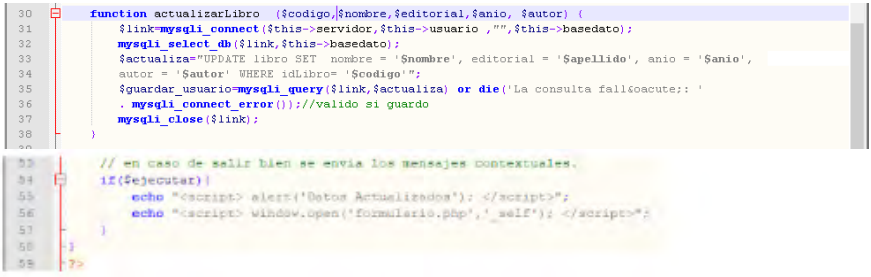
```php
echo "<tr><td>".$row["idLibro"]."</td><td>".$row["nombre"]."</td><td>".$row["editorial"].
"</td><td>".$row["anio"]."</td><td>".$row["autor"]."</td></tr> \n";
```

5) Código para mostrar la consulta a la base de datos, completando la tabla.



```
function listarLibro() {
        $link=mysqli_connect($this->servidor,$this->usuario ,"",$this->basedato);
        mysqli_select_db($link,$this->basedato);
         $consulta  = "SELECT idLibro,nombre,editorial,anio,autor FROM libro" ;
         $result = mysqli_query($link,$consulta);
         if ($row = mysqli_fetch_array($result)) {
           echo "<table border='1'>";
              echo "<tr>";
              // recorre el vector e imprime los campos definidos en la clase
              do {
                    echo "<tr><td>".$row["idLibro"]."</td><td>".$row["nombre"]."</td><td>".$row["editorial"].
                    "</td><td>".$row["anio"]."</td><td>".$row["autor"]."</td></tr> \n";
              } while ($row = mysqli_fetch_array($result));
              echo "</tr>";
                 } else {
           echo "¡ No se ha encontrado ningún registro !";
                 }
                 mysqli_close($link);

    }
  }
```

6) Cuando se quiere actualizar datos, primero captura el id del libro y lo busca en la base de datos, para poder cargar la información.

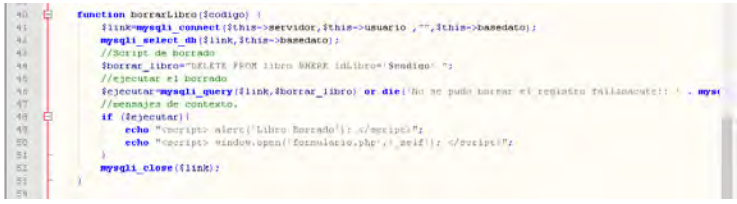


```
    function actualizarLibro  ($codigo,$nombre,$editorial,$anio, $autor) {
        $link=mysqli_connect($this->servidor,$this->usuario ,"",$this->basedato);
        mysqli_select_db($link,$this->basedato);
        $actualiza="UPDATE libro SET  nombre = '$nombre', editorial = '$apellido', anio = '$anio',
        autor = '$autor' WHERE idLibro= '$codigo'";
        $guardar_usuario=mysqli_query($link,$actualiza) or die('La consulta fall&oacute;: '
        . mysqli_connect_error());//valido si guardo
        mysqli_close($link);
    }
```



```
    // en caso de salir bien se envia los mensajes contextuales.
    if($ejecutar){
        echo "<script> alert('Datos Actualizados'); </script>";
        echo "<script> window.open('formulario.php','_self'); </script>";
    }
}
?>
```

7) En el caso de querer borrar, el proceso es más sencillo pues solamente se debe ejecutar la acción de borrado.



```
    function borrarLibro($codigo) {
        $link=mysqli_connect($this->servidor,$this->usuario ,"",$this->basedato);
        mysqli_select_db($link,$this->basedato);
        //Script de borrado
        $borrar_libro="DELETE FROM libro WHERE idLibro='$codigo' ";
        //ejecutar el borrado
        $ejecutar=mysqli_query($link,$borrar_libro) or die('No se pudo borrar el registro fallo&oacute;!: ' . mysq
        //mensajes de contexto.
        if ($ejecutar){
            echo "<script> alert('Libro Borrado'); </script>";
            echo "<script> window.open('formulario.php','_self'); </script>";
        }
        mysqli_close($link);
    }
```

