



EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA



**Dirección General de Educación Tecnológica
Industrial y de Servicios**

Dirección Académica e Innovación Educativa

Subdirección de Innovación Académica

Departamento de Planes, Programas y Superación Académica

Cuadernillo de Aprendizajes Esenciales

Módulo III

Programación



Aprendizajes esenciales			
Carrera:	Programación		Semestre: Cuarto
Módulo/Submódulo:	Módulo III. Desarrolla aplicaciones Web. Submódulo 1. Construye páginas Web		
Aprendizajes esenciales o Competencias esenciales 1er parcial	Estrategias de Aprendizaje		Productos a Evaluar
Competencia profesional 1: Emplea HTML para construir páginas web. Situación 1.1: Elaborando una página informativa. Contenido: 1.1.1. Elementos básicos. 1.1.2. Etiquetas y atributos. 1.1.3. Estructura de una página web. 1.1.4. Títulos, textos y párrafos. 1.1.5. Formatos, líneas, imágenes, propiedades. 1.1.6. Listas y tablas.	Competencia profesional 1: Emplea HTML para construir páginas web. Evaluación diagnóstica La evaluación diagnóstica, es un instrumento que tiene el objetivo de reconocer los conocimientos y habilidades que tiene el estudiante acerca de algún tema, en este caso, sobre los elementos que integran a un sitio web. Para lograrlo, observa con atención las imágenes que se presentan en las figuras 1.1 y 1.2. Con base en los conocimientos de navegación adquiridos en la asignatura de Tecnologías de la Información y la comunicación (TIC) así como de la vida cotidiana, responde a los siguientes cuestionamientos: 1. En la figura 1.1, señala la dirección del sitio web que se está mostrando. 2. La dirección de un sitio web se integra por diferentes elementos. En la dirección que aparece en la siguiente línea, reconoce y asigna los nombres de cada elemento presentado: http://www.google.com.mx 3. En la flecha 1 colocada en la figura 1, se señala una línea que conduce al espacio indicado en la figura 1.2. ¿Cuál es el nombre que se le da a esa línea? ¿Cómo se llama al elemento al que conduce? 4. Menciona la diferencia entre una página y un sitio web. 5. Cuando has navegado en internet en busca de información. ¿Qué navegador utilizas?		Evaluación diagnóstica: Cuestionamientos contestados, correspondiente a la evaluación diagnóstica.

6. Escribe los navegadores que conoces.



Figura 1.1. Captura de pantalla. Tomado de:
<http://www.dgeti.sep.gob.mx/>



Figura 1.2. Captura de pantalla. Tomado de:

<http://www.dgeti.sep.gob.mx/index.php/la-dgeti-hoy/953-regreso-seguro-a-clases-5>

Situación 1.1: Elaborando una página informativa

1.1.1. Elementos básicos

Existen muchas herramientas (tools) que permiten crear, de una forma fácil y sencilla páginas web eficientes. No obstante, si eres un principiante en el diseño de páginas web, te será de gran ayuda considerar lo siguiente:

1. Comprender qué hace que una página web sea eficaz.
2. Ajustar la idea al proceso de diseño.
3. Y lo más importante: ¡disfrutar con el diseño de la página web!

Para iniciar en el fantástico mundo del diseño de sitios web, es conveniente conocer los términos que se emplean en este ambiente de desarrollo.

Internet.

La *red* más amplia utilizada en la actualidad, es internet (Internetwork System. Sistema de intercomunicación de redes). Podemos citar conceptos como estos: “(1) Red extensa constituida por una cantidad de redes menores. (2) Red nacional orientada a la investigación que engloba más de tres redes gubernamentales y académicas en 40 países”, sin embargo, también podemos referirnos a ella como: el banco de datos más grande del mundo o la red de redes, en la cual, se puede encontrar información de cualquier tema y es útil para científicos, maestros, estudiantes y público en general.

Web, WWW o W3.

Término muy empleado en la navegación por internet, que nace en 1989 gracias a la propuesta de Tim Berners-Lee, por lo que el 29 de marzo del

año 2014 festejó sus 25 años. Se refiere normalmente a un sistema de documentos interconectados por enlaces de *hipertexto* disponibles en internet. “La World Wide Web es un sistema de hipertexto de internet que brinda una forma atractiva y sencilla de explorarlo” para esto, existen diferentes navegadores como: Internet Explorer, Mozilla Firefox, Google Chrome, Opera, Safari, Netscape Navigator, entre otros, que conducen hacia los sitios de interés a tan solo unos clics de distancia. También conocido como red, maya o telaraña, como se observa en la figura 1.3.



Figura 1.3 Representación amigable de la “telaraña” (web).

Página Web.

Es un único archivo o documento electrónico que puede contener imágenes, archivos multimedia, texto organizado en párrafos, viñetas o tablas y otros elementos estáticos o dinámicos, escritos en un lenguaje de hipertexto conocido como HTML.

Sitio Web.

Es un servidor que contiene páginas web y otros archivos vinculados o relacionados entre sí mediante *hipervínculos*, para proporcionar información atractiva al usuario. Dado que se encuentra en línea las 24 horas al día en internet, es útil como medio informativo permanente e

inclusive permite que el usuario pueda interactuar con sus elementos, realizando desde búsquedas, compras, comunicación, hasta juegos.

HTML.

Es el lenguaje con el que se escriben las páginas Web, HyperText Markup Language (Lenguaje de marcado de hipertexto), considerado como un lenguaje de alto nivel, el cual indica a los navegadores cómo mostrar el contenido de una página Web. Permite escribir texto de forma estructurada, integrado por etiquetas que marcan el inicio y fin de cada elemento de la página.

URL.

Localizador uniforme de recursos (Uniform Resource Locator URL). “Los URL se especifican al explorador para acceder a las páginas Web y se encuentran incrustados dentro de las mismas páginas, para proveer enlaces de hipertexto a otras páginas”.

HTTP

El HTTP (Protocolo de Transferencia de Hipertexto) está orientado a transacciones, en un sistema de petición – respuesta entre un cliente y un servidor. Es el que define la sintaxis que utilizan los elementos de software de la arquitectura WEB para comunicarse, por lo tanto, es el que se utiliza en las transacciones www. A este protocolo se le cataloga sin estado, puesto que no guarda ninguna información sobre conexiones anteriores, es por esto que se usan las *cookies*, que es la información que un servidor puede almacenar en el cliente, para tomarla cuando necesite “mantener estado”, de ahí se trabajan con “sesiones”, pero, además, esto deja una apertura en el sistema cliente para poder localizar usuarios, ya que las *cookies* pueden guardarse en el nodo cliente por tiempo indeterminado. La información que es transmitida por este protocolo, recibe el nombre de recurso y se identifica mediante un URL.

	HTTPS. Es una versión de http para la transferencia segura de información, que puede utilizar cualquier método de cifrado siempre que sea entendido tanto por el servidor como por el cliente. Actividad 1: Diseñe en su cuaderno de apuntes, un mapa mental con la información de los elementos básicos para la elaboración de una página web. Actividad 2: ¡Vamos a jugar! Diseña un crucigrama que contenga los elementos básicos estudiados para que fomentes tu creatividad para organizar el cruce de palabras, al mismo tiempo que ¡aprendes jugando! 1.1.2. Etiquetas y atributos. El proceso de indicar las diferentes partes que componen la información se denomina marcar (markup en inglés). HTML es un lenguaje de etiquetas (también llamado lenguaje de marcado) y las páginas web habituales están formadas por cientos o miles de pares de etiquetas. De hecho, las letras "ML" de la sigla HTML significan "markup language", que es como se denominan en inglés a los lenguajes de marcado. Además de HTML, existen muchos otros lenguajes de etiquetas como: XML, SGML, DocBook y MathML. Etiqueta. Cada una de las palabras que se emplean para marcar el inicio y el final de una sección se denominan etiquetas o elementos, que están marcadas por los símbolos < y >. Estos elementos no se muestran en la página web, sin embargo, son los encargados de la presentación de la información ante el usuario. Las etiquetas pueden escribirse indistintamente en minúsculas o mayúsculas sin que genere un error de sintaxis. Se escriben en parejas,	Actividad 1: Mapa mental con los elementos básicos para la elaboración de una página web. Actividad 2: Crucigrama con los elementos básicos para elaborar una página web.
--	--	---

	enmarcando el inicio y fin (salvo algunas excepciones). El nombre inicial y final es el mismo, distinguiéndose el cierre, al anteponerle una diagonal (/), tal como se muestra en la figura 1.4. Una de las ventajas de estas etiquetas es que son muy sencillas de leer y escribir, tanto por las personas como por los sistemas electrónicos, pero la principal desventaja es que pueden aumentar considerablemente el tamaño del documento, es por eso que se usan etiquetas con nombres muy cortos.	
--	--	--

Etiqueta	Acción
<HTML> </HTML>	Indica que es un documento HTML o una página web
<HEAD> </HEAD>	Encabezado
<BODY> </BODY>	Cuerpo
<TITLE> </TITLE>	Título. El que aparece en la barra de título de la página web
<H1> </H1>	Títulos automáticos dentro de la página. Desde h1, hasta h6
<P> </P>	Párrafo
<CENTER> </CENTER>	Centrado
 	Texto en negritas
<I> </I>	Texto en cursivas
<U> </U>	Texto en subrayado
<S> </S>	Texto en tachado
 	Formato del texto
 	Imágenes
 	Lista numerada
 	Lista con viñetas
 	Elemento en una lista numerada o con viñetas
<TABLE> </TABLE>	Tabla
<TR> </TR>	Línea en una tabla
<TD> </TD>	Celda en una tabla
<A> 	Enlace o vínculo
 	Salto de línea

Figura 1.4 Etiquetas básicas para el diseño de páginas web.

Atributos.

A pesar de que se trata de un número de etiquetas muy grande, no es suficiente para crear páginas web complejas. Como no es viable crear una etiqueta por cada enlace diferente, la solución consiste en personalizar las etiquetas HTML mediante cierta información adicional llamada atributos. Los atributos son características específicas que se le aplican a las etiquetas. Se incluyen siempre dentro de la etiqueta de inicio, colocando el nombre del atributo, seguido del signo de igual (=) e inmediatamente después, el valor del atributo entre comillas (" "). Para separar cada atributo, se emplea el punto y coma (;) o simplemente un espacio. Algunos atributos se aplican a un elemento concreto, mientras que otros se pueden usar para muchos elementos diferentes.

Ejemplo: observa la sintaxis para enviar un mensaje de bienvenida al diseño de páginas Web, mediante el uso de etiquetas y atributos, tal como se observa en la figura 1.5.

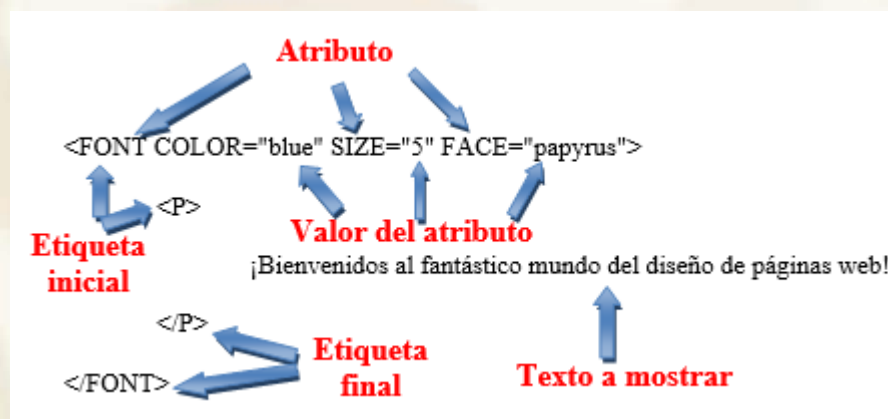


Figura 1.5 Etiquetas y atributos para colocar un mensaje en una página web.

Actividad 3:

Memorama alusivo a las etiquetas básicas de HTML.

Actividad 3: ¡Vamos a jugar! utilizando elementos que se puedan reciclar, como cartón, carpetas maltratadas o incluso CD's en desuso, diseña un memorama con las etiquetas de la figura 4, de tal forma que los pares, sean una etiqueta y su acción correspondiente. Una vez elaborado, invita a tus hermanos o familiares con los que vivas, a jugar y aprender juntos, fomentando la sana convivencia.

1.1.3. Estructura de una página web.

Un documento HTML inicia con la etiqueta `<html>` y finaliza con `</html>`. Todo lo que se encuentre entre estos elementos, será la página web. Dentro de `<html>` `</html>` se encuentran dos partes diferenciadas.

1. La primera `<head>` `</head>` es la cabecera. Reservada para incluir información que no se refiere directamente al contenido de la página, como: el título, los metadatos, estilos o código javascript (elementos que se estudiarán en capítulos venideros). La primera etiqueta que se suele aplicar es `<title>` `</title>`, que indica el título de la página (lo que el navegador muestra en la barra de títulos).
7. La segunda `<body>` `</body>` es el cuerpo. Reservado para mostrar el contenido de la página: fotos, párrafos, formularios, texto y todo lo que vemos en una página web.

La estructura básica de una página web es la siguiente:

`<html>`

`<head>`

`</head>`

`<body>`

`</body>`

`</html>`

Ejemplo: para mostrar el empleo de la estructura básica de un documento HTML, se realiza el diseño de una página, tal como se

muestra en la figura 1.6, con elementos mínimos, cuyo código se observa en la figura 1.7.

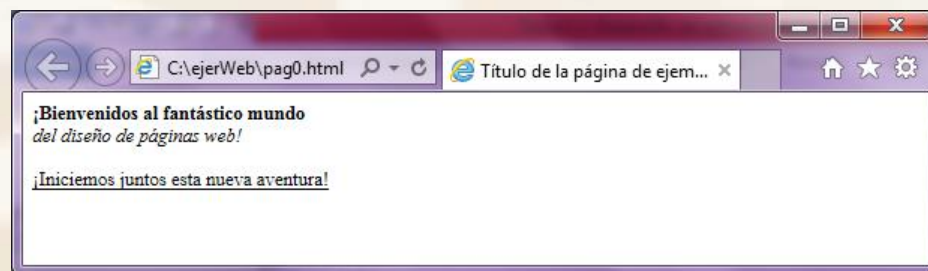
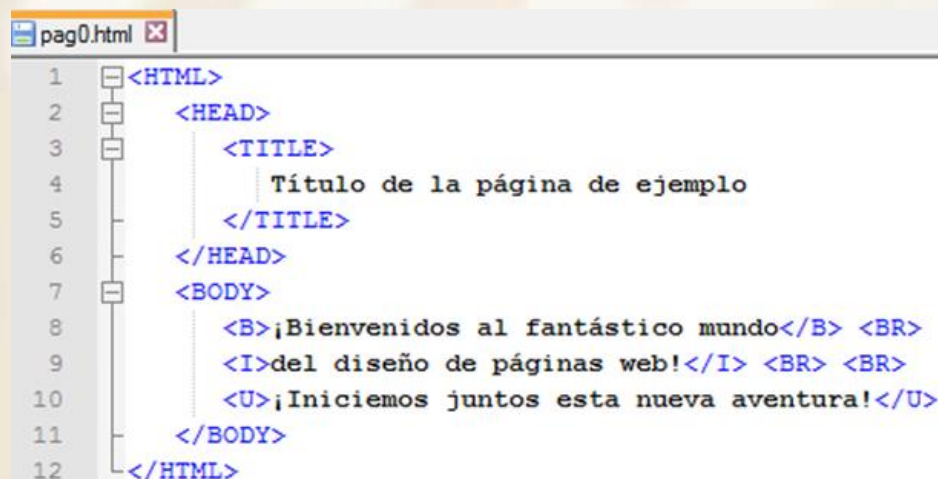


Figura 1.6. Página web básica.



The screenshot shows an HTML editor window with the file name 'pag0.html'. The code is as follows:

```
1 <HTML>
2   <HEAD>
3     <TITLE>
4       Título de la página de ejemplo
5     </TITLE>
6   </HEAD>
7   <BODY>
8     <B>¡Bienvenidos al fantástico mundo</B> <BR>
9     <I>del diseño de páginas web!</I> <BR> <BR>
10    <U>¡Iniciemos juntos esta nueva aventura!</U>
11  </BODY>
12 </HTML>
```

Figura 1.7. Etiquetas para la creación de una página web básica.

