

Aplicaciones web con **PHP**

Héctor Flórez Fernández
Jorge Hernández Rodríguez

edü

Aplicaciones web con P H P

Héctor Flórez Fernández
Jorge Hernández Rodríguez

edü[®]

Conocimiento a su alcance

BOGOTÁ - MÉXICO, D.F.

Flórez Fernández, Héctor, *et al.*

Aplicaciones web con PHP / Héctor Flórez Fernández y Jorge Hernández Rodríguez --
Bogotá : Ediciones de la U, 2021

293 p. ; 24 cm.

ISBN 978-958-792-234-9

e-ISBN 978-958-792-235-6

1. Informática 2. Programación 3. Bases de datos I. Tít.

657 cd

Área: Informática

Primera edición: Bogotá, Colombia, marzo de 2021

ISBN. 978-958-792-234-9

- © Héctor Flórez Fernández
Profesor titular en la Universidad Distrital Francisco José de Caldas, Bogotá, Colombia
Email: haflorenz@udistrital.edu.co
- © Jorge Hernández Rodríguez
Email: hrjorge@correo.udistrital.edu.co
- © Ediciones de la U - Carrera 27 # 27-43 - Tel. (+57-1) 3203510 - 3203499
www.edicionesdelau.com - E-mail: editor@edicionesdelau.com
Bogotá, Colombia

Ediciones de la U es una empresa editorial que, con una visión moderna y estratégica de las tecnologías, desarrolla, promueve, distribuye y comercializa contenidos, herramientas de formación, libros técnicos y profesionales, e-books, e-learning o aprendizaje en línea, realizados por autores con amplia experiencia en las diferentes áreas profesionales e investigativas, para brindar a nuestros usuarios soluciones útiles y prácticas que contribuyan al dominio de sus campos de trabajo y a su mejor desempeño en un mundo global, cambiante y cada vez más competitivo.

Coordinación editorial: Adriana Gutiérrez M.

Carátula: Ediciones de la U

Impresión: DGP Editores SAS

Calle 63 No. 70 D - 34, Pbx. (571) 7217756

Impreso y hecho en Colombia

Printed and made in Colombia

No está permitida la reproducción total o parcial de este libro, ni su tratamiento informático, ni la transmisión de ninguna forma o por cualquier medio, ya sea electrónico, mecánico, por fotocopia, por registro y otros medios, sin el permiso previo y por escrito de los titulares del Copyright.

Tabla de Contenido

Prólogo	xiv
1 Introducción	1
1.1 HTML	1
1.2 CSS	2
1.3 JavaScript	2
1.4 DOM	3
1.5 PHP	3
1.6 jQuery	4
1.7 Bootstrap	5
1.8 AJAX	5
1.9 REST	6
1.10 Patrón Modelo Vista Controlador	6
2 Conceptos Básicos de Programación	9
2.1 Tipos de datos	9
2.1.1 Variables	9
2.1.2 Variables constantes	10
2.1.3 Tipos primitivos de datos	10
2.1.4 Ámbito de las variables	11
2.1.5 Variables superglobales	11
2.2 Operadores	12
2.2.1 Operadores aritméticos	12
2.2.2 Operadores de asignación	12
2.2.3 Operadores lógicos	13
2.2.4 Operadores de comparación	13
2.2.5 Operadores a nivel de bits	13
2.3 Estructuras de programación	14
2.3.1 Sentencias	14
2.3.2 Comentarios	14
2.3.3 Sentencia de impresión	15
2.3.4 Estructura de condición if	15
2.3.5 Estructura de condición if else	15

2.3.6	Estructura de condición <code>if else if</code>	16
2.3.7	Operador ternario	16
2.3.8	Estructura de condición <code>switch case</code>	17
2.3.9	Estructura de repetición <code>while</code>	18
2.3.10	Estructura de repetición <code>for</code>	18
2.3.11	Estructura de repetición <code>do while</code>	18
2.3.12	Estructura de repetición <code>foreach</code>	19
2.4	Secuencias de escape	19
3	HTML, CSS y JavaScript	21
3.1	HTML	21
3.1.1	Estructura de HTML	21
3.1.2	Etiquetas básicas de HTML	22
3.1.3	Formularios	30
3.2	CSS	40
3.3	JavaScript	41
4	Introducción a PHP	45
4.1	Compilación en PHP	45
4.2	Editores para PHP	47
4.3	Inclusión de archivos	47
5	Conceptos Básicos de Programación Orientada a Objetos	49
5.1	Clase	49
5.1.1	Visibilidad	49
5.1.2	Atributos	50
5.1.3	Métodos	50
5.1.4	Encapsulamiento	51
5.1.5	Apuntador <code>\$this</code>	51
5.1.6	Operador <code>self::</code>	51
5.2	Objeto	51
5.3	Clasificación de métodos	52
5.4	Sobrecarga de métodos	53
5.5	Recursividad	54
5.6	Bajo acoplamiento	56
5.7	Alta cohesión	56
5.8	Arquitectura de software	56
6	Arreglos, Matrices y Colecciones	59
6.1	Arreglos	59
6.1.1	Búsqueda lineal	60
6.1.2	Búsqueda binaria	61