De esta manera, el archivo denominado Libro.php quedaría de la siguiente manera:

```php
<?php
class Libro {
    private $servidor;
    private $usuario;
    private $clave;
    private $basedato;

    function Libro () {
        $this->servidor = "127.0.0.1";
        $this->usuario = "root";
        $this->clave    = "";
        $this->basedato = "biblioteca";
    }

    function anexarLibro($nombre,$editorial,$anio, $autor) {

        $link=mysqli_connect($this->servidor,$this->usuario ,"",$this->
        basedato);
        mysqli_select_db($link,$this->basedato);

        //crear script de inserción con la información dada.
        $grabar_libro = "INSERT INTO libro (nombre, editorial, anio,autor)
         values ('$nombre', '$editorial', '$anio', '$autor')";
        //ejecutar el script en la base de datos
        $guardar_libro=mysqli_query($link,$grabar_libro) or die('El registro de
        datos fall&oacute;: ' . mysqli_connect_error());//valido si guardo
        mysqli_close($link);
    }

    function actualizarLibro  ($nombre,$editorial,$anio, $autor) {
        $link=mysqli_connect($this->servidor,$this->usuario ,"",$this-
>basedato);
        mysqli_select_db($link,$this->basedato);
        $actualiza="UPDATE libro SET  nombre = '$nombre', editorial =
        '$apellido', anio = '$anio', autor = '$autor' WHERE idLibro=
'$codigo'";
        $guardar_usuario=mysqli_query($link,$actualiza) or die('La consulta
        fall&oacute;: ' . mysqli_connect_error());//valido si guardo
        mysqli_close($link);
    }

    function borrarLibro($codigo) {
        $link=mysqli_connect($this->servidor,$this->usuario ,"",$this-
>basedato);
        mysqli_select_db($link,$this->basedato);
        //Script de borrado
        $borrar_libro="DELETE FROM libro WHERE idLibro='$codigo' ";
        //ejecutar el borrado
        $ejecutar=mysqli_query($link,$borrar_libro) or die('No se pudo borrar
e
         registro fall&oacute;: ' . mysqli_error());//valido si guardo
        //mensajes de contexto.
        if ($ejecutar){
            echo "<script> alert('Libro Borrado'); </script>";

            echo "<script> window.open('formulario.php','_self'); </script>";
        }
        mysqli_close($link);
    }


 function listarLibro() {
        $link=mysqli_connect($this->servidor,$this->usuario
,"",$this->basedato);
       mysqli_select_db($link,$this->basedato);
        $consulta  = "SELECT idLibro,nombre,editorial,anio,autor FROM libro" ;
              $result = mysqli_query($link,$consulta);
              if ($row = mysqli_fetch_array($result)) {
                echo "<table border='1'>";
                      echo "<tr>";
               // recorre el vector e imprime los campos definidos en la clase
                      do {
                          echo
                          "<tr><td>".$row["idLibro"]."</td><td>".$row["nombre"
                          ].
                          "</td><td>".$row["editorial"]."</td><td>".$row["anio
                          "].
                          "</td><td>".$row["autor"]."</td></tr> \n";
                      } while ($row = mysqli_fetch_array($result));
                      echo "</tr>";
                 } else {
                echo "¡ No se ha encontrado ningún registro !";
                 }
                 mysqli_close($link);

    }
   }
?>
```