Explicación: los números de línea que se observan en la figura, pertenecen al editor, se muestran solo por comodidad, para la explicación. Estos números, no se deben incluir en la creación de los documentos HTML.

En la línea 1, inicia el documento HTML, finalizando en la línea 12.

El encabezado se encuentra delimitado por las etiquetas de las líneas 2 y 6.

El título, se encuentra en la línea 4, delimitado por sus etiquetas en las líneas 3 y 5.

Entre las líneas 7 y 11, marcadas por el inicio y fin del cuerpo, se coloca lo que el usuario observará en la página web.

En la línea 8, se muestra un mensaje en negritas. Observa que se encuentra delimitado por la etiqueta `<B> </B>`. Al final de la misma la misma línea, se observa la etiqueta `<BR>` que indica salto de línea. En el código HTML pulsar la tecla Enter, no produce un salto de línea visible en la página, es por esto que se debe indicar de manera explícita.

En la línea 9, se muestra un mensaje en cursiva, delimitado por la etiqueta `<I> </I>`, incluyendo dos saltos de línea `<BR>`.

La última línea mostrada en la página, se encuentra en la línea 10 de la figura, siendo un texto subrayado delimitado por `<U> </U>`.

1.1.4. Títulos, textos y párrafos.

Títulos.

La información que se publica en una página web, debe estar organizada, estableciendo títulos que enmarquen y le asignen la importancia o jerarquía a lo que se muestra. Para establecer títulos, HTML proporciona seis diferentes títulos que varían en tamaño, los cuales representan a niveles de importancia del mayor al menor, representadas por la etiqueta H1 al H6. Posteriormente, se mostrará cómo se le puede asignar un formato específico a los títulos de forma manual mediante el cambio de atributos.

Ejemplo: en la figura 1.8 se muestra el empleo de los seis diferentes estilos de títulos predefinidos en HTML, en el diseño de los datos para una portada sencilla, cuyo código se observa en la figura 1.9.

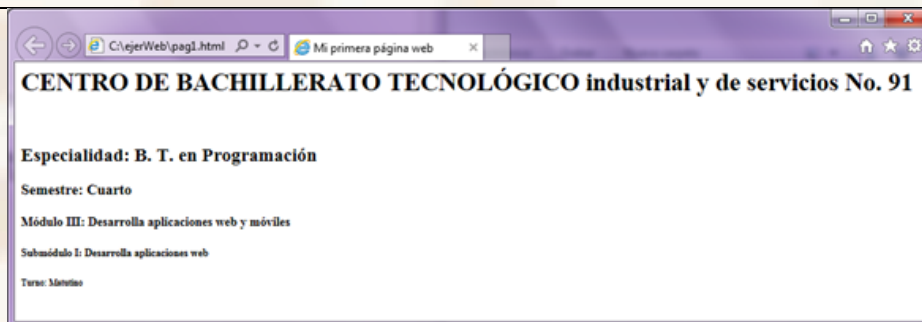


Figura 1.8. Página web con estilos de títulos automáticos.

```

1 <HTML>
2 <HEAD>
3   <TITLE>
4     Mi primera página web
5   </TITLE>
6 </HEAD>
7 <BODY>
8   <H1>CENTRO DE BACHILLERATO TECNOLÓGICO industrial y de servicios No. 91</H1><BR>
9   <H2>Especialidad: B. T. en Programación</H2>
10  <H3>Semestre: Cuarto</H3>
11  <H4>Módulo III: Desarrolla aplicaciones web y móviles</H4>
12  <H5>Submódulo I: Desarrolla aplicaciones web</H5>
13  <H6>Turno: Matutino</H6>
14 </BODY>
15 </HTML>
  
```

Figura 1.9. Código para escribir una página web con estilos de títulos automáticos.

Explicación: en la línea 1, inicia el documento HTML, finalizando en la línea 15.

El encabezado se encuentra delimitado por las etiquetas de las líneas 2 y 6.

El título, se encuentra en la línea 4, delimitado por sus etiquetas en las líneas 3 y 5.

Entre las líneas 7 y 14, marcadas por el inicio y fin del cuerpo, se coloca lo que el usuario observará en la página web, en este caso, los datos de una portada sencilla.

Actividad 4:

Documento en el bloc de notas con el código de la figura 1.9 y página web

	En la línea 8, se muestra el nombre de la escuela, delimitada por la etiqueta <H1></H1>, siendo la de mayor rango en los títulos, por lo que se muestra con el mayor tamaño de letra. Se observa también una etiqueta
 al final de la línea, para incluir un salto de línea adicional en el título principal. Observa que las demás líneas no necesitan de esa etiqueta dado que las etiquetas de títulos ya incluyen un salto de línea. De las líneas 9 a la 13, se emplean los títulos siguientes en menor rango. Observa que mientras más grande es el número que acompaña a la etiqueta H, menor es el tamaño de letra que se muestra en la página. Actividad 4: si tienes oportunidad de probarlo en una computadora... ¡Vamos a crearlo! Solo necesitas abrir un documento nuevo en un bloc de notas, escribir el código de la figura 1.9, guardarlo con el nombre pag1.html y con el tipo: todos los archivos. De preferencia, crea una carpeta con el nombre ejerWeb para que ahí guardes todas tus páginas. Una vez guardado el archivo, entra a la carpeta ejerWeb con el explorador de archivos, localiza la página: pag1.html y ejecuta con doble clic. Si necesitas hacer cambios, vuelve a guardar después de realizarlos y actualiza la página en tu navegador web con solo pulsar la tecla F5. Actividad 5: en tu cuaderno de apuntes, diseña una página web en la que coloques elementos para una portada, modificando el código de la figura 1.9 para que lo adaptes a los datos de tu plantel, turno e incluyas tanto tu nombre completo como los nombres del módulo y submódulos correspondientes al cuarto semestre de tu plan de estudios, empleando estilos automáticos. Textos. Los seres humanos nos comunicamos día a día mediante textos, que pueden ser orales o escritos. El texto permite la transmisión de mensajes	ejecutado en un navegador (en caso de contar con equipo de cómputo), en caso contrario, en el cuaderno de apuntes, con datos cambiados. Actividad 5: Página web diseñada en el cuaderno de apuntes, con las etiquetas necesarias, empleando una estructuración adecuada.
--	--	---

coherentes y ordenados a través de la palabra o la escritura auxiliándose de signos que, de acuerdo al contexto, emiten un significado diferente, pudiendo emplear pocas palabras, hasta un tamaño virtualmente infinito. Para mostrar la información organizada, se cuenta con el elemento: párrafo.

Párrafos.

La Real Academia de la Lengua Española, define al párrafo como: “Cada una de las divisiones de un escrito señaladas por letra mayúscula al principio de línea y punto y aparte al final del fragmento de escritura”. Es importante emplear el párrafo para organizar y proporcionar un sentido apropiado al mensaje que se pretende difundir, colocando en el párrafo inicial, la idea principal constituida por una oración que muestre la idea esencial del tema a tratar de la cual dependerán los otros párrafos.

En HTML para delimitar cada párrafo, se emplea la etiqueta <P> y </P>. Para que el párrafo se muestre alineado adecuadamente, la etiqueta se acompaña de los atributos que se presentan en la figura 1.10.

Atributo/Valor	Acción
ALIGN= “left”	Alineado recto a la izquierda, a la derecha se muestra en forma irregular o en bandera.
ALIGN = “right”	Alineado recto a la derecha, a la izquierda se muestra en forma irregular o en bandera.
ALIGN = “center”	Alineación centrada. Ambos extremos se muestran en forma irregular.
ALIGN = “justify”	Justificada. Ambos extremos se muestran rectos.

Actividad 6:

Mapa mental con los elementos del subtema 1.1.4.

Actividad 7:

Página web diseñada en el cuaderno de apuntes, con las etiquetas necesarias, empleando una estructuración adecuada.

Figura 1.10. Alineación de párrafos.

Actividad 6: Diseñe en un mapa mental, los elementos contenidos en el subtema 1.1.4.

Actividad 7: busca un libro que tengas en casa, localiza el primer capítulo, lee algunas páginas y selecciona, a manera de resumen, al menos 3 párrafos que consideres importantes. En tu cuaderno de apuntes, diseña una página web en la que coloques etiquetas y atributos necesarios para mostrar un letrero con el título del libro, otros para colocar el nombre del o los autores y la palabra: introducción. Posteriormente, coloca los párrafos seleccionados, finalizando con la palabra: resumen, la fecha y tu nombre. Debes incluir las alineaciones siguientes: Los títulos principales de manera centrada, los párrafos del resumen con alineación justificada, la fecha a la derecha y los restantes, a la izquierda.

1.1.5. Formatos, líneas, imágenes, propiedades.

Otro de los puntos importantes en el diseño web, es el aspecto que va a tomar el texto, es decir, la manera en que el usuario apreciará la información.

La forma de emplear estas etiquetas, inició con la aplicación tanto a fragmentos de texto como a párrafos enteros cada vez que se necesitara en la página, sin embargo, este sistema de trabajo ha quedado obsoleto, puesto que genera código redundante. La forma de trabajo actual sugiere el empleo de hojas de estilo en cascada (CSS) que estudiaremos en el tema 1.2.3. En este subtema lo trataremos con los formatos aplicados directamente al texto, para que conozcas su uso y puedas en un momento dado, realizar modificaciones a páginas ya existentes y que aún trabajen bajo este esquema.

Formatos de estilos.

Ya se ejemplificó en uso de la etiqueta H1 a la H6 que escribe títulos de diferentes tamaños, sin embargo, es conveniente resaltar los textos de forma más específica, empleando las etiquetas de estilos, tal como se exponen en la figura 1.11, cuyo resultado en una página web se observa en la figura 1.12.

Etiqueta	Acción
<CENTER></CENTER>	Centrado
 o 	Texto en negritas
<I> </I>	Texto en cursivas
<U> </U>	Texto en subrayado
<S> </S> o <STRIKE></STRIKE>	Texto en tachado
 	Texto enfatizado
<TT> </TT>	Texto en forma de teletipo
<BIG></BIG>	Texto en tamaño grande
<SMALL> </SMALL>	Texto en tamaño pequeño
	Subíndice
	Superíndice

Figura 1.11. Etiquetas para aplicarle estilo a un texto.

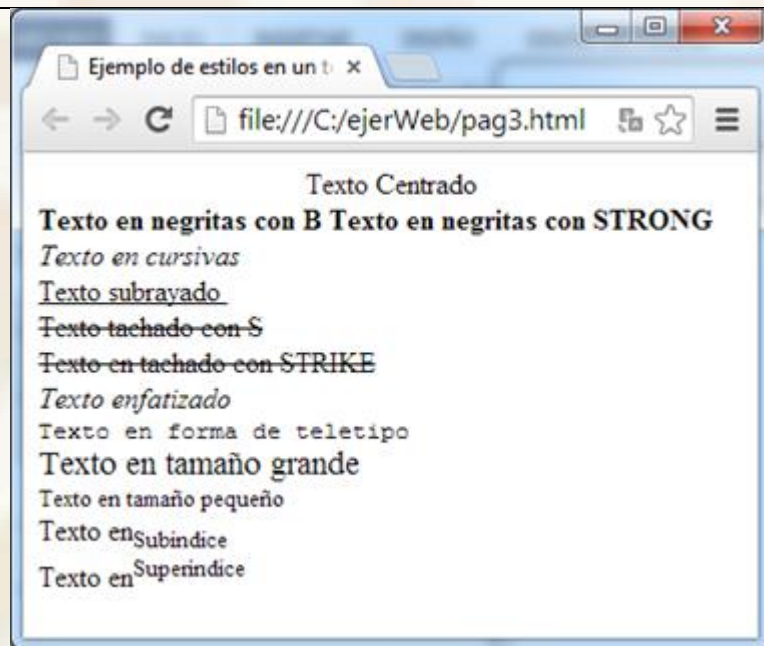


Figura 1.12. Vista de las etiquetas para aplicarle estilo a un texto.

Formato de fuentes.

Este formato es parecido al sistema que se emplea al darle formato a la fuente en un procesador de textos, requiere de algunas propiedades para cambiar de tamaño, tipo de letra y color. Los atributos en HTML para realizar estos cambios, se muestran en la figura 1.13.

Atributo	Acción
SIZE= "número"	Indica el tamaño del texto.
FACE= "tipo_letra"	Escribe el texto en el tipo de letra especificado.
COLOR= "código_color"	Indica el color del texto. El valor automático es negro.

Figura 1.13. Atributos de la etiqueta .

Ejemplo: escribiendo el siguiente código, se presentará la página web de la figura 1.14.

```
<FONT SIZE=6 FACE= "Algerian", "Times New Roman", "Arial" COLOR= "Blue">
```

Prueba de los atributos de la etiqueta FONT

```
</FONT>
```



Figura 1.14. Ejemplo de los atributos de la etiqueta .

Línea horizontal.

En ocasiones es necesario dividir o resaltar algún párrafo, es cuando se puede utilizar la etiqueta <HR>, que muestra una línea horizontal de tamaño determinable. Sus atributos se exhiben en la figura 1.15.

Atributo	Acción
WIDTH= "número" o "Porcentaje"	Indica el ancho de la línea. Se puede establecer un valor en píxeles o un porcentaje en función al ancho de la ventana del navegador.
ALIGN= "posición"	Coloca la línea en una posición específica. A la izquierda (left), derecha (right) o al centro(center) de la pantalla del navegador.
SIZE= "número"	Indica el grosor de la línea en píxeles.
NOSHADE	Rellena la línea. Se puede acompañar del atributo COLOR.

Figura 1.15. Atributos de la etiqueta <HR>

Ejemplo: para realiza la página web que se muestra en la figura 1.16, se escribe el código descrito en la figura 1.17.

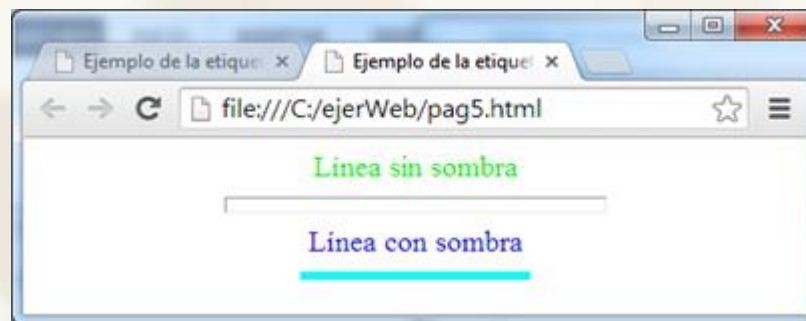


Figura 1.16. Modelo de la etiqueta <HR>

```
pag5.html
1 <HTML>
2   <HEAD>
3     <TITLE>
4       Ejemplo de la etiqueta HR
5     </TITLE>
6   </HEAD>
7   <BODY>
8     <FONT SIZE=4 FACE= "Arial" COLOR= "Green">
9       <CENTER>Línea sin sombra</CENTER>
10    </FONT>
11    <HR ALIGN= "center" SIZE= "10" width= "50%">
12    <FONT SIZE=4 FACE= "Arial" COLOR= "Blue">
13      <CENTER>Línea con sombra</CENTER>
14    </FONT>
15    <HR ALIGN= "center" SIZE= "5" width= "30%" COLOR="Aqua" NOSHADE>
16  </BODY>
17 </HTML>
```

Figura 1.17. Código para probar el empleo de la etiqueta <HR>.

Imágenes.

Uno de los elementos más fuertes que tienen los sitios web, es la inclusión de imágenes, ya que se puede aprovechar el impacto visual para atraer la atención hacia ella. En la figura 1.18, se muestran los atributos que dan formato a las imágenes en una página web, los cuales complementan a la etiqueta .