6.2	Matrices	61
6.2.1	Cálculo de la traspuesta de una matriz	62
6.2.2	Multiplicación de matrices	62
6.3	Colecciones	63
6.4	Funciones de ordenamiento	64
6.5	Algoritmos de ordenamiento	67
6.5.1	Burbuja (Bubble Sort)	67
6.5.2	Inserción (Insertion Sort)	69
6.5.3	Mezcla (Merge Sort)	69
6.5.4	Clasificación rápida (Quick Sort)	70
6.6	Arreglos multidimensionales	71
7	Bootstrap	75
7.1	Disposición de pantalla	76
7.2	Tablas	78
7.3	Alertas	80
7.4	Contenedores de contenido	82
7.5	Formularios	83
7.6	Barras de menú	85
8	jQuery	89
8.1	Inclusión de jQuery en un proyecto	89
8.2	Selectores de objetos	91
8.3	Tipos de selectores	92
8.4	Manipulación de objetos del DOM	94
8.5	Escuchador de eventos	95
8.6	Animaciones	97
8.7	Aplicación de jQuery	98
9	JSON	103
9.1	Estructura de un documento JSON	104
9.2	Tipos de datos	105
9.3	Uso de JSON en JavaScript	107
9.3.1	Codificación y decodificación	108
9.4	Uso de JSON en PHP	110
9.4.1	Función <code>json_decode</code>	110
9.4.2	Función <code>json_encode</code>	111
10	Acceso a Bases de Datos	113
10.1	Conexión a base de datos	113
10.2	Data Access Object (DAO)	115
10.3	Aplicación Web en PHP con acceso a base de datos	122

10.3.1	Inserción de datos	123
10.3.2	Consulta de datos	126
10.3.3	Actualización de datos	127
10.3.4	Eliminación de datos	132
11	Manejo de Sesiones	137
11.1	Clases requeridas para autenticación	138
11.2	Formulario de autenticación	142
11.3	Página de sesión	147
11.4	Cierre de sesión	153
12	Generación de Documentos PDF	155
12.1	EZPDF	155
12.2	Generación de documento PDF	161
13	Gráficas	175
13.1	Chartkick	175
13.1.1	<i>Line Chart</i>	176
13.1.2	<i>Area Chart</i>	178
13.1.3	<i>Column Chart</i>	179
13.1.4	<i>Bar Chart</i>	181
13.1.5	<i>Pie Chart</i>	182
13.1.6	<i>Donut Chart</i>	182
13.1.7	<i>Geo Chart</i>	183
13.2	Data-Driven Documents (D3)	184
13.2.1	<i>Tidy Tree</i>	184
13.2.2	<i>Bubble Chart</i>	188
13.3	Aplicación Web en PHP con despliegue de gráficas	191
13.3.1	Charts con Chartkick para la aplicación del caso de estudio	196
13.3.2	Chart con D3 para la aplicación del caso de estudio	203
14	Envío de Correo Electrónico	211
14.1	Función <code>mail</code>	211
14.2	Formulario de contacto	213
15	AJAX	217
15.1	HTTP	218
15.2	Uso de AJAX	220
15.3	Objeto XMLHttpRequest	221
15.4	Implementación de AJAX con jQuery	222
15.5	Fetch	223
15.6	Aplicación Web en PHP usando AJAX	223

16 Servicios REST	231
16.1 Modelo Cliente - Servidor	231
16.2 Representational State Transfer (REST)	232
16.3 Especificaciones de un sistema REST	233
16.4 Operaciones en servicios REST	234
16.5 Autenticación de solicitudes	235
16.6 PHP como cliente	237
16.7 Servicios REST con PHP	244
16.7.1 Cliente	244
16.7.2 Servidor	251
17 Patrón Modelo Vista Controlador	259
17.1 Flujo en MVC con enfoque Web	260
17.2 Modelo	261
17.3 Vista	261
17.4 Controlador	261
17.5 MVC en aplicaciones Web con PHP	262
Glosario de Términos	275
Bibliografía	281

Lista de Figuras

3.1	Encabezado en HTML	23
3.2	Texto en HTML	23
3.3	Texto con formato en HTML	24
3.4	Imagen en HTML	25
3.5	Listas en HTML	28
3.6	Tabla en HTML	29
3.7	Bloques en HTML	31
3.8	Paleta de colores en formulario HTML	32
3.9	Cuadro de fecha en formulario HTML	33
3.10	Cuadro de fecha y hora en formulario HTML	33
3.11	Cuadro de correo electrónico en formulario HTML	34
3.12	Cuadro de archivo en formulario HTML	35
3.13	Cuadro de mes y año en formulario HTML	36
3.14	Cuadro de hora en formulario HTML	37
3.15	Cuadro de semana y año en formulario HTML	37
3.16	Listas de selección en formulario HTML	39
3.17	Uso de hoja de estilo CSS en HTML	41
3.18	Factorial con JavaScript en HTML	43
3.19	Dibujo con JavaScript en HTML	44
6.1	Multiplicación de matrices	63
6.2	Ordenamiento burbuja	68
6.3	Resultado de recorrido de un arreglo multidimensional	73
7.1	Contenedores con Bootstrap	78
7.2	Tabla con Bootstrap	79
7.3	Alerta con título con Bootstrap	81
7.4	Alerta con botón de cierre con Bootstrap	82
7.5	Contenedor de contenido con Bootstrap	83
7.6	Formulario con Bootstrap	85
7.7	Barra de menú con Bootstrap	87
8.1	Ejemplo de jQuery	99