Y el archivo index.html sería:

```
<!DOCTYPE html>
<html lang="es">
	<head>
	<meta charset="utf-8">
	<title>HTML5 Contact Form</title>
	<link  rel = "stylesheet" href = "css/estilos.css" />
</head>

<body>
	<form class="contact_form" action="ControlLibro.php" method="post">
	<ul>
	<li>
	<h2>Formulario datos Libro</h2>
	</li>
	<li>
	<label for="name">Codigo:</label>
	<input type="text" name="codigo" placeholder="123456"  />
	</li>
	<li>
	<label for="name">Nombre:</label>
	<input type="text" name="Nombre" placeholder="Php OO "  />
	</li>

	<li>
	<label for="name">Editorial:</label>
	<input type="text" name="Editorial" placeholder="Pearson" />
	</li>


	<li>
	<label for="name">Annio:</label>
	<input type="text" name="Anio" placeholder="1968"  />
	</li>

	<li>
	<label for="name">Autor:</label>
	<input type="text" name="Autor" placeholder="Miguel Hernandez"  />
	</li>
	<li>
	<button class="submit" type="submit" name="anexar">Enviar </button>
	<button class="submit" type="submit" name="listar">Listar </button>
	<button class="submit" type="submit" name="borrar">Borrar  </button>
	<button class="submit" type="submit" name="limpiar">Limpiar </button>
	</li>
	</ul>
	</form>
</body>
</html>
```

Archivo controlLibro.php es quien toma los datos ingresados del libro y de acuerdo con la opción seleccionada por el usuario (anexar, actualizar, listar, borrar), invocará el método que realiza la acción correspondiente, como se muestra a continuación:

```
<?php
   include("Libro.php");
   class ControlLibro
   {
       private $obj;
         function Libro( ) {
         $obj=new Libro();
       }


       function seleccionarOpcion()
       {
          if(isset($_POST['anexar']))
          {
             $this->registrarDatos(1);
          }
          if(isset($_POST['actualizar']))
          {
             $this->registrarDatos(2);
          }
          if(isset($_POST['listar']))
          {
             $this->registrarDatos(3);
          }
          if(isset($_POST['borrar']))
          {
             $this->registrarDatos(4);
          }
       }

       public function registrarDatos($oper) {
        //captura de la información registrada en variables
        $nombre = $_POST['Nombre'];
         $editorial = $_POST['Editorial'];
        $anio = $_POST['Anio'];
         $autor = $_POST['Autor'];
         $obj=new Libro();
        switch($oper){
               case 1:
                     $obj->anexarLibro($nombre,$editorial,$anio,$autor);
                     break;
               case 2:
       $obj->actualizarLibro($codigo,$nombre,$editorial,$anio,$autor);
              break;
               case 3:
                     $obj->listarLibro( );
              break;
               case 4:
                     $obj->borrarLibro($codigo);
                     break;
              }
              echo "<a href=\"index.html\"> Formulario </a>";
       }
     }
```

```
        $obj1 = new ControlLibro();
        $obj1 -> seleccionarOpcion();
        //echo "<a href=\"index.html\"> Formulario </a>";
?>
```

El archivo estilos.css corresponde al mismo documento utilizado en un anterior ejercicio, y los dos archivos deben guardarse dentro de la carpeta de xampp; a su vez, tiene para publicar información denominada en C:\xampp\htdocs\dashboard. Para efectos de organizar mejor la información, se crea una carpeta denominada "biblioteca" y dentro de ella, los dos archivos (index.html, controlLibro.php y libro.php).

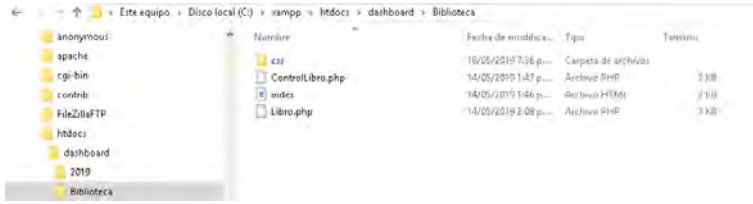


Ilustración 18: Creación carpeta.

Para probar el ejercicio, solo basta abrir un navegador y digitar en la url: **localhost/Biblioteca/index.html.**

Todas las aplicaciones web tienen estas características básicas, lo cual afirma que, a partir de estos desarrollos, se puede adelantar cualquier tipo de software.