Atributo	Acción	
SRC= "ruta"	Indica la ruta o dirección URL de la imagen a mostrar.	
ALIGN= "posición"	Define la forma de distribuir el texto alrededor de la imagen. Con cinco posiciones: izquierda (left), derecha (right), arriba (top), en medio (middle) y abajo (bottom).	
ALT= "texto"	Muestra un texto alternativo cuando: el navegador no está configurado para mostrar imágenes, la imagen no se encuentra disponible en la ruta o en navegadores para usuarios con discapacidades visuales.	
LONGDESC	Es útil en navegadores adaptados para funciones de accesibilidad, mostrando la dirección URL del archivo y una descripción detallada.	
BORDER= "número"	Indica el grosor del borde o marco que se muestra rodeando la imagen. Si no se especifica, toma el valor de cero y aparece sin borde.	
WIDTH= "número"	Permite especificar la anchura de la imagen. Debes tener cuidado al realizar cambios ya que la imagen se puede deformar.	
HEIGHT= "número"	Permite especificar la altura de la imagen. Debes tener cuidado al realizar cambios ya que la imagen se puede deformar.	
Figura 1.18. Atributos de la etiqueta imagen . Las imágenes aceptadas, dependen del navegador o de un programa externo que permita su visualización. Dentro de los formatos más usuales que reconocen los navegadores, se encuentran las imágenes con extensión .gif, .jpg, .jpeg y .png.		

Ejemplo: El código que se muestra en la figura 1.19, muestra la página que se observa en la figura 1.20.

```
pag6.html
1 <HTML>
2   <HEAD>
3     <TITLE>
4       Ejemplo de imagen en la misma carpeta
5     </TITLE>
6   </HEAD>
7   <BODY>
8     <P>
9       Prueba de imagen en una página web
10      <IMG SRC="www2.png" ALT="Lo siento" WIDTH="100"
11        HEIGHT="80" ALIGN="middle">
12    </P>
13  </BODY>
14 </HTML>
```

Figura 1.19. Código para probar la etiqueta .



Figura 1.20. Muestra de la etiqueta .

Propiedades de la página.

Actividad 8:

Es importante tomar en cuenta que existen prioridades para ejecutar algunos comandos, como: revisar primero el atributo BACKGROUND antes que el atributo BGCOLOR, o revisar primero el atributo COLOR de la etiqueta antes que el atributo TEXT de la etiqueta <BODY>. Existen propiedades que ayudan a dar formato a la página web completa, las cuales están mostradas en la figura 1.21 y se refieren a la etiqueta <BODY>.

Atributo	Acción
BGCOLOR=	
"código_color"	Especifica un color de fondo para la página web.
BACKGROUND=	
"imagen"	Especifica una imagen como fondo de la página web.
TEXT= "código_color"	Define el color del texto de la página.
LINK= "código_color"	Define el color de los hipervínculos no visitados.
VLINK= "código_color"	Define el color de los hipervínculos ya visitados.
ALINK= "código_color"	Define el color del hipervínculo activo.
LEFTMARGIN=	
"número" o	Especifica el margen izquierdo en donde se mostrará el
MARGINWIDTH=	primer elemento dentro de la página.
"número"	
TOPMARGIN= "número"	
o MARGINHEIGHT=	Especifica el margen superior en donde se mostrará el
"número"	primer elemento dentro de la página.

Página web diseñada en el cuaderno de apuntes, con las etiquetas necesarias, empleando una estructuración adecuada.

Figura 1.21. Atributos de la etiqueta <BODY> </BODY>.

Actividad 8: ¡Estás listo para crear páginas sencillas y atractivas! Las representaciones culturales proporcionan una identidad a cada región ¿cuál es tu preferida? De acuerdo a tu localidad, selecciona una representación cultural (festividades, rituales, encuentros, entre otros) y elabora un ensayo en tu cuaderno de apuntes, redactando las diferentes actividades que se realizan al respecto. Puedes acompañarlo de palabras a color, remarcando títulos, párrafos e incluir imágenes recortadas y pegadas o dibujadas directamente. Una vez que concluyas, ahora, escribe, también en tu cuaderno de apuntes, las etiquetas necesarias para generar una página web con esa información. No olvides incluir tu nombre y otros datos personales ¡Emplea tu creatividad y conocimientos adquiridos!

1.1.6. Listas y tablas.

Listas.

Una forma muy usual de organizar información es empleando listas, que pueden ser ordenadas o no. Se emplean en la vida diaria cuando se elabora una lista de asistencia, una lista de amigos para invitarlos a una celebración, una lista de ingredientes para preparar un postre, una lista de puntos a considerar en la exposición del tema en una asignatura, así como una gran cantidad de momentos o situaciones que podemos representar mediante una lista.

En una página web se pueden incluir listas de tres tipos:

- Ordenadas: Se acompañan de un número al lado izquierdo del texto. Cada elemento siguiente, toma en forma automática el valor consecutivo, por lo que siempre se muestran en orden.
- No ordenadas: las que muestran la información organizada mediante puntos como se emplea en este texto. La imagen que distingue a cada punto que se encuentra a la izquierda se le conoce como viñeta o boliche y puede ser sujeto a modificaciones.

- De definiciones: muestran un conjunto de definiciones, similar a un diccionario.

En la figura 1.22, se muestran las etiquetas con las que cuenta HTML para mostrar listas. La etiqueta `<LI></LI>`, se utiliza para registrar cada uno de los elementos a contemplar en la lista, ya sea en forma ordenada o no; por lo que se puede observar que, quien distingue la forma de presentación, es la etiqueta que la engloba.

Etiqueta	Acción
<code></code>	Diseño de listas ordenadas. Emplea números.
<code></code>	Diseño de listas no ordenadas. Utiliza viñetas
<code></code>	Cada uno de los puntos en una lista, pudiendo ser ordenada o no.
<code><DL></DL></code>	Indica que dentro del espacio que enmarca, se colocará una lista de definiciones.
<code><DT></DT></code>	Indica el término que se va a definir, es decir el título o palabra a definir.
<code><DD></DD></code>	Dentro de esta etiqueta se escribe la definición del término indicado en la etiqueta <code><DT></DT></code> .

Figura 1.22. Etiquetas para el diseño de listas.

Ejemplo de listas ordenadas: ¿Conoces los signos zodiacales? Representan a las 12 constelaciones que enmarca el universo en que vivimos y que, de acuerdo a nuestra fecha de nacimiento, se dice que nacimos bajo un signo zodiacal determinado, el cual, según los astrólogos, rige algunos rasgos de nuestra personalidad. En la figura 1.23, se muestran en forma ordenada, cada uno de los 12 signos zodiacales

mencionados, que se logra mediante el código de etiquetas mostrado en la figura 1.24.

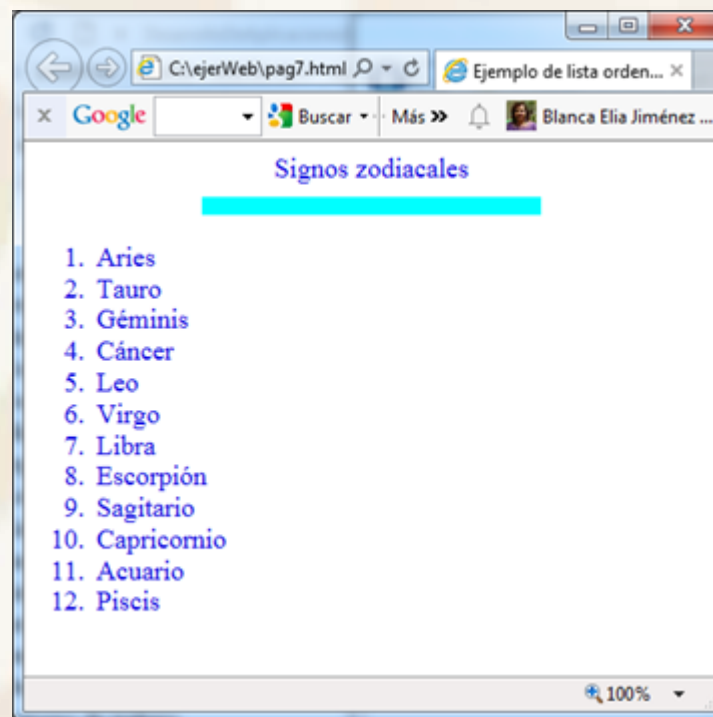


Figura 1.23. Lista ordenada de los signos zodiacales.

Actividad 9:

Página web diseñada en el cuaderno de apuntes, con las etiquetas necesarias, empleando una estructuración adecuada para resolver el problema planteado.

```
pag7.html x
1 <HTML>
2   <HEAD>
3     <TITLE>
4       Ejemplo de lista ordenada
5     </TITLE>
6   </HEAD>
7   <BODY>
8     <FONT SIZE=4 FACE= "Arial" COLOR= "Blue">
9     <CENTER>Signos zodiacales</CENTER>
10    </FONT>
11    <HR ALIGN= "center" SIZE= "10" width= "50%" COLOR="Aqua" NOSHADE>
12    <FONT SIZE=4 FACE= "Arial" COLOR= "Blue">
13    <OL>
14      <LI>Aries</LI>
15      <LI>Tauro</LI>
16      <LI>Géminis</LI>
17      <LI>Cáncer</LI>
18      <LI>Leo</LI>
19      <LI>Virgo</LI>
20      <LI>Libra</LI>
21      <LI>Escorpión</LI>
22      <LI>Sagitario</LI>
23      <LI>Capricornio</LI>
24      <LI>Acuario</LI>
25      <LI>Piscis</LI>
26    </OL>
27    </FONT>
28  </BODY>
29 </HTML>
```

Figura 1.24. Código para mostrar una lista ordenada de los signos zodiacales.

Actividad 9: Empleando tu creatividad, diseña una página web en tu cuaderno de apuntes, que contenga una lista con los números del uno al veinte, colocando una lista numerada, con su escritura en inglés. ¡No olvides colocar otros elementos que proporcionen calidad y vista agradable a tu página!

Tablas.

Una forma muy interesante de organizar la información, es el empleo de tablas, ya que permite distribuir los datos de forma atractiva, lógica,

divertida y atrevida, permitiendo el desarrollo de la creatividad para plasmar lo que se quiere mostrar para lograr un gran impacto. Las tablas clásicas, están integradas por filas (horizontales) y columnas (verticales), formando una rejilla, en donde se puede escribir texto, mostrar imágenes, archivos multimedia o colocar hipervínculos. Cada espacio que integra la tabla, recibe el nombre de celda, como la que se utiliza en Excel.

La figura 1.25, muestra una tabla clásica, integrada por cinco filas y tres columnas. La forma de organizar la información en la tabla (<TABLE>), es mediante filas o renglones (<TR>), que se refieren a un mismo elemento. En este ejemplo, cada estación del año. Dentro de cada fila, se distribuye celda a celda (<TD>) la información correspondiente a la estación en los dos hemisferios de la tierra, que, observándolas en conjunto, integran las columnas de la tabla. Cuenta con el título: “Datos astronómicos” mostrado en la parte superior de la tabla, las columnas tienen un encabezado con color de fondo rosa, cuyo texto se encuentra centrado respecto a la celda. La tabla tiene un ancho de 80% con respecto a la página, por lo que su tamaño cambia conforme se modifica la ventana del navegador.



Estaciones del año		
Datos astronómicos		
Estación	Hemisferio norte	Hemisferio sur
Primavera	Del 21 de marzo hasta el 20 de junio	Del 21 de septiembre hasta el 20 de diciembre
Verano	Del 21 de junio hasta el 20 de septiembre	Del 21 de diciembre hasta el 20 de marzo
Otoño	Del 21 de septiembre hasta el 20 de diciembre	Del 21 de marzo hasta el 20 de junio
Invierno	Del 21 de diciembre hasta el 20 de marzo	Del 21 de junio hasta el 20 de septiembre

Figura 1.25. Ejemplo de tabla clásica.

Para lograr la distribución de información en tablas dentro de una página web, se emplean las etiquetas mostradas en la figura 1.26.

Etiqueta	Acción
<code><TABLE></TABLE></code>	Crea una tabla.
<code><TR></TR></code>	Crea una fila. Debe estar dentro de una tabla y contener al menos una celda. Se coloca un juego de etiquetas por cada fila que se necesite en la tabla.
<code><TD></TD></code>	Crea cada celda. Debe estar dentro de una fila. Se coloca un juego de etiquetas por cada celda que se necesite.
<code><TH></TH></code>	Se emplea para colocar los encabezados en las columnas de la tabla. Muestra la información en negritas y centrado.
<code><CAPTION></CAPTION></code>	Coloca un título a la tabla, con la opción de colocarla encima o debajo de la misma, acompañada del atributo ALIGN.

Figura 1.26. Etiquetas para el diseño de tablas.

En la figura 1.27, se muestran los atributos que proveen de características propias a cada celda, fila o tabla, favoreciendo la calidad en la presentación de la información.

	Atributo/Valor	Acción	
	BORDER= "n"	Enmarca cada celda de la tabla con un borde de grosor "n" representado en pixeles. Si se omite, el borde es cero, es decir, la tabla se presenta sin bordes.	
	ALIGN= "posición"	Se emplea para alinear el contenido de la tabla, fila o celda, dependiendo de la etiqueta en donde se emplee. Las posiciones son: izquierda (left), derecha (right), centrado (center) y justificado (justify). Para la etiqueta CAPTION: arriba (top) y abajo (bottom).	
	VALIGN="posición"	Se emplea para alinear verticalmente el contenido de la tabla, fila o celda, dependiendo de la etiqueta en donde se emplee. Las posiciones son: arriba (top), en medio (middle) y abajo (bottom).	
	WIDTH= "valor"	Permite establecer el valor de la anchura de la tabla, ya sea en porcentaje (respecto a la página) o pixeles (fijo). Si se emplea dentro de una celda, el porcentaje se tomará respecto a la tabla.	Actividad 10: Página web diseñada mostrando la tabla de la figura 1.25, en el cuaderno de apuntes.
	HEIGHT= "valor"	Permite establecer el valor de la altura de la tabla, ya sea en porcentaje (respecto a la página) o pixeles (fijo).	Actividad 11: Página web con el horario de actividades semanales, en el cuaderno de apuntes.

	BGCOLOR= "código_color"	Se emplea para establecer un color de fondo en una tabla, fila o celda, dependiendo de la etiqueta en donde se emplee. El formato de color, es el mismo que se empleó en la figura 1.21 y que se ampliará en el subtema 1.2.3.3.	
	BACKGROUND="ruta_imagen"	Indica la ruta y nombre de la imagen a mostrar. Si se emplea en la tabla, la imagen se multiplicará, mostrándose varias veces en la tabla. Si se emplea en una celda, la imagen solo servirá de fondo en la celda indicada.	
	CELLSPACING= "n"	Permite establecer el valor (en pixeles) de la separación entre las celdas. Se emplea en la etiqueta <TABLE></TABLE>.	
	CELLPADDING= "n"	Permite establecer el valor (en pixeles) de la separación entre el borde y el contenido dentro de las celdas. Se emplea en la etiqueta <TABLE></TABLE>.	
	COLSPAN= "n"	Permite combinar una celda que abarque "n" columnas. Se emplea en la etiqueta <TD></TD>.	
	ROWSPAN= "n"	Permite combinar una celda que abarque "n" filas. Se emplea en la etiqueta <TD></TD>.	

FRAME= "posición"	Permite colocar un borde solo a una parte de la tabla. Las posiciones son: borde superior (above), borde inferior (below), borde superior e inferior (hsides), borde a la izquierda (lhs), borde a la derecha (rhs), bordes verticales (vsides) borde exterior (box) y sin borde (void).
BORDERCOLOR= "código_color"	Se emplea para cambiar el color de los bordes en una tabla, fila o celda, dependiendo de la etiqueta en donde se emplee. Además del código de color, puede emplear dos valores: claro (bordercolorlight) y oscuro (bordercolordark).

Figura 1.27. Atributos empleados en las etiquetas para el diseño de tablas.

Actividad 10: en tu cuaderno de apuntes, diseña una página web que muestre la tabla indicada en la figura 1.25 ¡Puedes modificar los elementos e incluir otros, de acuerdo a tu creatividad!

Actividad 11: un elemento muy importante para el logro de nuestras metas, es la planificación de tiempos, por tal motivo, te invito a que generes un horario de actividades semanales que te ayuden a organizarte, plasmándolo en tu cuaderno de apuntes. Después, crea ese horario en una página web y ¡Lleva a cabo la planeación de tu tiempo!