8.2 Estructura de archivos de proyecto jQuery	99
8.3 Interacción del usuario	101
9.1 Estructura JSON	105
10.1 Estructura de archivos para implementar DAO	116
10.2 Estructura de archivos de aplicación usando DAO	122
10.3 Formulario de inserción	125
10.4 Resultado de inserción	126
10.5 Resultado de consulta	127
10.6 Segunda versión de resultado de consulta	129
10.7 Formulario de actualización	131
10.8 Resultado de actualización	132
10.9 Confirmación de eliminación	135
10.10 Resultado de eliminación	135
11.1 Estructura de archivos para manejo de sesiones	139
11.2 Estructura de archivos para autenticación	142
11.3 Formulario de autenticación	146
11.4 Formulario de autenticación con error de correo o clave	146
11.5 Formulario de autenticación con usuario deshabilitado	147
11.6 Página de sesión	148
11.7 Página de sesión <i>Responsive</i> en dispositivos móviles	152
12.1 Tabla en documento PDF	158
12.2 Tabla con títulos de columnas en documento PDF	159
12.3 Tabla con títulos de columnas y opciones en documento PDF	160
12.4 Diagrama de clases	161
12.5 Modelo relacional	162
12.6 Estructura de archivos para generación de PDF	164
12.7 Factura en documento PDF	174
13.1 <i>Line Chart</i>	177
13.2 <i>Line Chart</i> Múltiple	177
13.3 <i>Area Chart</i>	178
13.4 <i>Area Chart</i> múltiple	179
13.5 <i>Column Chart</i>	179
13.6 <i>Column Chart</i> múltiple	180
13.7 <i>Column Chart</i> múltiple apilado	181
13.8 <i>Bar Chart</i>	181
13.9 <i>Pie Chart</i>	182
13.10 <i>Donut Chart</i>	183
13.11 <i>Geo Chart</i>	183

13.12 <i>Tidy Tree</i>	187
13.13 <i>Bubble Chart</i>	191
13.14 Diagrama de clases	192
13.15 Modelo relacional	193
13.16 Estructura de archivos para generación de gráficas	196
13.17 <i>Column Chart</i> de unidades vendidas por producto	202
13.18 <i>Column Chart</i> de unidades vendidas por producto por genero	202
13.19 <i>Pie Chart</i> de ventas por genero	202
13.20 <i>Tidy Tree</i> de productos	208
14.1 Formulario de contacto	215
14.2 Respuesta del formulario de contacto	216
14.3 Correo electrónico enviado	216
15.1 Cambio del DOM con el concepto AJAX	218
15.2 Estructura de archivos de aplicación AJAX	224
15.3 Despliegue de la aplicación	225
16.1 Modelo cliente - servidor	232
16.2 Formulario para consultar tweets	239
16.3 Resultado de búsqueda en API Twitter	242
16.4 Estructura de archivos del cliente	244
16.5 Interfaz gráfica del cliente	247
16.6 Estructura de archivos del servidor	251
17.1 Flujo en MVC	260
17.2 Estructura de archivos de aplicación MVC con enfoque web	262
17.3 Vista para crear	271
17.4 Vista para actualizar	271
17.5 Vista para consultar	273

Lista de Tablas

2.1	Tipos primitivos de datos en PHP	10
2.2	Variables superglobales	12
2.3	Operadores aritméticos	12
2.4	Operadores de asignación	13
2.5	Operadores lógicos	13
2.6	Operadores de comparación	14
2.7	Operadores a nivel de bits	14
2.8	Secuencias de escape	19
6.1	Arreglo multidimensional	71
7.1	Contenedores	77
8.1	Tipos de selectores por su familia	93
8.2	Principales métodos para manipular el DOM	95
8.3	Principales eventos de jQuery	96
15.1	Principales propiedades del objeto de configuración de la función \$.ajax()	223
16.1	Métodos HTTP en servicios REST	235

Prólogo

El libro *Aplicaciones Web con PHP* ofrece al lector una exposición clara y suficiente de los conceptos básicos de PHP mediante el paradigma orientación a objetos así como los conceptos de AJAX mediante la librería jQuery. Esta exposición se realiza mediante el desarrollo de aplicaciones que combina PHP, HTML, JavaScript mediante jQuery y CSS mediante Bootstrap.

El libro expone con una gran cantidad de ejemplos y demostraciones de las diferentes características de PHP basado en el paradigma de orientación a objetos, además de orientar el desarrollo mediante arquitecturas, patrones y buenas prácticas en el desarrollo de aplicaciones.

Se ofrecen explicaciones de conceptos básicos de programación, conceptos de programación orientada a objetos, desarrollo orientado a objetos con base en arquitectura de tres capas, acceso a repositorios de datos mediante bases de datos, generación de archivos PDF, conceptos básicos de AJAX, despliegue de información mediante gráficos obtenidos mediante servicios ofrecidos por plug ins, servicios REST, patron MVC, entre otros.

1

Introducción

1.1 HTML

HTML es el acrónimo de Hypertext Markup Language, que es el lenguaje estándar para desplegar documentos en un navegador web. Los navegadores web reciben documentos HTML de un servidor web o del almacenamiento local. Los documentos almacenados por un servidor web pueden ser accedidos mediante un dominio, es decir, estos documentos conforman un sitio web en producción. Los documentos locales pueden ser accedidos mediante un servidor de aplicaciones local con dominio estándar *localhost*, es decir, estos documentos conforman un sitio web en desarrollo.

HTML describe la estructura de una página web. Los elementos HTML son los componentes básicos de las páginas HTML. En la página web, se puede incrustar diferentes componentes HTML, tales como imágenes y formularios. Además, HTML proporciona un medio para crear documentos estructurados para texto como encabezados, párrafos, listas, enlaces, citas etc. Los elementos HTML están descritos por etiquetas que se encierran por los símbolos: `<` y `>`. Algunas etiquetas incrustan información y requieren el símbolo `/` antes del símbolo `>` (por ejemplo `<img />`) mientras que otras encierran el contenido entre dos etiquetas (por ejemplo `<p></p>`) [12].

HTML puede incrustar programas escritos en un lenguaje de secuencias de comandos como JavaScript, lo que afecta el comportamiento y el contenido de las páginas web. La inclusión de CSS define el aspecto y el diseño del contenido. El World Wide Web Consortium¹ (W3C) fue la organización que inicialmente mantuvo HTML y mantiene actualmente de los estándares CSS desde 1997.