Programa que permite realizar las operaciones básicas del CRUD dados en la tabla cliente. Los programas que la conforman son: un index.html, dos archivos de php y un archivo estilos.css.

Código del formulario en html:

Index.html

```
<!DOCTYPE html>
<html lang="es">
<head>
      <meta charset="utf-8">
      <title>HTML5 Contact Form</title>
      <link  rel = "stylesheet" href = "css/estilos.css" />
</head>
<body>
<form class="contact_form" action="Control.php" method="post">
 . . . . . .
</form>
</body>
</html>
```

Archivo CSS:

Estilos.css

```
@CHARSET "ISO-8859-1";
.contact_form h2 {
    background: none repeat scroll 0 0 #F3F3F3;
. . . . . .
.contact_form button {
      margin-left:50px;
        padding:12px;
      width:110px;
}
```

Código de la clase persona.php:

Persona.php

```
<?php
 class Persona {
    private $codigo;
    private $nombre;
    private $apellido;
    private $email;
    private $telefono;
    private $mensaje;

     function __construct() {
        $this->codigo = "";
        $this->nombre = "";
        $this->apellido = "";
       $this->email = "";
       $this->telefono = "";
       $this->mensaje = "";
    }

    function Persona($codigo,$nombre, $apellido, $email, $telefono, $mensaje)
{
       $this->codigo = $codigo;
          $this->nombre = $nombre;
          $this->apellido = $apellido;
       $this->email = $email;
       $this->telefono = $telefono;
       $this->mensaje = $mensaje;
    }

      function getCodigo() {
        return $this->codigo;
    }

    function setCodigo($codigo) {
       $this->codigo = $codigo;
    }

    function getNombre() {
        return $this->nombre;
    }

    function setNombre($nombre) {
       $this->nombre = $nombre;
    }

      function getApellido() {
        return $this->apellido;
    }

    function setApellido($apellido) {
       $this->apellido = $apellido;
    }


    function getEmail() {
```

```
        return $this->email;
    }

    function setEmail($email) {
       $this->email = $email;
    }

    function getTelefono() {
        return $this->telefono;
    }

    function setTelefono($telefono) {
       $this->telefono = $telefono;
    }

    function getMensaje() {
        return $this->mensaje;
    }

    function setMensaje($mensaje) {
       $this->mensaje = $mensaje;
    }
}
?>
```

Código de la clase control:

Control.php

```
<?php
       include("Cliente.php");
     class Control  {
            private $obj;

             function __construct( ) {
             $obj=new Cliente();
         }

             function seleccionarOpcion()
             {

                   if(isset($_POST['anexar']))
                   {
                       $this->registrarDatos(1);
                   }

                   if(isset($_POST['actualizar']))
                   {
                       $this->registrarDatos(2);
                   }

                   if(isset($_POST['listar']))
                   {
                       $this->registrarDatos(3);
                   }
                   if(isset($_POST['borrar']))
                   {
                       $this->registrarDatos(4);
                   }
             }


            public function registrarDatos($oper) {
                   $codigo=$_POST['codigo'];
                   $nombre=$_POST['nombre'];
                   $apellido=$_POST['apellido'];
                   $email=$_POST['email'];
                   $telefono=$_POST['telefono'];
                   $mensaje=$_POST['mensaje'];
                   $obj=new Cliente();
                   switch($oper){
                    case 1:
                          $obj-
>anexarCliente($codigo,$nombre,$apellido,$email,$telefono,$mensaje);
                          break;
                    case 2:
                          $obj-
>actualizarCliente($codigo,$nombre,$apellido,$email,$telefono,$mensaje);
                          break;
                    case 3:
                          $obj->listarCliente( );
                          break;
                    case 4:
```

```
    }
    $obj1 = new Control();
       $obj1 -> seleccionarOpcion();
    //echo "<a href=\"index.html\"> Formulario </a>";
?>
```

Código de la clase que implementa el CRUD:

Cliente.php

```
<?php
 class Cliente{
       private $servidor;
       private $usuario;
       private $clave;
       private $basedato;

       function Cliente() {
              $this->servidor = "127.0.0.1";
              $this->usuario  = "root";
              $this->clave    = "";
              $this->basedato = "empresa";
       }

       function anexarCliente($codigo,$nombre,$apellido,$email,
       $telefono,$mensaje) {
              $link=mysqli_connect($this->servidor,$this->usuario ,"",$this-
              >basedato);
              mysqli_select_db($link,$this->basedato);
                           $grabar_cliente="INSERT INTO
              Clientes(codcli,nomcli,apecli,telcli,emailcli,mensajecli)
              VALUES('$codigo','$nombre','$apellido','$telefono','$email','$men
              saje')";//guardo los datos
              $guardar_usuario=mysqli_query($link,$grabar_cliente) or die('El
              regstro de datos fall&oacute;: ' .
              mysqli_connect_error());//valido si guardo
              mysqli_close($link);
       }

      function actualizarCliente
      ($codigo,$nombre,$apellido,$email,$telefono,$mensaje) {
              $link=mysqli_connect($this->servidor,$this->usuario ,"",$this-
              >basedato);
              mysqli_select_db($link,$this->basedato);
        $actualiza="UPDATE Clientes SET  nomcli = '$nombre', apecli =
       '$apellido', telcli = '$telefono', emailcli = '$email', mensajecli =
       '$mensaje' WHERE codcli= '$codigo'";
              $guardar_usuario=mysqli_query($link,$actualiza) or die('La
        consulta fall&oacute;: ' . mysqli_connect_error());//valido si guardo
              mysqli_close($link);
       }

     function borrarCliente($codigo) {
              $link=mysqli_connect($this->servidor,$this->usuario ,"",$this-
       >basedato);
              mysqli_select_db($link,$this->basedato);
        $borrar_cliente="DELETE FROM Clientes WHERE codcli='$codigo' ";
              $borrar=mysqli_query($link,$borrar_cliente) or die('No se pudo
       borrar el registro fall&oacute;: ' . mysqli_error());//valido si guardo
              mysqli_close($link);
       }


     function listarCliente() {
```

```
            $link=mysqli_connect($this->servidor,$this->usuario
      ,"",$this->basedato);
      mysqli_select_db($link,$this->basedato);
        $consulta  = "SELECT codcli,nomcli,emailcli,telcli FROM Clientes" ;
       $result = mysqli_query($link,$consulta);
       if ($row = mysqli_fetch_array($result)) {
         echo "<table border='1'>";
               echo "<tr>";
               // recorre el vector e imprime los campos definidos en la clase
               do {
                    echo
                    "<tr><td>".$row["codcli"]."</td><td>".$row["nomcli"]."</td>
                    <td>".$row["emailcli"]."</td><td>".$row["telcli"]."</td><td
                    >".$row["emailcli"]."</td></tr> \n";
                      } while ($row = mysqli_fetch_array($result));
                      echo "</tr>";
                } else {
               echo "¡ No se ha encontrado ningún registro !";
                }
                mysqli_close($link);

                //$this->free_result($resultado); // libera de memoria la
tabla
                   de resultados
    } // fin de la consulta
   }

?>
```

Ejercicio que implementa el código de acceso a una aplicación. Ingreso de credenciales válidas al formulario.

ingreso.html

```
<!DOCTYPE html>
<html lang="es">
<head>
      <meta charset="utf-8">
      <title>HTML5 Contact Form</title>
      <link  rel = "stylesheet" href = "css/estilos.css" />
</head>
<body>

<form class="contact_form" action="Controlador.php" method="post">
    <ul>
        <li>
             <h2>LOGIN</h2>
        </li>
       <li>
            <label for="name">Usuario:</label>
            <input type="text" name="usuario" required/>
        </li>


        <li>
            <label for="name">Clave:</label>
            <input type="password" name="password" required/>
        </li>

         <button class="submit" type="submit" name="aceptar">Aceptar </button>
       <button class="submit" type="submit" name="borrar">Borrar </button>
        </li>
    </ul>
</form>
</body>
</html>
```