Nota: la información, figuras y ejemplos, fueron tomados del libro: **Módulo III. Desarrolla aplicaciones web y móviles.** Autora: Blanca Elia

	Jiménez Guzmán. Editorial: FCE, SEP, DGETI. Primera edición. 2016. México.	
Aprendizajes esenciales o Competencias esenciales 2º parcial	Estrategias de Aprendizaje	Productos a Evaluar
Situación 1.2: Planteando, diseñando y construyendo un sitio web. Contenido: 1.2.1. Sitios web. 1.2.2. Vínculos. 1.2.3. Organización del sitio. Competencia profesional 2: Utiliza CSS para dar formato a páginas web. Situación 2.1: Creando y aplicando un tema mediante hojas de estilo. Contenido: 2.1.1. Definición. 2.1.2. Evolución. 2.1.3. Ventajas. 2.1.4. Elementos de diseño. 2.1.5. Creación.	Situación 1.2: Planteando, diseñando y construyendo un sitio web. 1.2.1. Sitios web. “Un sitio web es un conjunto organizado y coherente de páginas Web que tiene como función ofrecer, informar, publicitar o vender contenidos, productos y servicios al resto del mundo. Para que un sitio Web pueda ser visitado por otras personas es necesario que se encuentre alojado en un servidor. Se trata de una computadora conectada a la World Wide Web con espacio en disco y conectividad suficiente para albergar sitios y servirlos al resto de la comunidad de usuarios de Internet a través de direcciones IP o nombres de dominio”. Tomado de EcuRed (www.ecured.cu). 1.2.2. Vínculos. En el diseño de una página web es importante la incorporación de elementos que faciliten la exploración o navegación, lo cual aumenta considerablemente la funcionalidad y organización de un sitio web, permitiendo su interconexión con diferentes recursos y sitios afines en todo el mundo. Vínculo (link). Conocido también como liga, hipervínculo, enlace o hiperenlace. Es una ruta de acceso que permite un desplazamiento rápido hacia temas	

específicos, ya sea de manera interna (dentro de la página web) o externa (en otra página web, pudiendo estar dentro o fuera del sitio web).

Tipos de enlaces:

8. A la misma página (anclaje).
9. A otra página dentro del mismo sitio web.
10. A otra página en diferente sitio web.
11. A una dirección de correo electrónico.

Ejemplo de vínculos:

En la figura 1.28, se plasman algunos códigos para ejemplificar el empleo de los vínculos más utilizados en una página web.

Tipo	Código	Significado
Definiendo el ancla.	<code></code>	"cap1" es el ancla invisible que se ubica en el destino del vínculo.
A la misma página.	<code></code> Capítulo 1 <code></code>	"#cap1" es la referencia al ancla previamente definida. Se ubica en el origen del vínculo. Capítulo 1, es el texto que aparecerá subrayado y que ejecutará el vínculo.
A otra página dentro del mismo sitio web.	<code></code> Advertencia y Dedicatoria <code></code>	"pag13.html" es el nombre del documento que se abrirá al ejecutar el vínculo. En caso de encontrarse en una carpeta diferente, es necesario indicar la ruta para acceder a ella.
A otra página fuera del sitio web.	<code></code> Acerca del autor <code></code>	Observe la ruta del sitio web al que se hace referencia. Al pulsar el vínculo, la página se abre en una ventana nueva, al indicárselo el valor "_blank" del atributo TARGET.

Figura 1.28. Ejemplos de tipos de enlaces.

1.2.3. Organización del sitio.

La organización de información en cualquier ámbito, se ha considerado prioritario, es especial por que influye en el proceso de la toma de decisiones. Es por esto que, en los sitios web, se vuelve fundamental

considerar una adecuada organización, desde el momento del diseño, considerando dos aspectos relevantes:

1. La forma en que se presenta la información al usuario.
2. La calidad del contenido.

En la figura 1.29, se observan herramientas que pueden ser útiles para la organización de un sitio web, que además, te servirá para crear un mapa del sitio (un archivo en el que puedes enumerar las páginas web de tu sitio, para informar a los motores de búsqueda acerca de la organización del contenido del sitio, cuya imagen se representa en la figura 1.30).

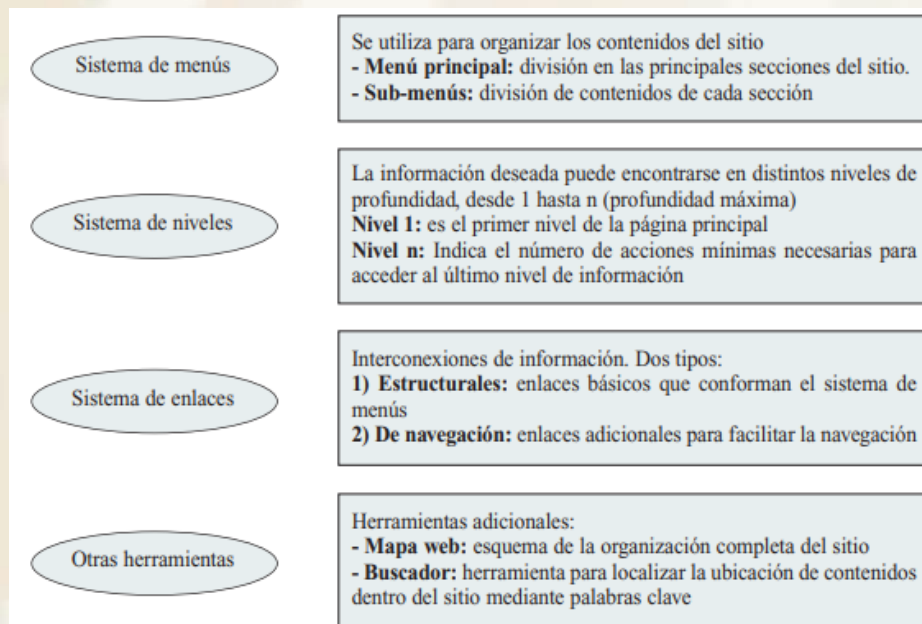


Figura 1.29. Herramientas utilizadas para la organización de un sitio web. Tomado de:

https://www.esic.edu/documentos/revistas/reim/100916_155728_E.pdf

Actividad 12:

Mapa del sitio.

Sitio web integrado con las páginas web indicadas en el mapa del sitio, conteniendo enlaces de llamada y regreso a la página principal. Todo realizado en el cuaderno de apuntes.



Figura 1.30. Mapa del sitio en imagen. Tomado de internet.

Actividad 12: En tu cuaderno de apuntes, diseña un sitio web, iniciando con la estructuración visual del mapa del sitio, tomando como ejemplo la figura 1.30. Deberás realizar un sitio web de al menos tres niveles, desarrollando la página web principal que va a contener los enlaces de llamada a cada página, que, a su vez, tendrán información específica. Deberás incluir enlaces que retornen a la página principal para continuar la navegación. El sitio web lo realizarás con el tema de tu preferencia, considerando los elementos que tengas en tu entorno, sin salir de casa y con una página de: “Acerca de” para mostrar los datos del autor y detalles de la especialidad, módulo y Submódulo que cursas, entre otros elementos de identificación ;Observa a tu alrededor y notarás muchos temas que puedes analizar, profundizar y representar mediante el diseño de un sitio web!

Competencia profesional 2: Utiliza CSS para dar formato a páginas web.

Situación 2.1: creando y aplicando un tema mediante hojas de estilo.

2.1.1. Definición.

Las hojas de estilos, también conocidas como CSS (Cascading Style Sheets), que significan hojas de estilos en cascada, por sus siglas en inglés, son archivos que acompañan a las páginas web para agrupar atributos que brindan formato a un texto, párrafo, tabla, vínculo o cualquier otro elemento utilizado en la página web. Se guardan con una extensión .CSS. A partir del HTML5, se le caracteriza por ser un lenguaje de marcas de tipo descriptivo, debiendo limitar el empleo de sus etiquetas a la estructura de la información en una página web y para los aspectos de diseño, en cuanto a la presentación visual ante el usuario, se debe realizar en los CSS3. Además, combinados con instrucciones utilizadas en lenguajes como: *JavaScript* o *VBScript*, pueden proporcionar movimientos, efectos y hasta interactividad con el sitio, convirtiéndolo en dinámico.

2.1.2. Evolución.

Las versiones de las hojas de estilos, también van avanzando. Actualmente se cuenta con la versión tres o nivel 3, por lo que se puede encontrar información referente a CSS3.

Con información obtenida del sitio: <https://uniwebsidad.com/libros/css/capitulo-1/breve-historia-de-css>, se presenta lo siguiente:

“Las hojas de estilos aparecieron poco después que el lenguaje de etiquetas SGML, alrededor del año 1970. Desde la creación de SGML, se observó la necesidad de definir un mecanismo que permitiera aplicar de forma consistente diferentes estilos a los documentos electrónicos.

El gran impulso de los lenguajes de hojas de estilos se produjo con el boom de Internet y el crecimiento exponencial del lenguaje HTML para la creación de documentos electrónicos. La guerra de navegadores y la falta de un estándar para la definición de los estilos dificultaban la

Actividad 13:
Línea de tiempo representando las hojas de estilos.

creación de documentos con la misma apariencia en diferentes navegadores.

El organismo W3C (World Wide Web Consortium), encargado de crear todos los estándares relacionados con la web, propuso la creación de un lenguaje de hojas de estilos específico para el lenguaje HTML y se presentaron nueve propuestas. Las dos propuestas que se tuvieron en cuenta fueron la CHSS (*Cascading HTML Style Sheets*) y la SSP (*Stream-based Style Sheet Proposal*).

La propuesta CHSS fue realizada por Håkon Wium Lie y SSP fue propuesto por Bert Bos. Entre finales de 1994 y 1995 Lie y Bos se unieron para definir un nuevo lenguaje que tomaba lo mejor de cada propuesta y lo llamaron CSS (*Cascading Style Sheets*).

En 1995, el W3C decidió apostar por el desarrollo y estandarización de CSS y lo añadió a su grupo de trabajo de HTML. A finales de 1996, el W3C publicó la primera recomendación oficial, conocida como "CSS nivel 1". A principios de 1997, el W3C decide separar los trabajos del grupo de HTML en tres secciones: el grupo de trabajo de HTML, el grupo de trabajo de DOM y el grupo de trabajo de CSS.

El 12 de mayo de 1998, el grupo de trabajo de CSS publica su segunda recomendación oficial, conocida como "CSS nivel 2". La versión de CSS que utilizan todos los navegadores de hoy en día es CSS 2.1, una revisión de CSS 2 que aún se está elaborando (la última actualización es del 8 de septiembre de 2009). Al mismo tiempo, la siguiente recomendación de CSS, conocida como "CSS nivel 3", continúa en desarrollo desde 1998".

Actividad 13: haciendo uso de tu creatividad, diseña en tu cuaderno de apuntes, una línea de tiempo para representar la evolución de las hojas de estilos.

2.1.3. Ventajas.

1. Que se guarde en un archivo independiente, permite su empleo en todas las páginas que integren el sitio web.
2. Al tener la información de diseño de todo un sitio web, de manera centralizada, facilita su mantenimiento y actualización.
3. Cada usuario puede diseñar localmente su propia hoja de estilo, mejorando la accesibilidad y brindando comodidad.
4. Se pueden programar diferentes hojas de estilos en un mismo sitio, para ser utilizados en diferentes dispositivos o medios a mostrar.
5. El programar y guardar por separado el diseño, hace que el código HTML5 se reduzca y sea más entendible.

2.1.4. Elementos de diseño.

Selectores en CSS3.

Para diseñar una hoja de estilos, se utilizan selectores, que son los elementos de la página web a la que se le aplicará el estilo. Consta de un nombre, un par de llaves y dentro de ellas, se escriben las propiedades y valores que establecerán el diseño. La figura 1.31, muestra la sintaxis.

```
Selector{
    propiedad: valor;
    propiedad: valor;
    ...
}
```

Figura 1.31. Sintaxis de los selectores en un CSS3.

Existen varios tipos de selectores, de los cuales se ejemplificarán los tres más utilizados.

Selectores de tipo.

Representan a las etiquetas de HTML, por ejemplo: <H1>, <P>, <TABLE>, <BODY>, entre otras.

Selectores de clase.

Son atributos generales, agrupados y diseñados para que puedan ser aplicados a distintos elementos. Se les asigna un nombre que lo identifique y se antecede por un punto. Para referenciarlos en el código HTML, se anexa el atributo: CLASS.

Pseudo-clases y pseudo-elementos.

Hacen referencia a un estado y no a una etiqueta, por ejemplo: si un vínculo ha sido visitado. Se nombran con el atributo correspondiente, seguido de dos puntos y posteriormente el estado.

- Pseudo-clases: : first-child, : link, : visited, : hover, :active, : focus y : lang.
- Pseudo-elementos: : first-letter.

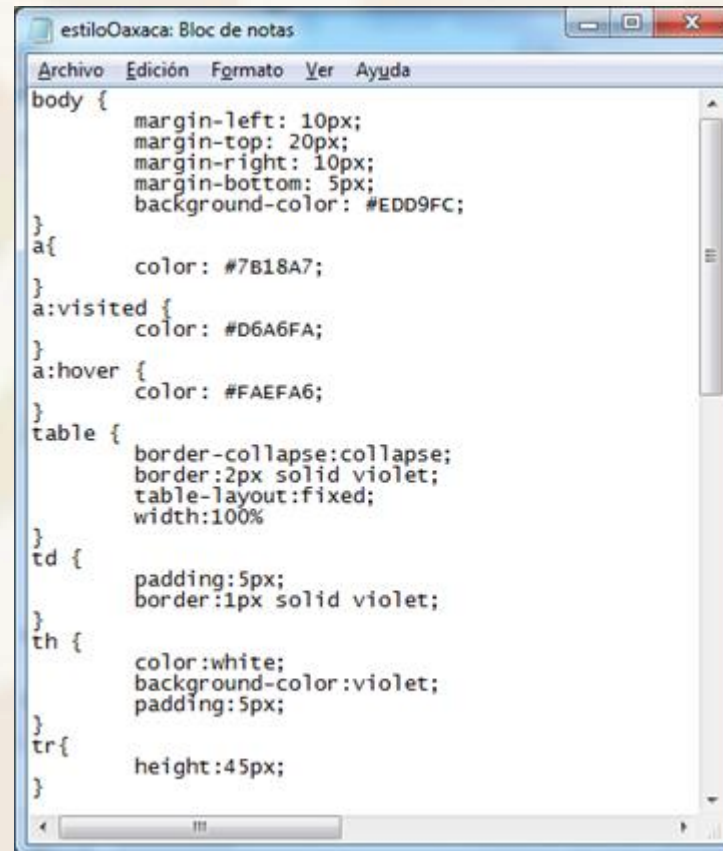
2.1.5. Creación.

Se puede crear una hoja de estilos en un archivo independiente y posteriormente adjuntarla al sitio web y modificarla para ajustarla a las necesidades de la página o crear una nueva con características propias.