¹<https://www.w3.org/>

HTML ha evolucionado con varias versiones actualizadas. Cada versión ha permitido crear páginas web de una manera mucho más fácil, estética y eficiente. HTML 1.0 se lanzó en 1993 con la intención de compartir información que pueda ser legible y accesible a través de navegadores web. HTML 2.0 fue publicado en 1995 y contiene todas las características de HTML 1.0 junto con algunas características adicionales que permaneció como el lenguaje de marcado estándar para diseñar y crear sitios web hasta enero de 1997. HTML 3.0 incluyó nuevas características mejoradas de HTML brindando características más potentes para el diseño de páginas web. Sin embargo, estas nuevas características ralentizaron el navegador al aplicar mejoras adicionales. HTML 4.01 es ampliamente utilizado y era una versión exitosa de HTML antes de HTML 5.0, el cual actualmente se usa en todo el mundo. Se puede decir que HTML 5 es una versión extendida de HTML 4.01 que se publicó en el año 2012.

1.2 CSS

Cascading Style Sheets (CSS) es un lenguaje de hojas de estilo utilizado para describir la presentación de un documento escrito en HTML. CSS está diseñado para permitir la separación del contenido, colores y fuentes en una página web. Esta separación puede mejorar la accesibilidad al contenido, proporcionar más flexibilidad y control en la especificación de las características de presentación. Además, permitir que múltiples páginas web compartan el formato al especificar el CSS relevante en un archivo con extensión `css` y reducir la complejidad y la repetición en el contenido estructural CSS [13]. La W3C mantiene y proporciona las especificaciones de CSS.

1.3 JavaScript

JavaScript es un lenguaje de secuencias de comandos interpretado. Junto con HTML y CSS, JavaScript es una de las tecnologías principales de la World Wide Web Foundation². JavaScript habilita páginas web interactivas y es una parte esencial de las aplicaciones web. Los navegadores web tienen un motor de JavaScript dedicado para ejecutarlo. JavaScript admite estilos de programación impulsados por eventos, funcionales e imperativos. Tiene API para trabajar con texto, matrices, fechas, expresiones regulares y DOM [8].

Inicialmente solo se implementó en el lado del cliente en los navegadores web, sin embargo, los motores de JavaScript ahora están integrados en muchos otros tipos de software incluido el lado del servidor en servidores web, y/o bases de datos. JavaScript surgió gracias a la influencia de lenguajes de programación como Self y Scheme. Además, el formato de texto JSON, el cual es utilizado para almacenar estructuras de datos en archivos o transmitirlos a través de redes, se basa en JavaScript.

²<https://webfoundation.org/>

1.4 DOM

DOM es el acrónimo de Document Object Model. DOM es un modelo de objetos de documento que proporciona las especificaciones para representar documentos HTML y XML. Es un modelo estándar de cómo se pueden combinar estos objetos y una interfaz estándar para procesarlos y manipularlos [20].

DOM es una forma de representar objetos del documento HTML. El DOM es un árbol de nodos organizados por una jerarquía. Cada nodo es un objeto con unas propiedades y métodos. El objetivo del DOM es definir cómo los lenguajes de programación pueden manipular los objetos de manera dinámica. El lenguaje de programación más usado para manipular el DOM es JavaScript.

La W3C (World Wide Web Consortium), especifica como debe ser el DOM. Un objeto puede ser por ejemplo una tabla incluida con la sintaxis: `<table></table>`, en la cual se definen algunas propiedades como el espacio entre celdas y su borde mediante el atributo `cellpadding` y otros objetos dentro de la tabla que en este caso representan las filas y columnas `<tr>`, `<td>`.

1.5 PHP

PHP³ es un lenguaje de programación que permite incorporar HTML, el cual se usa principalmente para aplicaciones web dinámicas. De esta manera, PHP puede intercalarse con HTML, lo que simplifica la construcción de páginas web. PHP es un lenguaje que se interpreta en un explorador mediante Apache, el cual actúa como servidor de aplicaciones. Entonces, PHP no es un lenguaje que se compila y genera archivos ejecutables independientes. La sintaxis de PHP es conocida debido a que toma la mayor parte de C, Java y Perl. PHP es un lenguaje de código abierto y se ejecuta en la mayoría de los sistemas operativos y con la mayoría de los servidores web [16].

PHP fue escrito en el lenguaje de programación C por Rasmus Lerdorf en 1994 para ser usado en el monitoreo de su currículum en línea e información personal. Por esta razón, originalmente PHP era el acrónimo de “Personal Home Page”. Lerdorf combinó PHP con su propio intérprete de formularios (generalmente conocido como PHP 2.0) el 8 de junio de 1995. Sin embargo, Zeev Suraski y Andi Gutmans reconstruyeron el núcleo de PHP liberando el resultado actualizado como PHP/FI 2.0 (Personal Home Page/Forms Interpreter) en 1997. En ese momento, el acrónimo PHP se cambió formalmente a “HyperText Preprocessor”.

En 1998, se lanzó PHP 3.0 y esta se convirtió en la versión más utilizada. Una de las mayores fortalezas de PHP 3.0 fue su fuerte capacidad de extensibilidad. Además de proporcionar a los usuarios finales una interfaz madura para múltiples bases de datos, protocolos y API, la facilidad de extender el lenguaje motivó a los desarrolladores para que desarrollaran y enviaran nuevos módulos. Otras características clave introducidas en

³<https://php.net/>

PHP 3.0 incluyen soporte de programación orientada a objetos y una sintaxis de lenguaje mucho más potente y consistente. En junio de 1998, con muchos nuevos desarrolladores de todo el mundo uniéndose al esfuerzo, PHP 3.0 fue anunciado por el nuevo equipo de desarrollo de PHP como el sucesor oficial de PHP/FI 2.0.