Código del controlador U PHP

ControladorU.php

```php
<?php
include("Usuario.php");

    class ControladorU  {
          private $obj;

           function __construct( ) {
           $obj=new LUsuario();
       }

           function seleccionarOpcion()
           {

                if(isset($_POST['aceptar']))
                {
                    $this->registrarDatos(1);
                }

           }

          public function registrarDatos($oper) {
                $usuario=$_POST['usuario'];
                $password=$_POST['password'];
                $obj=new LUsuario();
                switch($oper){
                 case 1:
                      $obj->consultarUsuario($usuario,$password);
                      break;
                       }
          }

   }
   $obj1 = new Controlador();
   $obj1 -> seleccionarOpcion();
   //echo "<a href=\"index.html\"> Formulario </a>";
?>
```

Usuario.php

```php
<?php
 class Usuario{
      private $servidor;
      private $usuario;
      private $clave;
      private $basedato;

      function __construct() {
            $this->servidor = "127.0.0.1:3306";
            $this->usuario  = "root";
                $this->clave    = "";
            $this->basedato = "empresa";
      }
        function consultarUsuario($usuario,$password){
              $link=mysqli_connect($this->servidor,$this->usuario
              ,"",$this->basedato);
            mysqli_select_db($link,$this->basedato);
                $consulta  = "SELECT LoginUsu,PassUsu FROM usuario
                WHERE LoginUsu = '$usuario' AND PassUsu = '$password'";
                $result = mysqli_query($link,$consulta);
                if ($row = mysqli_fetch_array($result) ) {
                   header('Location: index2.html');
                }else {
                    echo'<script>
                          alert("Acceso Denegado,Credenciales
                          erradas");
                             window.location.href="index.php";
                        </script>';

                }

            mysqli_close($link);
      }
     }
?>
```

En caso de que las credenciales "login" y "password" sean correctas:

Index.php

```
<!DOCTYPE html>
<html lang="es">
<head>
      <meta charset="utf-8">
      <title>HTML5 Contact Form</title>
      <link  rel = "stylesheet" href = "css/estilos.css" />
</head>
<body>
<form class="contact_form" action="Control.php" method="post">
    <ul>
        <li>
             <h2>Formulario de datos personales</h2>
        </li>
            <li>
            <label for="name">Codigo:</label>
            <input type="text" name="codigo" placeholder="123456" />
        </li>


        <li>
            <label for="name">Nombre:</label>
            <input type="text" name="nombre" placeholder="Julieth"/>
        </li>

            <li>
            <label for="name">Apellido:</label>
            <input type="text" name="apellido" placeholder="Hernandez" />
        </li>

        <li>
            <label for="email">Email:</label>
            <input type="email" name="email"
placeholder="julieth@ejemplo.com.co" />
        </li>
        <li>
            <label for="telefono">Telefono:</label>
            <input type="tel" name="telefono" placeholder="2916520" />
        </li>
        <li>
            <label for="Mensaje">Mensaje:</label>
            <textarea name="mensaje" cols="40" rows="6" / ></textarea>
        </li>
        <li>

         <button class="submit" type="submit" name="anexar">Enviar
</button>
             <button class="submit" type="submit"
name="actualizar">Actualizar </button>
          <button class="submit" type="submit" name="listar">Listar
</button>
```

```
            <button class="submit" type="submit" name="borrar">Borrar
</button>
            <button class="submit" type="submit" name="limpiar">Limpiar
</button>

        </li>
    </ul>
</form>
</body>
</html>
```

La figura figura *-19-* presenta el ingreso de las credenciales (usuario y clave) de un usuario para acceder al sistema.