Ejemplo: si cuentas con un equipo de cómputo, deberás ir creando la hoja de estilos de acuerdo a los siguientes pasos:

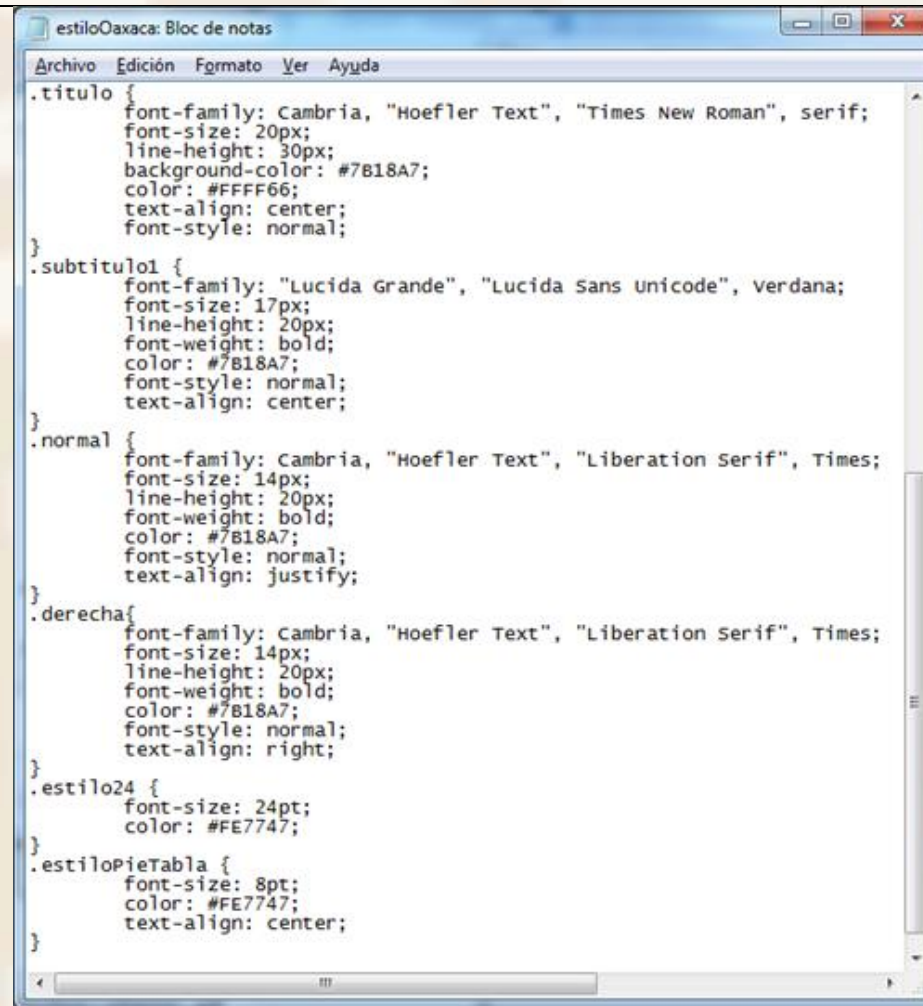
1. Crear una carpeta con el nombre: estilos, dentro de la carpeta del sitio de trabajo: ejerWeb ubicada en la unidad C.
2. Escribir en un único archivo de block de notas, el código de las figuras 1.32 y 1.33. Guardarlo en la carpeta: estilos con el nombre de: estiloOaxaca.CSS. Recuerda que, al grabar un archivo de este

tipo, se debe seleccionar: Todos los archivos, en lugar del tipo: archivos de texto.



```
estiloOaxaca: Bloc de notas
Archivo  Edición  Formato  Ver  Ayuda
body {
    margin-left: 10px;
    margin-top: 20px;
    margin-right: 10px;
    margin-bottom: 5px;
    background-color: #EDD9FC;
}
a {
    color: #7B18A7;
}
a:visited {
    color: #D6A6FA;
}
a:hover {
    color: #FAEFA6;
}
table {
    border-collapse: collapse;
    border: 2px solid violet;
    table-layout: fixed;
    width: 100%;
}
td {
    padding: 5px;
    border: 1px solid violet;
}
th {
    color: white;
    background-color: violet;
    padding: 5px;
}
tr {
    height: 45px;
}
```

Figura 1.32 Archivo: estiloOaxaca.CSS parte 1.



```
.titulo {
font-family: Cambria, "Hoefler Text", "Times New Roman", serif;
font-size: 20px;
line-height: 30px;
background-color: #7B18A7;
color: #FFFF66;
text-align: center;
font-style: normal;
}
.subtitulo1 {
font-family: "Lucida Grande", "Lucida Sans Unicode", Verdana;
font-size: 17px;
line-height: 20px;
font-weight: bold;
color: #7B18A7;
font-style: normal;
text-align: center;
}
.normal {
font-family: Cambria, "Hoefler Text", "Liberation Serif", Times;
font-size: 14px;
line-height: 20px;
font-weight: bold;
color: #7B18A7;
font-style: normal;
text-align: justify;
}
.derecha {
font-family: Cambria, "Hoefler Text", "Liberation Serif", Times;
font-size: 14px;
line-height: 20px;
font-weight: bold;
color: #7B18A7;
font-style: normal;
text-align: right;
}
.estilo24 {
font-size: 24pt;
color: #FE7747;
}
.estiloPieTabla {
font-size: 8pt;
color: #FE7747;
text-align: center;
}
```

Figura 1.33 Archivo: estiloOaxaca.CSS parte 2.

3. Crear el archivo: lenguasIndigenasConHojaDeEstilos.html, con el código mostrado en la figura 1.34, que deberá estar guardado en la carpeta ejerWeb, creada en la unidad C.

```

lenguasIndigenasConHojaDeEstilos.html: Bloc de notas
Archivo Edición Formato Ver Ayuda
<html>
  <head>
    <title>Lenguas indígenas en Oaxaca</title>
    <link href="estilos/estiloOaxaca.css" rel="stylesheet" type="text/css">
  </head>
  <body>
    <table width="80" border="2" align="center" summary="Información del censo de Población y vivienda 2010.">
      <caption class="estilo24">
        Las lenguas indígenas más habladas en el estado de Oaxaca son:
      </caption>
      <tr>
        <th width="40" bgcolor="#D6A6FA" class="titulo" scope="col">Lengua indígena</th>
        <th width="40" bgcolor="#D6A6FA" class="titulo" scope="col">Número de hablantes (año 2010)</th>
      </tr>
      <tr>
        <td><span class="normal">Lenguas zapotecas</span></td>
        <td class="derecha">371 740</td>
      </tr>
      <tr>
        <td><span class="normal">Lenguas mixtecas</span></td>
        <td class="derecha">264 047</td>
      </tr>
      <tr>
        <td><span class="normal">Mazateco</span></td>
        <td class="derecha">175 970</td>
      </tr>
      <tr>
        <td><span class="normal">Mixe</span></td>
        <td class="derecha">117 935</td>
      </tr>
    </table>
    <h5 align="center" class="estiloPieTabla">FUENTE: INEGI. Centro de Población y Vivienda 2010.</h5>
    <p align="center" class="subtitulo1">En Oaxaca, hay 1 165 186 personas mayores de 5 años que hablan alguna lengua indígena, lo que representa 34% de la población de la entidad.</p>
  </body>
</html>

```

Figura 1.34. Código para crear el archivo:

lenguasIndigenasConHojaDeEstilos.html.

4. Ejecute el archivo creado y se mostrará en el navegador, la página que se observa en la figura 1.35.

Actividad 14:

Figura 1.34 con notas aclaratorias.

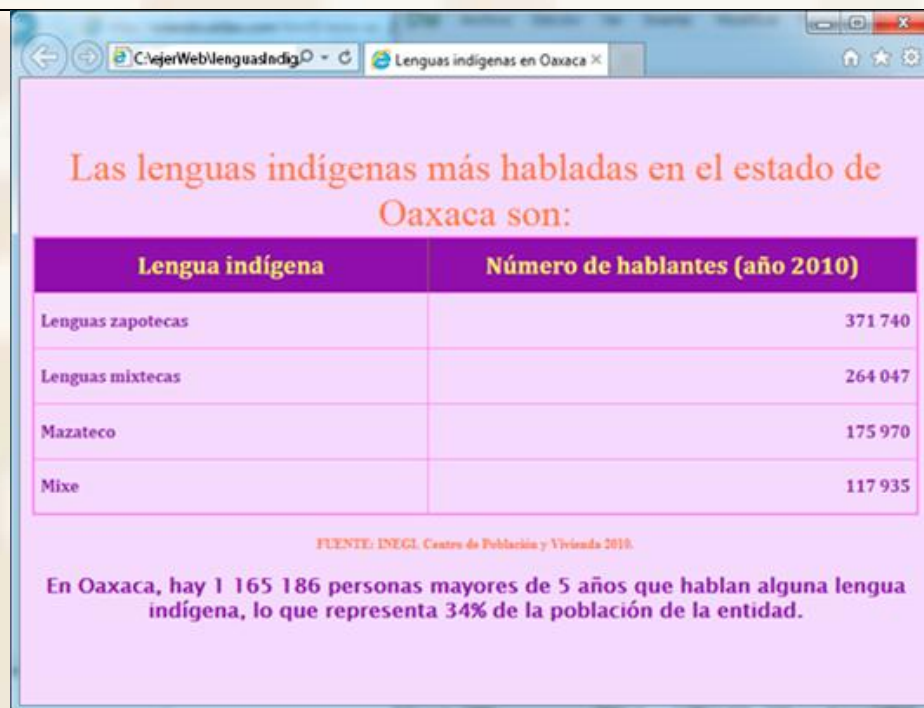
Actividad 15:

Archivo con hoja de estilos: (estiloOaxaca.CSS) creado en el cuaderno de apuntes.

Archivo de página web creado en el cuaderno de apuntes: lenguasIndigenasConHojaDeEstilos.html

Actividad 16:

Hojas de estilo y página web con hoja de estilo creadas en el cuaderno.



Lengua indígena	Número de hablantes (año 2010)
Lenguas zapotecas	371 740
Lenguas mixtecas	264 047
Mazateco	175 970
Mixe	117 935

FUENTE: INEGI, Censos de Población y Vivienda 2010.

En Oaxaca, hay 1 165 186 personas mayores de 5 años que hablan alguna lengua indígena, lo que representa 34% de la población de la entidad.

Figura 1.35. Página lenguasIndigenasConHojaDeEstilos.html en ejecución.

Actividad 14: identifique los elementos referentes a la hoja de estilos, en el archivo de la figura 1.34, mediante notas aclaratorias, que indiquen a lo que se refieren. Puedes emplear pequeños “post” con recortes de papel o papel auto adherible para las notas aclaratorias, así como remarcar con marca textos para una fácil identificación.

Actividad 15: Realizar el ejemplo propuesto, siguiendo cada uno de los pasos y guiándose de las figuras 1.32 a la 1.35.

Actividad 16: con base en el ejemplo propuesto correspondiente al estado de Oaxaca, presenta datos de acuerdo a tu estado. Puedes presentar datos de los hablantes en cada una de las regiones, lenguas, vestimenta, comidas, flora, fauna,

	entre otros que tú decidas. Crea una hoja de estilos con los colores, fuentes y otros elementos de tu preferencia y aplícalo a tu página web. Recuerda que, si no cuentas con un equipo de cómputo, lo puedes realizar todo en tu cuaderno de apuntes, de manera organizada, incluyendo letreros y notas aclaratorias. Puedes incluir recortes para las imágenes o ¡dibujarlas tú mismo! Lo importante es que reconozcas y aprendas cómo se crean y aplican las hojas de estilos en las páginas web para que puedas llegar a practicarlas en un futuro, ¡presentando excelentes diseños creativos y con calidad! Nota: la información, figuras y ejemplos, fueron tomados del libro: Módulo III. Desarrolla aplicaciones web y móviles. Autora: Blanca Elia Jiménez Guzmán. Editorial: FCE, SEP, DGETI. Primera edición. 2016. México.	
Aprendizajes esenciales o Competencias esenciales 3er parcial	Estrategias de Aprendizaje	Productos a Evaluar
Situación 2.2: Creando una página web responsiva. Contenido: 2.2.1. Responsive Web Design (Diseño web responsivo). 2.2.2. Elementos. 2.2.3. Importancia. 2.2.4. Concepto mobile first. 2.2.5. Creación.	Con ayuda de la siguiente información, Realiza un "mapa mental" de los elementos básicos de un Diseño web responsivo. Elementos del diseño web responsivo: Los elementos centrales de un diseño adaptable o "responsivo" son al menos cuatro: 1. Esquema typographic flexible. 2. Maquetación adaptable usando Media Queries, modificando la cantidad de columnas del diseño, con grillas flexibles, adaptando los anchos dentro de cada rango. 3. Imágenes y multimedia adaptables, cambiando el tamaño de las imágenes y otros elementos vinculados. No debemos olvidar, si el sitio los incluye, la adaptación de videos y animaciones, y demás contenidos complejos como mapas, tablas, slides, etc. 4. Navegación adaptable, optimizada para touch en pantallas pequeñas. Complementariamente, aplicaremos sistemáticamente en todas nuestras páginas adaptables, dos elementos extra: a. Una etiqueta meta viewport que impida la "miniaturización" de nuestras páginas. b. Y un script compatibilizador para que funcione en navegadores antiguos la técnica de media queries.	• Mapa mental: Elementos del Diseño web responsivo.

Comencemos a analizar uno por uno estos elementos, así aprenderemos a incluirlos en nuestros proyectos, para que podamos volverlos adaptables y dejemos atrás la era de la rigidez.

1. ESQUEMA TIPOGRÁFICO FLEXIBLE:

El primer elemento del diseño que volveremos flexible desde nuestra hoja de estilos será el texto. La novedad será que cambiaremos la unidad de medida más utilizada hasta el momento para definir el tamaño de las fuentes mediante la propiedad font-size (es decir, los píxeles), por otras dos unidades de medida más versátiles.

Si continuamos utilizando píxeles como unidad de medida para nuestras fuentes, tendremos dos tipos de problemas:

1. El primero se lo causaremos a algunos de nuestros usuarios, aquellos que utilicen Internet Explorer, ya que no podrán escalar el tamaño de la fuente si ésta fue definida en píxeles (un problema de accesibilidad que afectará a nuestros usuarios miopes).
2. Y el segundo, será un problema de mantenimiento para nosotros mismos, ya que deberíamos duplicar, triplicar o más, las declaraciones de tamaños de fuentes dentro de cada zona de la hoja de estilos, para poder definir cuál debe ser su tamaño en cada rango o condición, es decir, en cada media query. Entenderemos este problema a continuación.

Texto primero

¿Por dónde empezamos entonces el proceso de codificación (HTML) si queremos aplicar Media Queries? Por el principio: el texto.

Pregunta al paso que siempre me gusta plantear a mis alumnos: ¿Cuál es el denominador común de una pizza? ¿Aceitunas? ¿Cebollas? ¿Tomates? ¿O la masa?

¿Y el denominador común de una Web? ¿Flash? ¿Imágenes? ¿Videos? ¿O el texto?

Una vez marcados los textos semánticamente (con h1, h2, p, ul con li), aplicamos los identificadores y clases que creamos necesarios, y ya tendremos el código HTML básico terminado. Ese texto bien estructurado con HTML elemental, será suficiente para cualquier celular de bajas prestaciones que aún pudiera estar utilizando algún usuario.

Tipografía, siempre en EM

Ahora sí, llegó el momento de definir en nuestra hoja de estilos las unidades de medida que volverán flexible nuestro esquema tipográfico, para garantizar la legibilidad de nuestros contenidos. Pensemos que la distancia de lectura en una PC es cercana al metro, mucho mayor a la que existe entre nuestros ojos y un teléfono o tableta que sostenemos en nuestras manos.

Por ese motivo, los tamaños de fuentes deberán ser mayores para una PC que para una tableta, y los de una tableta, mayores que los de un celular.

Para lograrlo, no usaremos más píxeles para definir el tamaño de fuentes, sino que usaremos la unidad de medida em y los porcentajes, combinados de una

manera particular: al body (y solamente al body) le definiremos un font-size base en porcentaje, y al resto de textos, se lo definiremos en em:

```
body {
font-size:80%;
}
```

```
/* porcentaje base, solo en el body */
```

```
p {
font-size:0.9em;
}
```

```
h1 {
font-size:2em;
}
```

```
h2 {
font-size:1.4em;
}
```

```
#pie {
font-size:1.2em;
}
```

```
.epigrafes {
font-size:1.1em;
}
```

```
/* fin de zona común a todas las resoluciones */
```

```
@media screen and (min-width:800px) {
body {
font-size:100%;
/* ampliamos los textos si mide más de 800px */
}
}
```

```
/* fin de la zona para más de 800px de ancho de pantalla */
```

```
@media screen and (min-width:1200px) {
body {
font-size:120%;
/* ampliamos más aún los textos si mide más de 1200px */
}
}
```

Observemos que hemos definido en em los tamaños de fuentes de nuestros contenidos usando indistintamente etiquetas, id o clases, en un solo lugar: la zona

inicial de nuestra hoja de estilos, aquella que leerán todos los dispositivos sin condiciones, ya que no se encuentra envuelta en ninguna media query. Por otro lado, cuando aplicamos una condición para cierto tamaño, lo único que cambiamos es el valor base en porcentaje aplicado al body, lo que hará que todo el esquema tipográfico definido en ems se escale proporcionalmente a un nuevo tamaño. Es decir, lo que estamos haciendo es cambiar el tamaño del em, al cambiar su punto de referencia (que es ese porcentaje que definimos en el body).

Como orientación, podemos calcular que en la mayoría de las pantallas de PC, a tamaño de fuente normal, la equivalencia entre ems y pixeles es que 1em = 16px, por lo que, si queremos definir en la hoja de estilos un texto que en Photoshop ocupaba 24px, lo dividiremos por 16 para saber su valor en em, que en este caso sería 1.5em. Atención: utilicemos puntos como separador decimal, no comas.

Por supuesto, nada impide que realicemos ajustes en alguna de las media queries si queremos que determinado texto tenga una medida especial en una de ellas. Pero el punto de partida ya quedará establecido automáticamente.