Después de aproximadamente nueve meses de pruebas públicas abiertas, cuando se anunció el lanzamiento oficial de PHP 3.0, ya estaba instalado en más de 70.000 dominios en todo el mundo y ya no estaba limitado a los sistemas operativos compatibles con POSIX. Una parte relativamente pequeña de los dominios que informan que PHP está instalado se alojó en servidores que ejecutan Windows 95, 98, NT y Macintosh. En su apogeo, PHP 3.0 se instaló en aproximadamente el 10% de los servidores web en Internet. De esta forma, PHP en todas sus versiones se encontraba instalado en más del 80% de los servidores web del mundo.

PHP 4.0 fue lanzado en mayo de 2000 con un nuevo núcleo conocido como Zend Engine 1.0. PHP 4.0 mejoró la velocidad y la confiabilidad con respecto a PHP 3.0. Además, PHP 4.0 agregó referencias, el tipo primitivo de dato Boolean, el soporte COM en Windows, el búfer de salida, muchas nuevas funciones de matrices, sesiones HTTP, entre servicios que fueron muy importantes en su época.

PHP 5.0 se lanzó en julio de 2004 con la versión actualizada de Zend Engine 2.0. Entre las muchas características nuevas en PHP 5.0 están: SQLite, soporte para nuevas funciones de MySQL, manejo de excepciones usando estructura `try..catch`, soporte SOAP integrado, biblioteca de filtros (en PHP 5.1), iteradores, entre muchos otros servicios que son de gran interés en el desarrollo de aplicaciones web en la actualidad.

El desarrollo de PHP 6.0 se inició en octubre de 2006. El cambio más significativo en esta nueva versión es el soporte nativo para Unicode. Aunque PHP todavía se usa principalmente para la generación de páginas web del lado del servidor, también se puede usar para ejecutar la línea de comandos Scripting o para crear aplicaciones gráficas con la ayuda de GTK+⁴.

1.6 jQuery

jQuery⁵ es una librería de JavaScript que proporciona algunas funciones útiles y oculta algunas incompatibilidades entre las implementaciones de JavaScript de diferentes navegadores. La característica principal de jQuery es que es más fácil de entender y escribir que el JavaScript simple.

La sintaxis de jQuery está diseñada para facilitar la navegación por un documento, seleccionar elementos DOM, crear animaciones, manejar eventos y desarrollar aplicaciones AJAX. jQuery también proporciona capacidades para que los desarrolladores creen complementos en la parte superior de la biblioteca de JavaScript. Esto permite a los desarrolladores crear abstracciones para interacción y animación de bajo nivel, efectos avanzados

⁴<https://www.gtk.org/>

⁵<https://jquery.com/>

y widgets temáticos de alto nivel. El enfoque modular de la biblioteca jQuery permite la creación de poderosas páginas web dinámicas y aplicaciones web [1].

1.7 Bootstrap

Bootstrap⁶ fue creado en Twitter a mediados de 2010. Inicialmente, Bootstrap era conocido como Twitter Blueprint. Tras unos pocos meses de desarrollo, Twitter celebró su primera semana de hackeo y el proyecto explotó a medida que los desarrolladores de todos los niveles de habilidad saltaban sin ninguna guía externa. Sirvió como la guía de estilo para el desarrollo de herramientas internas en la compañía durante más de un año antes de su lanzamiento público.

Bootstrap fue lanzado originalmente en agosto de 2011. En enero de 2012, se lanzó Bootstrap 2.0 que agregó soporte incorporado para Glyphicons. Bootstrap 2 soporta diseño web responsivo que significa que el diseño de las páginas web se ajusta dinámicamente, teniendo en cuenta las características del dispositivo utilizado. Bootstrap 3, lanzada en agosto de 2013, fue reescrita con el fin de que responda de forma predeterminada para desarrollo en dispositivos móviles. En agosto de 2015 se lanzó la versión alfa de Bootstrap 4. En agosto de 2017, se lanzó la versión beta de Bootstrap 4. Finalmente, Bootstrap 4 fue terminado en enero de 2018. Bootstrap 4 es compatible con las últimas versiones de Google Chrome, Firefox, Internet Explorer, Opera y Safari (excepto en Windows). Además, es compatible con IE9 y la última versión de soporte extendido de Firefox.

1.8 AJAX

AJAX es el acrónimo de Asynchronous JavaScript And XML. AJAX no es una tecnología sino que es un conjunto de principios basados en el trabajo realizado en Google que describe cómo manejar JavaScript en aplicaciones web más exigentes. Gmail y Google Maps tenían una característica particular. Sus interfaces eran dinámicas y de alto rendimiento y estaban a la vanguardia. Ambos hicieron un uso juicioso de JavaScript. Esto les permitió conectarse a un servidor de forma asíncrona y descargar datos nuevos sin cargar una página nueva.

El término AJAX fue propuesto por James Garrett en 2005. AJAX es básicamente una forma de usar muchas tecnologías existentes como HTML, CSS, JavaScript, XML, DOM y XMLHttpRequest. Cuando todas estas tecnologías funcionan en conjunto con la técnica AJAX, la interfaz de usuario se actualiza por capas sin recargar toda la página web. De esta manera, usando AJAX la aplicación web se comunica con el servidor obteniendo una porción de datos que se actualizan en la interfaz de usuario sin tener que recargar toda la página web.

Antes de 2005, la comunicación entre el lado del cliente y el lado del servidor era difícil de establecer. Los desarrolladores utilizaban iframes ocultos para rellenar los datos del

⁶<https://getbootstrap.com/>

servidor en el lado del cliente. Pero en 2005, James Garrett escribió el artículo llamado “AJAX: a new approach to Web applications” [6]. La tecnología clave utilizada en AJAX es XMLHttpRequest (XHR), primero inventada por Microsoft y luego utilizada por otros navegadores. XHR tiene capacidades para recuperar datos del lado del servidor y rellenar en el lado del cliente con la ayuda de las tecnologías existentes. Además, antes de 2005, los desarrolladores utilizaban diferentes tecnologías para la comunicación con el servidor como Java Applets.