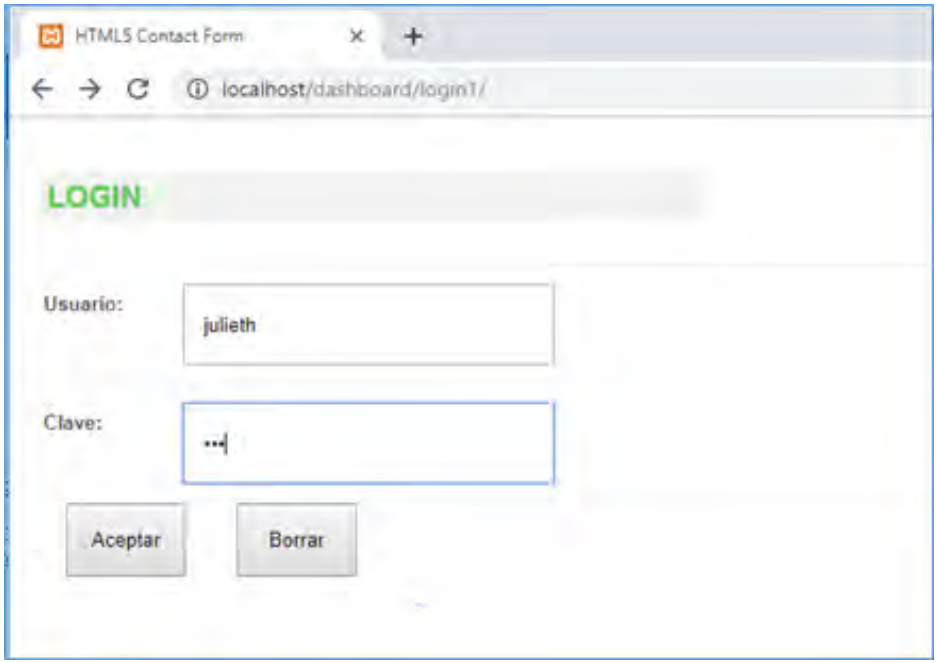


Ilustración 19: Usuario y clave.

En el caso que las credenciales sean válidas, se tendrá el acceso al formulario de datos personales, como se visualiza en la siguiente imagen:

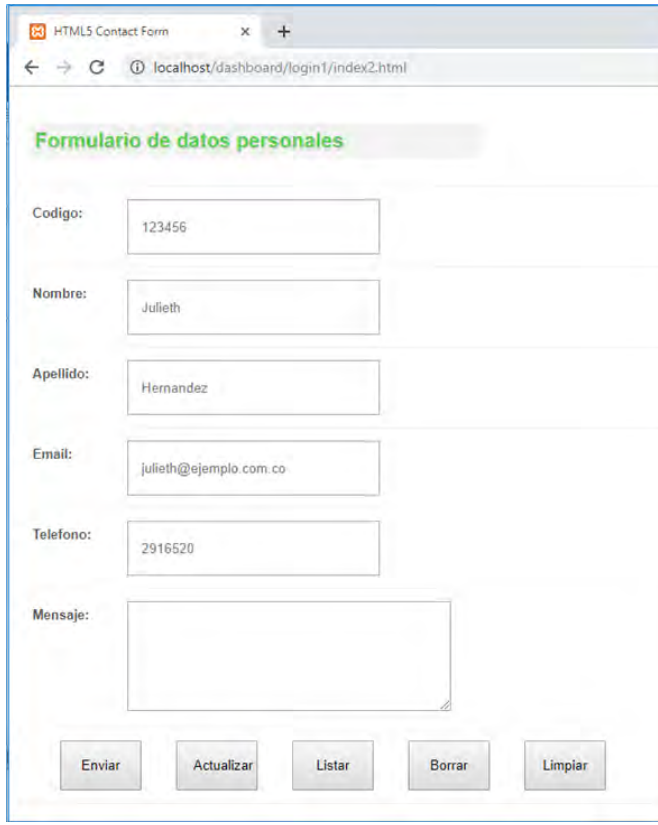


Ilustración 20: Formulario de datos personales.

En caso de ingresar las credenciales erradas, se negará el acceso, como se presenta en las figuras *-21- y -22-*.