Con estas nuevas unidades de medida ya podremos crear esquemas tipográficos adaptables, que se visualicen de manera óptima según la distancia de lectura de cada dispositivo.

Notemos que no estamos definiendo un tamaño rígido, sino una relación proporcional entre los distintos tamaños de texto de nuestra página, proporción que se mantendrá a lo largo de todos los dispositivos gracias a que vamos a escalar el esquema completo en cada media query.

2. MAQUETACIÓN ADAPTABLE Y GRILLA FLEXIBLE:

Aprovechando que cada zona de la hoja de estilos que envolvamos entre media queries permite que ciertos estilos se apliquen solo en un rango de tamaños de pantalla, en cada zona acomodaremos los contenidos en columnas de una manera optimizada para el tamaño del dispositivo.

- “Flotar o no flotar, that is the question”

En principio, convengamos que en un celular básico no es suficiente el espacio para flotar dos o más columnas una al lado de la otra, por lo que el layout será extremadamente simple: solo dejaremos que los bloques se apilen uno debajo del otro por orden de aparición en el código HTML.

A partir de allí, haremos flotar tantos bloques como creamos necesario en cada media query.

Grillas flexibles

La grilla flexible consiste en definir anchos de contenedor y de columnas en porcentajes, para que los bloques de un diseño mantengan una proporción entre sí dentro del rango de ancho mínimo y máximo definido en cada media query del punto anterior.

La fórmula para calcular los porcentajes a partir de un boceto en pixeles es la de dividir el ancho del elemento por el del que lo envuelve:

Tamaño Elemento / Tamaño Contexto

Ejemplo: 600px / 960px = 0,625

Es decir, ese bloque que medía 600px dentro de un contenedor de 960px ahora deberá medir 62.5% (ese valor surge del 0.625 de la cuenta que acabamos de realizar)

Repetamos la fórmula en otro caso:

Tamaño Elemento / Tamaño Contexto

Ejemplo de 3 columnas para fotos ubicadas dentro de una zona de 480px:

150px / 480px = 0,3125

Es decir, deberemos definir un 31.25% de ancho a cada una de las 3 columnas.

Márgenes y paddings flexibles

En este contexto, también debemos volver flexibles los márgenes y paddings, para que no arruinen con pixeles la proporcionalidad de los espacios conseguida.

1. Margen: el contexto es el ancho del elemento padre (contenedor):
 - o Ej. 24px / 900px = 0,02666 = 2,66%
2. Padding: el contexto es el ancho del elemento mismo al que lo aplico:
 - o Ej. 24px / 200px = 0,12 = 12%

3. MEDIOS ADAPTABLES (IMÁGENES, VIDEOS):

Si en cada zona de los estilos CSS delimitada por una media querie apuntamos a distintas imágenes (de distintos tamaños), no tendremos problemas con background-image:

```
/* La imagen más pequeña va primero, sin condiciones, porque es para el
celular más chico */ .logo { background-image: url(logo-chico.jpg); } /*
Logo mediano, para anchos de pantalla de entre 480 y 800px */ @media
screen and (min-width:480px) { .logo { background-image:
url(logo-mediano.jpg); } } /* Logo grande, para anchos de pantalla
de entre 800 y 1280px */ @media screen and (min-width:800px) { .logo
{ background-image: url(logo-grande.jpg); } } /* Logo
gigante, para anchos de pantalla de más de 1280px */ @media screen and
(min-width:1280px) { .logo { background-image: url(logo-
gigante.jpg); } }
```

en el caso de etiquetas IMG, haremos flexible la imagen dentro del rango mínimo y máximo de un layout:

```
img, video, object { max-width:100%; }
```

Eso hará que dentro del rango de ancho del elemento que envuelva a la foto (columna) la imagen se estire desde un mínimo y hasta un máximo. Como el máximo es su tamaño real, 100%, consideremos ese ancho al definir el ancho de

su elemento contenedor, o viceversa: creamos la imagen al tamaño máximo al que llegará su elemento contenedor.

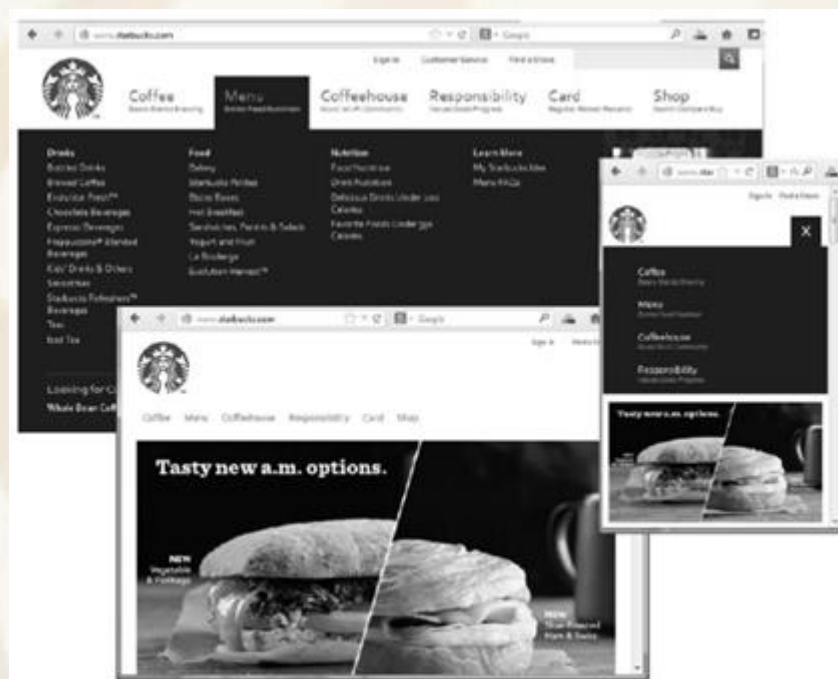
Existen varias propuestas del W3C para especificar que el navegador descargue distintas imágenes según el tamaño de pantalla (tal como en las imágenes de fondo) como por ejemplo, la posible etiqueta "picture" con varias etiquetas "source", cada una condicionada por una media query, o que se modifique la etiqueta "img" permitiendo especificar más de un source mediante el atributo srcset. Al momento de escribir este libro ninguna de ellas está estandarizada ni implementada por los navegadores.

Otros casos son los de utilizar dibujos vectoriales escalables SVG en vez de una etiqueta IMG cuando lo que muestre la imagen sea un dibujo con pocos colores (logotipos, iconos, etc.).

4. NAVEGACIÓN ADAPTABLE:

La navegación es otro de los puntos críticos que debe adaptarse en un sitio para permitir su uso en teléfonos y tabletas.

Tomemos como ejemplo la navegación de este sitio:



En el caso de la PC, vemos una serie de submenús desplegables, a partir de una lista (ul) de primer nivel que contiene el nombre de cada sección en letras grandes, y con una segunda línea descriptiva. Vemos cómo, en un tamaño

mediano de pantalla (tableta), esa segunda línea desaparece, y el tamaño de fuente es menor. Ese cambio puede lograrse con algo tan sencillo como cambiar el font-size (ya lo vimos en el primer punto de este capítulo) y ocultar con display:none los subtítulos por defecto, haciéndolos visibles con display:block a partir de cierto tamaño de pantalla.

Por ejemplo:

```
.subtitulos {
    display:none;
}
/* Hacemos visibles los subtítulos para anchos de pantalla mayores a 1024px
*/@media screen and (min-width:1024px) {
    subtitulos {
        display:block;
    }
}
```

Y en el caso de la navegación en la pantalla más pequeña (teléfono), vemos que se aplicó el mismo criterio pero para ocultar el enorme listado de submenús, dejando solamente los botones de primer nivel, haciendo que no floten uno al lado del otro y dándoles un tamaño grande, para que puedan ser fácilmente pulsados con los dedos en una pantalla táctil.

Otro caso bastante común es el de reemplazar un menú visualmente muy amplio que es el que se usará en la PC, por un select con options en el teléfono.

En el código HTML, que es el mismo para todos los dispositivos, estará el código HTML de ambos menús (el formulario con el select y la "ul" con botones).

Por ejemplo:

```
<ul id="menupc">
    <li>HOME</li>
    <li>CONTACTO</li>
</ul>
<form id="menumovil">
    <select>...etc... </select>
</form>
```

Desde la hoja de estilos, en la zona inicial de la hoja sin condiciones, ocultamos el menú de PC y mostramos el de celular:

```
#menupc {
    display:none;
}
#menumovil {
    display:block;
}
```

Y en una media query, invertiremos esto para cuando estemos en pantallas grandes:

```
@media screen and (min-width:640px) {
```

```
#menupc {
  display:block;
  /* Mostramos el menú de PC */
}
#menumovil {
  display:none;
  /* Ocultamos el select para teléfono */
}
}
```

De esta manera, ya podemos manipular la visualización de diferentes herramientas de navegación, gracias a las media queries.

Configurando el Viewport

El tamaño de la "ventana" del navegador en una PC, siempre coincide o es menor que el tamaño de la pantalla. Es decir, o usamos el navegador maximizado, a pantalla completa, o lo achicamos un poco. Pero nunca es más grande que la pantalla.

En móviles, en cambio, o la ventana del navegador coincide con el tamaño de pantalla (siempre maximizada), o es mayor que el tamaño de pantalla, viendo solo una parte de ella por vez. Nunca es menor ya que no podemos "achicar" la ventana para que ocupe menos de una pantalla. Pero sí puede suceder lo contrario, que el contenido supere el tamaño de la pantalla porque estemos viendo solo una parte, debido a la utilización del zoom.

Por ejemplo: Safari y Opera Mini asignan 980px de ancho al viewport y hacen zoom para mostrar lo que suponen una web hecha para PC (y en el 99% de los casos, ¡aciertan!).

Veamos un ejemplo comparando la misma página sin que el navegador le haga zoom, "miniaturizándola", y con la etiqueta viewport que lo impide, que veremos a continuación:

¿Cómo podemos impedir que los navegadores móviles escalen nuestros sitios y los miniaturicen? La solución es una nueva etiqueta meta cuyo name es "viewport", que solo es aplicada por dispositivos móviles y es ignorada en computadoras de escritorio.

Su sintaxis es la siguiente:

```
<metaname="viewport" content="width=device-width,initial-scale=1.0" />
```

Dentro del atributo content, lo que estamos haciendo es definir que el width de la ventana del navegador tenga un valor "real"; es decir, que el navegador no nos "mienta", sino que utilice como valor de su propiedad "width" el valor de otra propiedad, que es "device-width", es decir, el ancho físico de su pantalla. De esa manera, un navegador dentro de un dispositivo de por ejemplo, 320px de ancho, ya no declarará un width ficticio de 980px, sino que lo dejará en 320px, que es el ancho del dispositivo, con lo cual no miniaturizará nuestro diseño.

Por otro lado, notemos que hemos definido un valor inicial para el zoom, mediante la propiedad initial-scale a un valor de "1" que equivale a 100%, es decir, el tamaño natural. De esta manera, cuando el navegador ingrese a nuestra página, no aplicará ningún nivel de zoom que previamente el usuario hubiera configurado.

- Atención con el zoom: no debemos desactivar por completo la posibilidad de realizar zoom por parte del usuario, ya que ésta es una atribución suya. Pensemos, por ejemplo, en un usuario miope, que necesita ampliar parte de nuestro sitio. Para eso, evitemos definir en la etiqueta viewport los valores de minimum-scale y de maximum-scale, lamentablemente muy difundidos.

Soluciones para navegadores que no entienden Media Queries

Los celulares más antiguos, cuyos navegadores no entienden media queries, no tendrán problemas en mostrar un diseño acorde a su tamaño si ese diseño es el primero en la hoja de estilos y no está envuelto dentro de ninguna condición. Los navegadores móviles más modernos procesan media queries, así que tampoco son un problema.

Las tabletas son dispositivos nuevos, creados a partir del año 2009, con lo cual todos sus navegadores soportan media queries.

El único problema de compatibilidad lo tendremos en PCs de escritorio. Los navegadores de PC más modernos tienen soporte para media queries, pero algunos navegadores antiguos como Explorer 8 todavía subsisten, y no interpretan las media queries.

Simplemente vinculamos nuestra página HTML a ese script con una etiqueta `<script>` y a partir de ese momento el Explorer 8 interpretará las media queries.

A manera de conclusión, vimos que es perfectamente posible crear experiencias de navegación por sitios web multidispositivo, es decir, que se puedan usar con un diseño y unas herramientas optimizadas para diferentes tamaños de pantalla, si aprovechamos la técnica de media queries para crear esquemas tipográficos adaptables, layouts y grillas flexibles, elementos visuales como imágenes y videos líquidos, y herramientas de navegación optimizadas para el uso en pantallas táctiles.

Delgado, Hugo. (2018). Diseño Web Responsive - Tutorial con ejemplos adaptables. Recuperado 24 de enero, 2021, de <https://disenowebakus.net/diseño-web-responsive.php>

2. Contesta correctamente el siguiente cuestionario tomando de referencia la información aquí proporcionada.

1.- Explica que es una página web responsiva.

	2.- Explique Por qué en ocasiones se confunde responsive con una página web móvil. 3.- Menciona tres recomendaciones para tener una página web responsiva. 4.- Menciona cinco dispositivos que los mexicanos utilizan para conectarse a internet actualmente. 5.- Que página web mostrara el navegador Google, una responsiva o una no responsiva. Explique él porque. 6.- Seguramente habrás entrado en alguna web desde tu teléfono y has tenido que ir ampliando y alejando la imagen, moviéndote por la pantalla. ¿Eso quiere decir que esa página es responsive o no es responsive?. 7.- ¿Qué significa que una página web tenga un mejor SEO?. 8.- ¿En qué consiste la practica SEO?. 9.- ¿Por qué es más sencillo y económico tener una página web responsiva?. 10.- Los mexicanos cuanto invierten de tiempo en Internet a diferencia de los medios tradicionales. En primer lugar... ¿qué es exactamente un diseño web responsive? Un diseño web responsive es el que es capaz de adaptarse a pantallas de diferentes tamaños con un solo sitio web. El sistema detecta automáticamente el ancho de la pantalla y a partir de ahí adapta todos los elementos de la página, desde el tamaño de letra hasta las imágenes y los menús, para ofrecer al usuario la mejor experiencia posible. ¡Parece magia! En ocasiones, se confunde el responsive con las webs para móviles, pero no se trata de lo mismo. En el caso del diseño responsive, tenemos un solo sitio web que	
--	---	--

puede adaptarse para dispositivos de todo tipo, desde ordenadores de escritorio con grandes monitores hasta móviles, pasando por tablets y otros. En cambio, crear un sitio móvil implica diseñar desde cero una web independiente, cuyos contenidos y formato están especialmente adaptados para funcionar mejor en dispositivos móviles.

En mi opinión, la opción más recomendable es sin duda el diseño responsive, ya que crear una web para móviles implica tener dos sitios diferentes y duplicar las tareas de mantenimiento y actualización. Además, el responsive se adapta automáticamente a todo tipo de tamaños. Dicho esto, podemos usar algún plugin para crear sitios para móviles (como los de WordPress) como solución temporal mientras diseñamos un sitio realmente adaptable.

¿Por qué tener una página responsiva?

- **Es recomendada por Google™**

El algoritmo de Google fue actualizado recientemente para dar mayor ranking de búsqueda a las páginas que están elaboradas con un diseño responsive.

- **Una página, muchos dispositivos**

Permite proveer una excelente experiencia al usuario desde cualquier dispositivo y tamaño de pantalla: teléfonos móviles, tabletas, laptops y computadoras de escritorio.

- **Es más sencillo y económico**

Permite tener un mejor SEO(Search Engine Optimization) es la práctica de utilizar un rango de técnicas, incluidas la reescritura del código html, la edición de contenidos, la navegación en el site, campañas de enlaces y más acciones, con el fin de mejorar la posición de un website en los resultados de los buscadores para unos términos de búsqueda concretos. ya que se tienen las mismas URLs (direcciones) para móvil y web. Es más económica de administrar. Permite una descarga más rápida y adecuada en móviles.

Algunas cifras que debes conocer...

Fuente: iab México

- Para conectarse a Internet, en 2014, los internautas mexicanos usaron **3 dispositivos en promedio**.
- Los internautas mexicanos invierten en Internet **más del doble** del tiempo que en medios tradicionales.
- **9 de cada 10** internautas mexicanos siempre llevan consigo sus dispositivos móviles.
- **42%** no puede salir de su casa sin sus dispositivos móviles.