1.9 REST

REST es el acrónimo de Representational State Transfer. REST es un modelo de arquitectura de software para crear sistemas que están distribuidas en red, el cual se originó en la tesis doctoral [4] de Roy Fielding, quien ayudó a especificar el protocolo HTTP, es miembro de la organización sin ánimo de lucro Apache Software Foundation⁷ y ha participado activamente en la World Wide Web desde 1993. REST define diferentes principios y elementos para cumplir con este modelo.

En la tesis doctoral, Fielding estableció algunas pautas o principios para trabajar con sistemas en red. Inicialmente, no se hablaba de respuestas con formato JSON o que la comunicación debía ser estrictamente con el protocolo HTTP o HTTPS. Por el contrario, REST habla de recurso, el cual es un elemento muy importante que se puede llevar a términos de bits, por lo tanto, podría llegar a ser desde un documento o un conjunto de caracteres.

Uno de los principales enfoques de REST es tener sistemas que puedan ser escalables. Para ello, se debería implementar el modelo arquitectural cliente - servidor, hacer uso de hiper-medios, identificar los recursos en la red y desarrollar un conjunto de operaciones para el recurso.

1.10 Patrón Modelo Vista Controlador

El patrón Modelo Vista Controlador (MVC) es un patrón arquitectural que tiene como objetivo separar la interfaz gráfica de la lógica que es donde se deben manejar las peticiones a la persistencia. Este patrón fue propuesto en los años 70 por Trygve Reenskaug como una solución al problema de dar a los usuarios finales el control sobre su información cuando existen múltiples vistas. Desde su creación, MVC ha creado un gran interés, dado a que se ha presentado diferentes versiones a partir de sus fundamentos.

Los componentes que se presentan son modelo, vista y controlador. Cada componente que interviene tiene una responsabilidad y comunicación restringida con los otros componentes. El modelo es la abstracción de la lógica de negocio que puede incluir servicios de persistencia, la vista permite mostrar el estado del modelo al usuario y el controlador gestiona las acciones del usuario.

⁷<http://www.apache.org/>

MVC fue propuesto inicialmente para aplicaciones de escritorio, pero se han presentado diferentes propuestas para adaptar este patrón en el desarrollo de aplicaciones web. Sin embargo, siempre se ha enfocado en separar las responsabilidades según el modelo cliente - servidor.

2

Conceptos Básicos de Programación

2.1 Tipos de datos

Las variables son las unidades de memoria en la ejecución de un programa. Cada variable corresponde a un tipo de dato para su debido procesamiento. En cada lenguaje de programación se define la sintaxis, para crear una instancia de la variable y su debida asignación. El valor de las variables pueden llegar a ser textuales, numéricas, booleanas, entre otras y a partir de allí se podrían formar estructuras como arreglos u objetos.

2.1.1 Variables

Para representar una variable en PHP, se usa el símbolo `$` al inicio del nombre. Generalmente, por buena práctica, el nombre de la variable debe comenzar en minúscula y por regla de sintaxis, no debe contener en el primer carácter números ni símbolos especiales. Para ser un nombre válido, se debe empezar por una letra o el carácter *underscore* `'_'`. Además, el nombre es sensible a mayúsculas y minúsculas. Así, en el siguiente ejemplo, cada variable es diferente:

```
1 | $indice = 3;  
2 | $Indice = 1;  
3 | $INDICE = 200;
```

Las variables se asignan por valor y no por referencia, lo que indica que si se copia una variable, la variable original no será afectada cuando se hagan cambios en la variable copia. En PHP, no es necesario inicializar variables. Aunque, dependiendo del algoritmo se deberá realizar una primer instancia de la variable. Si una variable no está instanciada

y no es referenciada, su valor será `NULL`. Para saber el tipo de variable, se puede usar la función `var_dump()`. El siguiente ejemplo muestra cómo usar la función `var_dump()`:

```
1 | $libro = "Fahrenheit 451";
2 | var_dump($libro);
3 | var_dump($editorial);
4 | $editorial = "Ballantine Books";
5 | var_dump($editorial);
```

La línea 2 produce la salida `string(14) "Fahrenheit 451"`, la línea 3 produce la salida `NULL` y la línea 5 produce la salida `string(16) "Ballantine Books"`

2.1.2 Variables constantes

Mediante la palabra reservada `const` se pueden definir valores que no van a cambiar en el transcurso de la ejecución de la página web. Cuando se define una constante no se debe usar el operador `$`. Las constantes normalmente se definen como atributos de clase y por buena práctica deben ser nombradas en mayúsculas para señalar que no pueden ser variables. Además, hay que recordar que se distingue entre mayúscula, minúsculas y las demás reglas especificadas anteriormente al nombrar una variable.

```
1 | const GRAVEDAD = 9.8;
2 | echo "Gravedad, en nuestro caso no cambia! ". GRAVEDAD;
```

2.1.3 Tipos primitivos de datos

PHP es un lenguaje débilmente tipado. Esto indica que al inicializar una variable no se define el tipo. Cuando se asigna un valor a una variable, PHP interpreta el tipo de dato a tratar.

La Tabla 2.1, muestra los tipos de datos que acepta de forma nativa en PHP.