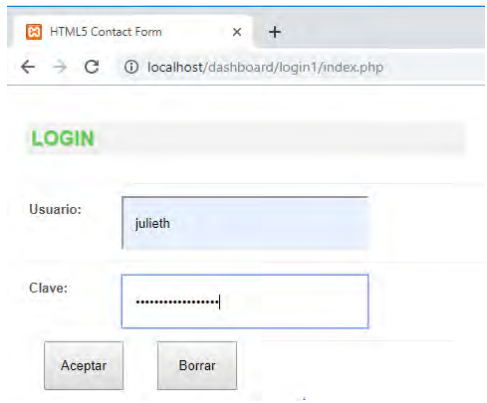


Ilustración 21: Ingreso credenciales erradas.

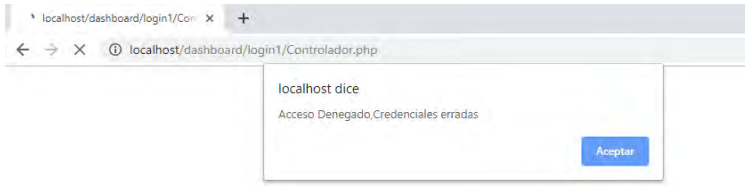


Ilustración 22: Mensaje credenciales erradas.

Se plantea que al ejercicio anterior se anexen los módulos de un captcha y la creación de un nuevo usuario para el caso que este no se encuentre registrado en la base de datos.

6.10. Ejercicio

1) Crear una base de datos que se llame ejercicio.php

Dentro de dicha base de datos, crear una tabla que se llame "artículos", la cual debe tener los siguientes campos: id articulo (identificador único del artículo, el cual debe ser autonumérico – debe ser llave primaria), descripcionarticulo (nombre del artículo), precio unitario (valor en pesos que debe tener el artículo). Otra tabla que se llame "factura", la cual va a tener los siguientes campos: id factura (autonumérico que define el número de la factura – debe ser llave primaria), id articulo (es llave foránea de la tabla articulo), fecha compra (fecha de compra del artículo), cantidad (cantidad de artículos comprados), nombrecomprador y telefonocomprador.

Crear una aplicación que permita registrar facturas con el formato de factura del ejercicio del capítulo 3.

Mostrar una factura específica (tener en cuenta que no es necesario guardar los datos de IVA, totales y subtotales, pues esto debe estar cargado a partir de la información de cada factura).

2) Adelantar un ejercicio en el cual se pueda cargar un archivo (upload) y guardar la ubicación física dentro de una tabla llamada archivos.

6.11. Preguntas revisión de conceptos

1) Coloque los números en orden sobre las acciones que se deben adelantar al momento de realizar una conexión con bases de datos.

Acción	No
Definir base de datos	
Crear la consulta	
Crear conexión base de datos	
Ejecutar consulta	

Tabla 19: Ejercicio revisión de conceptos.

2) El comando para actualizar un dato dentro de la base de datos con SQL es:
a) update
b) insert
c) actualize
d) drop

3) Las etiquetas para trabajar código PHP son:
a) <php> y </php>
b) <?php y php?>
c) <$php y php$>
d) <?php y ? >

4) La forma para definir una variable llamada figura, de tipo entero en PHP es:
a) var int figura;
b) int figura;
c) var figura;
d) $figura;

5) El comando if dentro de PHP sirve para:
a) Definir una condición.
b) Mostrar por pantalla un mensaje.
c) Definir una variable.Zakas Nicholas C. (2006) JavaScript para desarrolladores Web. Anaya Multimedia.
d) Mostrar un registro dentro de una tabla.

Sitios Web de interés

http://www.w3.org/TR/css3-transforms/
http://www.w3schools.com/css/default.asp
http://www.w3.org/People/Bos/
http://www.w3.org/TR/2009/PR-css3-selectors-20091215/selectors
http://www.w3schools.com/css/default.asp
http://www.w3.org/People/Bos/
http://www.w3schools.com/html/html5_intro.asp
http://www.w3.org/TR/2011/WD-html5-20110525/
http://www.w3schools.com/html/html5_intro.asp
http://www.w3.org/TR/2011/WD-html5-20110525/
http://www.w3schools.com/js/default.asp
http://www.w3.org/standards/webdesign/script.html

Bibliografía