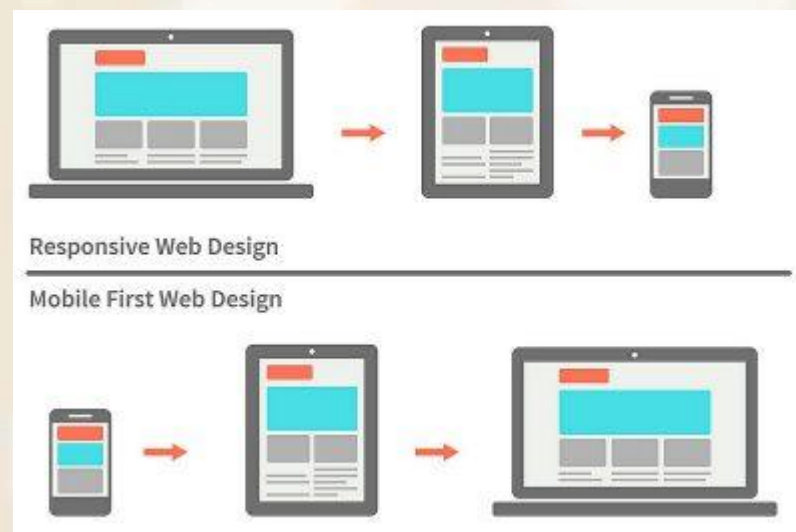
3. Con ayuda de la información que a continuación aparece, Relaciona correctamente ambas columnas:

a) Diseño dinámico.	[] los móviles primero
b) SEO.	[] Search Engine Optimization

c) Diseño 100% responsive.	[] Search Engine Results Page
d) Mobile Friendly.	[] sitio web amigable para móviles
e) SERPS.	[] consiste en un tipo de programación de código único (para una misma URL), que se adapta a cualquier tipo de pantalla y dispositivo
f) mobile first.	[] ofrecer una experiencia específica en la versión móvil, tienes que dar contenido variado y optimizado para las diferentes consultas de búsqueda.
g) Diseño web único para versión móvil.	[] disponer de dos versiones de tu web, una para móviles y otra para ordenadores de escritorio.

¿Qué es mobile first?

Como su propio nombre indica **mobile first** o por su traducción en inglés, «los móviles primero», resume perfectamente esta filosofía de desarrollo. Básicamente se refiere a un modo de diseñar que tenga en cuenta, en primera instancia, un dispositivo móvil. Pantallas reducidas en comparación a los monitores que usamos normalmente con los ordenadores, y tras tener la maqueta preparada, realizar un escalado, es decir, aumentar el tamaño y adaptarlo a una pantalla de escritorio.



A día de hoy, el diseño responsive, la filosofía de diseño opuesta, es un estándar. Este tipo de páginas web, son adaptativas, es decir, al reducir la resolución se reduce el tamaño del contenido, y es, realmente, lo que tiene Google bajo su lupa.

- Cuestionario: [Importancia del Diseño responsive.](#)

Que nuestro sitio web sea *responsive* no es opcional sino obligatorio, de lo contrario nuestra visibilidad y efectividad SEO se verá reducida ya que estaremos perdiendo todas las visitas de potenciales clientes que usen dispositivos móviles para navegar.

En 2015, Google ya hablaba sobre esta tendencia, y pasó a priorizar los sitios web que son *mobile friendly* (sitios web amigables para dispositivos móviles) en los resultados de las búsquedas.

Cuando hablamos sobre el algoritmo de Google en 2017, le dimos una gran importancia al desarrollo web para dispositivos móviles, ya que es un factor en SEO cada vez más importante y por ello fundamental en cualquier proyecto online que se lance.

El objetivo principal de todo esto es que un usuario desde móvil tenga la misma experiencia a la hora de navegar por nuestro sitio web que un usuario desde escritorio. Todo tiene que adaptarse, los botones, los enlaces, imágenes, etc.

Google considerará siempre, en primera estancia, la versión móvil de tu sitio, para posicionarla.

Así que significa que si la parte móvil está bien optimizada, ambas partes, saldrán beneficiadas y, ganarán posicionamiento. Si no contamos con una, no nos estaremos beneficiando de esos puestos extra que ofrece a los sitios web móviles.

Es por ello, que tener un sitio web *mobile friendly*, es decir, adaptado a dispositivos móviles, puede significar la diferencia entre un buen posicionamiento y ser top en SERPs.

Qué son las SERPS?

SERPS, significa **“Search Engine Results Page”** o en su traducción al español, **“resultados del buscador”** y estos resultados son los que aparecen entre las 10 primeras opciones de las páginas de Google, o cualquier otro buscador. La idea con las SERPs es que tu marca (página web) se ubique en los primeros lugares. De este modo, todas las tareas que hagamos por mejorar ese aspecto, estarán afectando directamente, y de forma positiva, a nuestro posicionamiento.

Vías para crear un diseño web “Mobile Friendly”

Si aún no tienes una web de diseño *mobile friendly*, a continuación conocerás las 3 vías para que así sea, de forma que mejore la experiencia de tus usuarios y Google le otorgue a tu site mejores posiciones en sus rankings:

1) Diseño 100% responsive

El diseño responsive consiste en un tipo de programación de código único (para una misma URL), que se adapta a cualquier tipo de pantalla y dispositivo.

Esta es una muy buena opción para cualquier sitio web, pues resulta relativamente fácil de mantener y actualizar. Por tanto, suele ser la vía favorita para tener contento a Google en términos de usabilidad y experiencia de usuario.

2) Diseño dinámico

	Tener un diseño dinámico significa que si quieres ofrecer una experiencia específica en la versión móvil, tienes que dar contenido variado y optimizado para las diferentes consultas de búsqueda. Por tanto, esto quiere decir que deberás distinguir las versiones de tu sitio web en el caso de que sea necesario. No implica hacer un rediseño, solamente se trata de más carga de trabajo en mantenimiento y actualización. La ventaja de este diseño mobile friendly, es que habrá una URL única para todos los dispositivos, así como una optimización específica para cada dispositivo. 3) Diseño web único para versión móvil Puedes disponer de dos versiones de tu web, una para móviles y otra para ordenadores de escritorio. La ventaja de esto es que la web para móviles tendrá una URL diferente para aquellos usuarios que entren desde sus dispositivos móviles. Por tanto, se brinda una experiencia enfocada a móviles, sin necesidad de que tengas que rediseñar el sitio principal. Conclusiones: "Diseño Mobile Friendly" vs "Diseño UX" Después de revisar todo lo anterior, te habrás dado cuenta de lo importante que es ponerse en la piel del navegante, pues aspectos como el tamaño de la letra o el espaciado entre los botones determinarán la buena o mala experiencia del usuario. En definitiva, los conceptos "UX" y "Mobile Friendly" van de la mano, y trabajan bajo diseños sencillos y comprensibles que facilitan la navegación en el menor número de pasos. De lo contrario, tus visitantes se marcharán de tu web y no volverán más.	• Relacionar: <u>Concepto mobilefirst y creación.</u>
--	---	---

Aprendizajes esenciales			
Carrera:	Programación	Semestre:	Cuarto
Módulo/Submódulo:	Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.		
Aprendizajes esenciales o Competencias esenciales 1er parcial	Estrategias de Aprendizaje	Productos a Evaluar	
Competencia Profesional El alumno Usa JavaScript para manejar eventos Contenido: 1.1 Qué es JavaScript 1.2 Características y funcionamiento del lenguaje JavaScript 1.3 Clientes y Navegadores 1.4 URL y Recurso 1.5 Introducción a JavaScript (Tipos, objetos y valores) 1.6 Operadores y expresiones 1.7 Sobrecarga de operadores 1.8 Conversión de tipos en expresiones 1.9 Operador typeof 1.10 Operadores JavaScript 1.11 Comentarios	a. Resuelve la evaluación diagnóstica (Anexo 1 pág. 2 y 3) b. Realiza un mapa mental con la información proporcionada en la documentación adjunta. (Anexo 1 pág. 4-12) c. Revisa la información del editor para la construcción del código o bien identifica en tu computadora el bloc de notas que incluye Windows (Anexo 1 pág. 13) d. Realiza un resumen de la Introducción a JavaScript Anexo 1 pág. 14-21) e. Realiza práctica guiada 1 con el fin de identificar la sintaxis del lenguaje JavaScript y la estructura del desarrollo web básico. Emplea HTML, CSS e Inserta un Script utilizando las etiquetas <code><script ></code> y <code></script></code> (Anexo 1 pág. 21)	1) Examen escrito en su cuaderno. 2) Mapa Mental en su cuaderno 3) Elabora un informe escrito en su libreta, de la herramienta que va a utilizar para la escritura del código 4) Resumen escrito en su libreta de la Introducción al Lenguaje JavaScript 5) Código de la Práctica guiada escrita en libreta.	

1.12 Práctica Guiada Aplicación
WEB utilizando HTML, CSS y
JavaScript

ANEXO 1

Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.



ANEXO 1 **Parcial 1** **Módulo III. Desarrolla aplicaciones Web.**
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Evaluación Diagnostica JavaScript

1- ¿En qué lugar se ejecuta generalmente el código JavaScript?

- ☐ Servidor
- ☐ Cliente (en el propio navegador de internet)

2- ¿Cuáles de estas son marcas para la inserción del código JavaScript en las páginas HTML?

- ☐ `<javascript _code > y </javascript_code >`
- ☐ `<script > y </script >`
- ☐ `<?script > y <script? >`

3- La llamada al código Javascript debe colocarse en:

- ☐ La sección Body de la página
- ☐ Antes de la etiqueta HTML
- ☐ Puede colocarse en la sección Head o en Body

ANEXO 1

Parcial 1

Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Evaluación Diagnostica JavaScript

4- En JavaScript, para darle el nombre a una variable, objeto o función, debemos tener en cuenta que:

- ☐ No se pueden usar mayúsculas
- ☐ JavaScript no distingue entre mayúsculas y minúsculas
- ☐ JavaScript diferencia entre mayúsculas y minúsculas

5-¿Cuál es la instrucción usada para devolver un valor en una función de JavaScript?

- ☐ Send
- ☐ Return
- ☐ Value

ANEXO 1 **Parcial 1** **Módulo III. Desarrolla aplicaciones Web.**
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.



Es uno de los lenguajes de programación más importantes y según datos, **lo utilizan un 80% de los desarrolladores y un 95% de todos los sitios web**. Las ventajas de [JavaScript](#) se sitúan en el **lado del front-end**, y varios frameworks que soporta como React y Angular JS tienen un gran potencial para mejorar la experiencia del usuario en la web. **Se trata en definitiva de un lenguaje ligero, multiplataforma, estructurado y orientado a objetos y eventos.**

- Es un lenguaje de programación **seguro y fiable**.
- De **fácil uso y muy completo**.
- **Es ligero** y permite la elaboración de múltiples aplicaciones web.
- Es compatible con la mayoría de navegadores.

ANEXO 1

Parcial 1

Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Brendan Eich

Creador del lenguaje
JavaScript



A Brendan cuando trabajaba en la empresa **Netscape** en 1995, le encomendaron crear un nuevo lenguaje de programación en 10 días porque tenía que salir con la nueva versión del navegador Netscape.

El objetivo de dicho lenguaje era que las páginas web pudieran aprovechar el poder de procesamiento de los ordenadores para poder hacer la navegación por Internet más rápida.

Información personal	
Nacimiento	1961 Pittsburgh, Pensilvania, EE.UU.
Nacionalidad	Estadounidense
Educación	
Educado en	Universidad de Illinois en Urbana-Champaign Universidad Santa Clara
Información profesional	
Ocupación	Informático teórico, programador y director de tecnología
Área	Programador
Conocido por	JavaScript
Cargos ocupados	Director de tecnología de Corporación Mozilla (2005-2014) Director ejecutivo de Corporación Mozilla (2014) Director ejecutivo de Brave Software (desde 2015)
Empleador	Corporación Mozilla
Obras notables	JavaScript



ANEXO 1

Parcial 1

Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.



JavaScript es un robusto lenguaje de programación que se puede aplicar a un documento HTML y usarse para crear interactividad dinámica en los sitios web.

Fue inventado por Brendan Eich, cofundador del proyecto Mozilla

Puedes hacer casi cualquier cosa con JavaScript. Puedes empezar con pequeñas cosas como carruseles, galerías de imágenes, diseños fluctuantes, y respuestas a las pulsaciones de botones. Con más experiencia, serás capaz de crear juegos, animaciones 2D y gráficos 3D, aplicaciones integradas basadas en bases de datos ¡y mucho más!

ANEXO 1

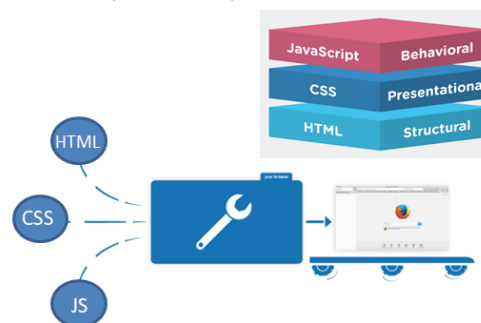
Parcial 1

Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.



Cuando cargas una página web en tu navegador, estás ejecutando tu código (HTML, CSS y JavaScript) dentro de un entorno de ejecución (la pestaña del navegador). Esto es como una fábrica que toma materias primas (el código) y genera un producto (la página web).



JavaScript MODIFICA dinámicamente HTML y CSS para actualizar la interfaz de usuario

ANEXO 1 **Parcial 1** Módulo III. Desarrolla aplicaciones Web Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente

JavaScript

- ◆ Es un lenguaje de programación de lado cliente. El término «lado cliente» significa que se ejecuta en nuestro navegador web sin necesidad de un servidor web.
- ◆ Es una puerta de acceso a otras tecnologías como AJAX, jQuery, node.js. Y un requerimiento indispensable para cualquier diseñador web que quiera hacer mejores webs
- ◆ Junto con html y css es uno de los 3 pilares del diseño web.



ANEXO 1

Parcial 1

Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Clientes y navegadores



◆ Clientes de acceso a Internet más importantes

- PCs, portátiles, tabletas, teléfonos inteligentes

◆ Navegador (browser) cliente Web de acceso a servidores

- Utilizando: **URL, HTTP, HTML, CSS y JS**
 - ♦ p.e. Chrome, Firefox, Internet Explorer, Opera, Safari, ...



◆ Tiendas de aplicaciones

- Instalan aplicaciones en móviles y tabletas
 - ♦ Las aplicaciones usan las normas de la Web (URL, HTTP,)

ANEXO 1

Parcial 1

Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Un **cliente** es un ordenador o software que accede a un servidor y recupera servicios especiales o datos de él.

Función principal del cliente:

- Estandarizar las solicitudes
- Transmitirlas al servidor
- Procesar los datos obtenidos para que puedan visualizarse en un dispositivo de salida como una pantalla.

Un cliente no ejecuta tareas de servidor, el cliente es un elemento intermedio.

- Los clientes que usas de forma cotidiana son los navegadores web y los clientes de correo electrónico.

ANEXO 1 **Parcial 1** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Ejemplos de Clientes que se utilizan en diversos ámbitos cotidianamente:

- ❖ **Sistemas operativos:** acceso al servidor a través de la línea de comandos utilizando el protocolo Telnet
- ❖ **Navegador web:** La comunicación entre el servidor y el navegador se realiza a través del protocolo [HTTP](#). El navegador finalmente evalúa los documentos [HTML](#) recibidos o las aplicaciones [JavaScript](#).
- ❖ **Clientes de correo electrónico:** Incluso la recuperación de correos electrónicos desde un servidor se realiza con una función de cliente. Los protocolos comunes son POP3, SMTP o IMAP.
- ❖ **MMPORG:** En los juegos de rol online, el software instalado actúa como un cliente, que recupera y proporciona la información necesaria para el juego desde un servidor.
- ❖ **Clientes ligeros:** Se utilizan para aplicaciones que se ejecutan sólo en un servidor y requieren un hardware mínimo. Las soluciones en la nube serían un ejemplo de ello.
- ❖ **Clientes DNS:** Estos clientes trabajan automáticamente en segundo plano y obtienen la dirección IP apropiada a una URL desde el servidor DNS apropiado.
- ❖ **Aplicaciones basadas en la web:** como herramientas de análisis web como [Google Analytics](#), trabajando con clientes.
- ❖ **Clientes VPN:** Estos clientes establecen una conexión segura entre un servidor y un PC a través de una VPN (Virtual Private Network).