Tipo de dato	Tipo
string	Escalar
integer	Escalar
float	Escalar
double	Escalar
boolean	Escalar
array	Compuesto
object	Compuesto
callable	Compuesto
iterable	Compuesto

Tabla 2.1: Tipos primitivos de datos en PHP

2.1.4 Ámbito de las variables

El ámbito hace referencia al contexto desde donde se define y se puede acceder a una variable. Existen dos ámbitos: global y local. En el ámbito global se definen las variables que son visibles para todo el contexto, en el cual el llamado se puede referenciar mediante la palabra reservada `global`. Así, una variable global puede ser accedida desde cualquier punto del código. El ámbito local normalmente se presenta en las funciones, de esta manera, las variables que se creen desde ahí, solo estarán visibles en dicha función. El siguiente código presenta la idea del ámbito global y local.

```
1 $b = 10;
2 $m = 2;
3 function calcularRecta($x){
4     global $m, $b;
5     echo $m * $x + $b;
6 }
7 calcularRecta(4);
8 echo "Se puede acceder a x? " . $x;
```

El anterior código permite calcular la ecuación de la recta, mediante una función que recibe de manera local una variable llamada `x` y dos variables globales (`m`, `b`), que están definidas por fuera de la función `calcularRecta`. La línea 4, indica que las variables `m` y `b`, son variables globales por lo que PHP, debe buscarlas en el contexto global. La línea 5 imprime el resultado del calculo accediendo a las variables locales y globales. Además, el código presenta una forma de crear variables globales y como poder llamarlas. La última línea generará un error debido a que en el contexto global no existe una variable llamada `x`, dado a que solo es visible dentro de la función.

2.1.5 Variables superglobales

Existen variables llamadas superglobales que se pueden llamar en cualquier contexto y fueron definidas internamente por PHP. Se puede acceder a este tipo de variables desde cualquier clase o función. De este modo, no es necesario hacer uso de la palabra reservada `global` antes de llamar a la variable.

Para llamar una variable superglobal, se debe anteriormente definir. Además de la palabra reservada `const` para definir una variable constante, hay otra forma de definir una constante de manera global. El siguiente código presenta esta definición.

```
1 define("NOMBRE_SITIO", "ABC");
2 echo "Este sitio web se llama: " . NOMBRE_SITIO;
```

La Tabla 2.2, presenta las principales variables de tipo superglobal, todas las variables son de tipo Array asociativo (llave - valor).

Variable	Descripción
<code>\$GLOBALS</code>	Contiene variables disponibles en el ámbito global.
<code>\$SERVER</code>	Contiene información del servidor como cabeceras, nombre del host, etc.
<code>\$GET</code>	Contiene datos enviados por el método GET o por URL.
<code>\$POST</code>	Contiene datos enviados por el método POST.
<code>\$FILES</code>	Contiene datos de archivos que se están subiendo al servidor.
<code>\$COOKIE</code>	Contiene Cookies HTTP.
<code>\$SESSION</code>	Contiene variables de sesión.
<code>\$REQUEST</code>	Contiene datos de una petición HTTP.
<code>\$ENV</code>	Contiene variables del entorno del servidor web.

Tabla 2.2: Variables superglobales

2.2 Operadores

En los lenguajes de programación, existen diferentes tipos de operadores que permiten realizar diferentes procesos. Estos operadores se clasifican en aritméticos, de asignación, lógicos y de comparación.

2.2.1 Operadores aritméticos

Los operadores aritméticos resuelven las operaciones básicas de suma, resta, multiplicación, división y módulo. Estos operadores se pueden aplicar a variables numéricas. Existen operadores especiales de incremento y decremento que permiten sumar y restar el valor 1 respectivamente a la variable. Los símbolos de las operaciones aritméticas se presentan en la Tabla 2.3.

Operador	Símbolo	Implementación
Suma	+	<code>\$a + \$b</code>
Resta	-	<code>\$a - \$b</code>
Multiplicación	*	<code>\$a * \$b</code>
División	/	<code>\$a / \$b</code>
Incremento	++	<code>\$a++</code>
Decremento	--	<code>\$a--</code>

Tabla 2.3: Operadores aritméticos

2.2.2 Operadores de asignación

Los operadores de asignación permiten depositar un valor en una variable. En muchos casos, es necesario realizar una operación aritmética de dos variables cuyo resultado debe depositarse en una de esas variables. Para estos casos, también se puede aplicar un operador especial. Los símbolos de los operadores de asignación se presentan en la Tabla

2.4.

Operador	Símbolo	Implementación
Asignación	=	\$a = \$b
Asignación con Suma	+=	\$a += \$b $\Leftrightarrow$ \$a = \$a + \$b
Asignación con Resta	-=	\$a -= \$b $\Leftrightarrow$ \$a = \$a - \$b
Asignación con Multiplicación	*=	\$a *= \$b $\Leftrightarrow$ \$a = \$a * \$b
Asignación con División	/=	\$a /= \$b $\Leftrightarrow$ \$a = \$a / \$b

Tabla 2.4: Operadores de asignación

2.2.3 Operadores lógicos

Los operadores lógicos resuelven expresiones booleanas. El resultado de estas operaciones será siempre verdadero **true** o falso **false**. Las operaciones booleanas básicas son AND, OR y NOT.

La operación lógica NOT, se aplica siempre sobre una premisa, que en un lenguaje de programación estará descrita en una variable booleana. Esta operación consiste en cambiar el valor de la premisa de falso a verdadero y viceversa. La operación lógica AND, indica que la salida será verdadera si y solo si todas sus entradas son verdaderas. La operación lógica OR, indica que la salida será falsa si y solo si todas sus entradas son falsas. Los símbolos de los operadores lógicos se presentan en la Tabla 2.5.

Operador	Símbolo	Implementación	Observación
NOT	!	!\$a	
AND	&&	\$a && \$b	Si \$a es falso, no se evalúa \$b
OR		\$a \$b	Si \$a es verdadero, no se evalúa \$b
AND	&	\$a & \$b	Siempre se evalúan \$a y \$b
OR		\$a \$b	Siempre se evalúan \$a y \$b

Tabla 2.5: Operadores lógicos

2.2.4 Operadores de comparación

Los operadores de comparación permiten la verificación de dos variables determinando si una de ellas es mayor, igual, menor o diferente de la otra. El resultado de estas operaciones será siempre verdadero **true** o falso **false**. Los símbolos de las operaciones de comparación se presentan en la Tabla 2.6.

2.2.5 Operadores a nivel de bits

Los operadores a nivel de bits permiten aplicar operaciones a los bits de los datos. Los símbolos de los operadores a nivel de bits se presentan en la Tabla 2.7.

Operador	Símbolo	Implementación
Igual	==	\$a == \$b
Diferente	!=	\$a != \$b
Mayor	>	\$a > \$b
Menor	<	\$a < \$b
Mayor o Igual	>=	\$a >= \$b
Menor o Igual	<=	\$a <= \$b

Tabla 2.6: Operadores de comparación

Símbolo	Implementación	Descripción
>>	\$a >> \$b	Desplaza a la derecha los bits de &a una distancia &b
<<	\$a << \$b	Desplaza a la izquierda los bits de &a una distancia &b
&	\$a & \$b	AND a nivel de bits
	\$a \$b	OR a nivel de bits
^	\$a ^ \$b	XOR a nivel de bits
~	~\$a	Complemento a nivel de bits

Tabla 2.7: Operadores a nivel de bits

2.3 Estructuras de programación

Las estructuras de programación o también llamadas estructuras de control permiten implementar procesos, tomar decisiones y realizar procesos con varias repeticiones.

2.3.1 Sentencias

Una expresión es un conjunto de variables unidas por operadores. Equivalen a instrucciones que el computador interpreta para realizar un proceso determinado. Una sentencia es una expresión que tiene al final punto y coma (;). Es posible incluir varias sentencias en una línea, sin embargo, se considera una buena práctica utilizar una línea para cada sentencia. Las siguientes líneas son ejemplos de sentencias en un programa.

```
1 $a = 10;
2 $b = 20;
3 $c = $a + $b;
```

2.3.2 Comentarios

Existen dos formas diferentes de introducir comentarios entre el código de la aplicación. Los comentarios son útiles para documentar el código implementado. Los comentarios se realizan de dos formas. La primera consiste en colocar el símbolo “//” en una línea de código y en seguida el texto del comentario. La segunda consiste en incluir el símbolo “/*”

al inicio del comentario y el símbolo “*/” al final del comentario. Esta segunda forma, permite hacer comentarios en varias líneas de código.

```
1 //Este es un comentario en una linea de código
2 /*
3 Este es un comentario
4 En diferentes lineas de código
5 */
```

2.3.3 Sentencia de impresión

Para imprimir un mensaje en el navegador se usa la sentencia `echo`.

```
1 echo "Hola mundo";
```

Esta sentencia permite concatenación de variables utilizando el carácter punto.

```
1 $a = 10;
2 echo "El valor de la variable a es: " . $a;
```

2.3.4 Estructura de condición if

La estructura de condición `if` se compone de una condición, la cual siempre debe arrojar un valor booleano, es decir, verdadero o falso. Esta condición debe encontrarse entre paréntesis. Esta estructura permite ejecutar un conjunto de instrucciones si se cumple la condición establecida. Este conjunto de instrucciones deben estar incluidos entre los símbolos `{` y `}`. Sin embargo, si solo se desea ejecutar una instrucción, no es necesario incluir los símbolos `{` y `}`. La sintaxis de esta estructura es la siguiente.

```
1 if(condicion){
2     instruccion_1;
3     instruccion_2;
4     ..
5     instruccion_n;
6 }
```

O también

```
1 if(condicion)
2     instruccion;
```

2.3.5 Estructura de condición if else

La estructura de condición `if else` se compone de una condición, la cual siempre debe arrojar un valor booleano, es decir, verdadero o falso. Esta condición debe encontrarse entre paréntesis. Esta estructura permite ejecutar un conjunto de instrucciones si se cumple la condición establecida y permite ejecutar otro conjunto de instrucciones diferentes si no se cumple la condición establecida. Este conjunto de instrucciones deben estar incluidos entre los símbolos `{` y `}`. La sintaxis de esta estructura es la siguiente.

```
1 | if(condicion){
2 |     instruccion_1_1;
3 |     instruccion_1_2;
4 |     ..
5 |     instruccion_1_n;
6 | }else{
7 |     instruccion_2_1;
8 |     instruccion_2_2;
9 |     ..
10 |    instruccion_2_n;
11 | }
```

2.3.6 Estructura de condición if else if

La estructura de condición `if else if` se compone de múltiples condiciones las cuales siempre deben arrojar un valor booleano, es decir, verdadero o falso. Estas condiciones deben encontrarse entre paréntesis. Esta estructura permite ejecutar un conjunto de instrucciones si se cumple la condición establecida y permite ejecutar otro conjunto de instrucciones diferentes si se cumple la otra condición establecida y así sucesivamente. Este conjunto de instrucciones deben estar incluidos entre los símbolos `{` y `}`. Esta estructura permite tener opcionalmente al final una estructura `else`, la cual se ejecuta si ninguna de las condiciones fueron verdaderas. Si una condición es verdadera, se ejecuta las instrucciones correspondientes y no consulta las siguientes condiciones. La sintaxis de esta estructura es la siguiente.

```
1 | if(condicion_1){
2 |     instruccion_1_1;
3 |     instruccion_1_2;
4 |     ..
5 |     instruccion_1_n;
6 | }else if(condicion 2){
7 |     instruccion_2_1;
8 |     instruccion_2_2;
9 |     ..
10 |    instruccion_2_n;
11 | }else if(condicion 3){
12 |     instruccion_3_1;
13 |     instruccion_3_2;
14 |     ..
15 |     instruccion_3_n;
16 | }
```

2.3.7 Operador ternario

La estructura de condición `if else` se puede implementar con una sintaxis avanzada que permite la codificación en una sola línea, la cual se denomina operador ternario. Este