ANEXO 1

Parcial 1

Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente

URL y Recurso



◆ URL (Uniform Resource Locator)

- Inicialmente se diseñó como dirección de un recurso (página Web)
 - ♦ Se generalizó como dirección de acceso a un servicio o recurso en Internet

◆ Recurso

- Contenido digital de interés para un usuario
 - ♦ página Web, foto, película, fichero o parte de él, registro de una BD,

◆ URL tiene 3 componentes básicos

- **protocolo:** protocolo de acceso (**http**)
- **servidor:** dirección del servidor en Internet (**google.com**)
- **camino:** identificador del fichero en servidor (**/picture.com**)

ANEXO 1

Parcial 1

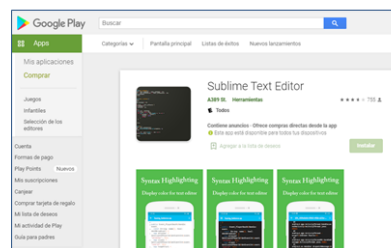
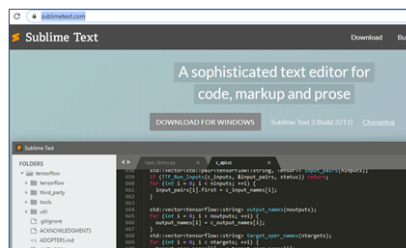
Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Equipos y Software a utilizar

- PC o Dispositivo móvil (Celular o tablet)
- Editor de código Sublime-Text <https://www.sublimetext.com/>
- O aplicación para el celular Sublime Text Editor

- ✓ En caso de contar con una señal inestable de internet, puedes utilizar el editor bloc de notas incluido en windows y tener instalado en tu pc o en tu celular un navegador como Chrome o Mozilla
- ✓ Las aplicaciones no requieren forzosamente la conectividad permanente con internet



ANEXO 1 **Parcial 1** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Tipos, objetos y valores

◆ Tipos de JavaScript

- **number**
 - ♦ Literales de números: 32, 1000, 3.8
- **boolean**
 - ♦ Los literales son los valores **true** y **false**
- **string**
 - ♦ Los literales de string son caracteres delimitados entre comillas o apóstrofes
 - "Hola, que tal", 'Hola, que tal',
 - ♦ Internacionalización con Unicode: 'Γεια σου, ίσως', '嗨，你好吗'
- **undefined**
 - ♦ **undefined**: representa indefinido



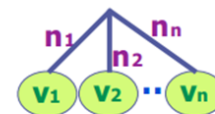
FALSE
true



UNDEFINED

◆ Objetos: agregaciones estructuradas de valores

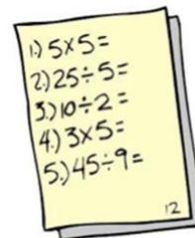
- Se agrupan en **clases**: Object, Array, Date, ...
 - ♦ Objeto **null**: valor especial que representa objeto nulo



Introducción a JavaScript

ANEXO 1 **Parcial 1** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente

Operadores y expresiones



- ◆ JavaScript incluye **operadores** de tipos y objetos
 - Los **operadores** permiten formar **expresiones**
 - ♦ Componiendo **valores** con los operadores
 - Que Javascript evalua hasta obtener un resultado
- ◆ Por ejemplo, con las operaciones aritmeticas +, -, *, /
 - podemos formar expresiones numéricas
 - ♦ Expresiones con sintaxis erronea abortan la ejecución del programa

<code>13 + 7</code>	<code>=> 20</code>	<code>// Suma de números</code>
<code>13 - 7</code>	<code>=> 6</code>	<code>// Resta de números</code>
<code>(8*2 - 4)/3</code>	<code>=> 4</code>	<code>// Expresión con paréntesis</code>
<code>8 / * 3</code>	<code>=> Error_de_ejecución</code>	<code>// Ejecución se interrumpe</code>
<code>8 3</code>	<code>=> Error_de_ejecución</code>	<code>// Ejecución se interrumpe</code>

ANEXO 1

Parcial 1

Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente

Sobrecarga de operadores

- ◆ Algunos operadores tienen varias semánticas diferentes
- ◆ Por ejemplo, el operador **+** tiene 3 semánticas diferentes
 - **Suma de enteros** (operador binario)
 - **Signo de un número** (operador unitario)
 - **Concatenación de strings** (operador binario)



`13 + 7` `=> 20` // Suma de números

`+13` `=> 13` // Signo de un número

`"Hola " + "Pepe"` `=> "Hola Pepe"` // Concatenación de strings



ANEXO 1

Parcial 1

Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Conversión de tipos en expresiones

◆ JavaScript realiza conversión automática de tipos

- cuando hay ambigüedad en una expresión
 - utiliza las prioridades para resolver la ambigüedad

◆ La expresión `"13" + 7` es ambigua

- porque combina un **string** con un **number**
 - JavaScript asigna más prioridad al **operador +** de strings, convirtiendo `7` a string

◆ La expresión `+"13"` también necesita conversión automática de tipos

- El **operador +** solo está definido para **number**
 - JavaScript debe convertir el **string** `"13"` a **number** antes de aplicar operador **+**

<code>13 + 7</code>	<code>=> 20</code>
<code>"13" + "7"</code>	<code>=> "137"</code>
<code>"13" + 7</code>	<code>=> "137"</code>
<code>+"13" + 7</code>	<code>=> 20</code>

ANEXO 1 **Parcial 1** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Operador typeof

- ◆ El operador **typeof** permite conocer el tipo de un valor
 - Devuelve un string con el nombre del tipo
 - "number", "string", "boolean", "undefined", "object" y "function"

<code>typeof 7</code>	=> "number"	
<code>typeof "hola"</code>	=> "string"	
<code>typeof true</code>	=> "boolean"	FALSE
<code>typeof undefined</code>	=> "undefined"	UNDEFINED
<code>typeof null</code>	=> "object"	
<code>typeof new Date()</code>	=> "object"	
<code>typeof new Function()</code>	=> "function"	$f(x)$

ANEXO 1

Parcial 1

Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

. []	Acceso a propiedad o invocar método; índice a array
new	Crear objeto con constructor de clase
()	Invocación de función/método o agrupar expresión
++ --	Pre o post auto-incremento; pre o post auto-decremento
! ~	Negación lógica (NOT); complemento de bits
+ -	Operador unitario, números. signo positivo; signo negativo
delete	Borrar propiedad de un objeto
typeof void	Devolver tipo; valor indefinido
* / %	Números. Multiplicación; división; modulo (o resto)
+	Concatenación de string
+ -	Números. Suma; resta
<< >> >>>	Desplazamientos de bit
< <= > >=	Menor; menor o igual; mayor; mayor o igual
instanceof in	¿objeto pertenece a clase?; ¿propiedad pertenece a objeto?
== != === !==	Igualdad; desigualdad; identidad; no identidad
&	Operacion y (AND) de bits
^	Operacion ó exclusivo (XOR) de bits
	Operacion ó (OR) de bits
&&	Operación lógica y (AND)
	Operación lógica o (OR)
?:	Asignación condicional
=	Asignación de valor
OP=	Asig. con operación: += -= *= /= %= <<= >>= >>>= &= ^= =
,	Evaluación múltiple

Operadores JavaScript

+"3" + 7 => 10

"+" unitario tiene mas prioridad y se evalúa antes que "+" binario

ANEXO 1 **Parcial 1** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Comentarios

- ◆ Los comentarios son mensajes informativos
 - Deben ser claros, concisos y explicar todo lo importante del programa
 - Incluso el propio autor los necesita con el tiempo para recordar detalles del programa
- ◆ En JavaScript hay 2 tipos de comentarios
 - Monolínea: Delimitados por // y **final de línea**
 - Multilínea: Delimitados por /* y */
 - **OJO!** Los comentarios multi-línea tienen problemas con las expresiones regulares

```
/* Ejemplo de comentario multilínea que ocupa 2 líneas  
   -> al tener ambigüedades, se recomienda utilizarlos con cuidado */  
  
var x = 1; // Comentario monolínea que acaba al final de esta línea
```

ANEXO 1 **Parcial 1** **Módulo III. Desarrolla aplicaciones Web.**
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Práctica guiada 1

Aplicación Web: HTML, CSS y JavaScript

- HTML**
 - Lenguaje de marcado
- CSS**
 - Estilo la visualización
- JavaScript**
 - Lenguaje de programación



```

<!DOCTYPE html><html>
<head>
<title>Ejemplo</title>
<meta charset="UTF-8">
<style type="text/css">
  body {color: blue;}
</style>
</head>
<body>
<h3>Fecha y hora</h3>
<script type="text/javascript">
  document.write(new Date());
</script>
</body>
</html>
  
```





21

Aprendizajes esenciales o Competencias esenciales 2º parcial	Estrategias de Aprendizaje	Productos a Evaluar
Competencia Profesional El alumno Usa JavaScript para manejar eventos Contenido - Ejercicio Practica Guiada 2 Script con fecha - Variables	- Realiza práctica guiada 2 Script Fecha; con la utilización del elemento document.write() de ser posible. (Anexo 2 pág. 1) - Realiza un mapa mental con la información proporcionada en la documentación adjunta. (Anexo 2 pág. 2-19) - Realiza práctica guiada 3 Script con expresión; con la utilización del elemento Función alert() (Anexo 2 pág. 11) - Realiza práctica guiada 4 Script con conversor; con la utilización del elemento Función confirm() (Anexo 2 pág. 12) - Realiza un mapa mental con la información proporcionada en la documentación adjunta. (Anexo 2 pág. 20-30) - Realiza Practica Guiada 5 Ejemplo con Sentencia If/else (Anexo 2 pág. 21)	- Código de la Práctica guiada 2 escrita en libreta. (Anexo 2 pág. 1) - Mapa Mental (Anexo 2 pág. 2-19) en su cuaderno. - Código de la Práctica guiada 3 escrita en libreta. (Anexo 2 pág. 11) - Código de la Práctica guiada 4 escrita en libreta. (Anexo 2 pág. 12)

3. Funciones alert(), confirm() y prompt()
4. Ejercicio Practica Guiada 3 Script con expresión
5. Ejercicio Practica Guiada 4 Script conversor
6. Valores Booleanos
7. Operador de Identidad e igualdad
8. Operadores de Comparación o Relacionales AND y OR
9. Operador de Asignación condicional
10. Ejercicio Practica Guiada 5 Ejemplo con Sentencia If/else
11. Ejercicio Practica Guiada 6 Función prompt y estructura if/else
12. Clase String
13. Clase Number
14. Números: tipo number
15. Conversión a enteros
16. Módulo Math
17. Función y estructura y parámetros
18. Array de argumentos
19. Parámetros por defecto

- g. Realiza Ejercicio Practica Guiada 6 Función prompt y estructura if/else (Anexo 2 pág. 22)

SECCION DE EVALUACION

- h. Realiza Practica Autónoma 1 utilizando funciones con parámetros
- i. Realiza Practica Autónoma 2 utilizando funciones con retorno de valor

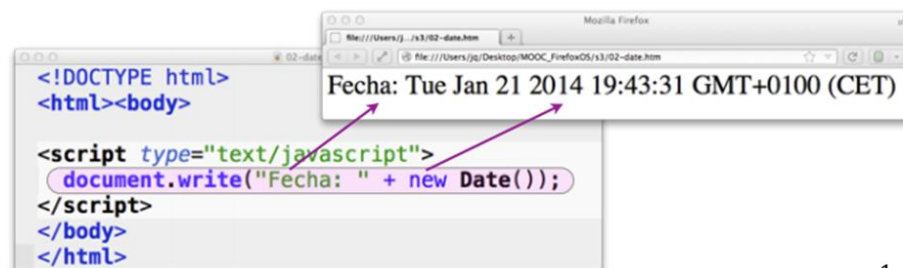
Anexo 2

Parcial 2 Módulo III. Desarrolla aplicaciones Web.
 Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Práctica guiada 2

Script con fecha document.write()

- ◆ Script: programa JavaScript insertado en una página HTML
 - Delimitado con las marcas `<script>` con atributo `"type=text/javascript"`
 - Se ejecuta al cargar la página HTML en el navegador
- ◆ `document.write("Texto")` inserta "Texto" en la página Web
 - En lugar del bloque script, al cargar la página y ejecutar el script
 - `document.writeln("Texto")` inserta además de **Texto**, nueva línea al final



1

5. Mapa Mental (Anexo 2 pág. 20-30) en su cuaderno
6. Código de la Práctica guiada 5 escrita en libreta. (Anexo 2 pág. 21)
7. Código de la Práctica guiada 6 escrita en libreta. (Anexo 2 pág. 22)
8. Código de la Práctica autónoma 1, funciones con parámetros; escrita en libreta.
9. Código de la Práctica autónoma 2; funciones con retorno de valor; escrita en libreta.

Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

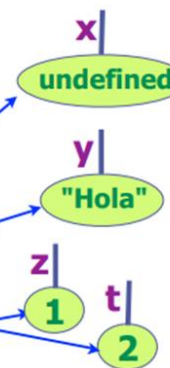
Tema: Variables

Variables y estado del programa

- ◆ Las variables se crean con la sentencia de definición de variables
 - Comienza con la palabra reservada **var**
 - Seguida de la variable, a la que se puede asignar un valor inicial
 - Se pueden crear varias variables en una sentencia
 - separando las definiciones por comas

- ◆ Estado del programa
 - conjunto de valores contenido en todas sus variables

```
var x;           // crea la variable x y asigna undefined
var y = "Hola"; // crea y, asignándole el valor "Hola"
var z = 1, t = 2; // crea x e y, asignándoles 1 y 2 respectivamente
```



Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Tema: Variables

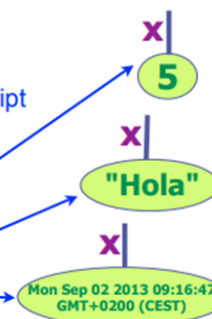
Sentencia de asignación de variables

- ♦ Una variable es un contenedor de valores
 - La sentencia de asignación introduce un nuevo valor en la variable
 - Modificando, por tanto, el estado del programa
- ♦ Las variables de JavaScript son **no tipadas**
 - pueden contener **valores de cualquier tipo** de JavaScript

`var x = 5; // Crea la variable x y le asigna el valor inicial 5`

`x = "Hola"; // Asigna el string (texto) "hola" a la variable x`

`x = new Date(); // Asigna objeto Date a la variable x`

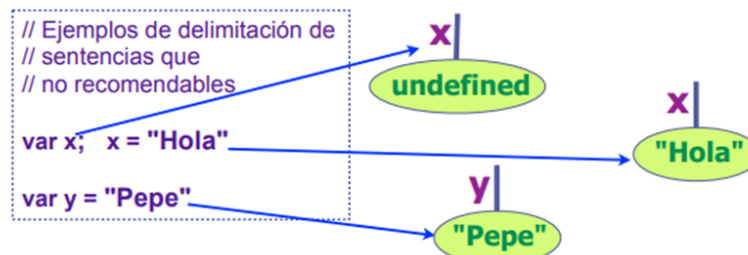


Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Tema: Variables

Delimitación de sentencias

- ◆ “;” delimita el final de una sentencia
- ◆ El final de sentencia también puede delimitarse con nueva línea
 - Pero hay ambigüedades y no se recomienda hacerlo
- ◆ **Recomendación:** cada sentencia en **una línea terminada con “;”**
-> es mas legible y seguro



Anexo 2

Parcial 2

Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Tema: Variables

Nombres de variables

- ◆ El **nombre** (o identificador) de una variable debe comenzar por:
 - **letra, _ o \$**
 - ♦ El nombre pueden contener además **números**
 - Nombres **bien contruidos**: **x, ya_vás, \$A1, \$, _43dias**
 - Nombres **mal contruidos**: **1A, 123, %3, v=7, a?b, ..**
 - ♦ Nombre incorrecto: da error_de_ejecución e interrumpe el programa
- ◆ Un nombre de variable
 - **no** debe ser una **palabra reservada** de JavaScript
- ◆ Las variables son sensibles a **mayúsculas**
 - **mi_var** y **Mi_var** son variables distintas

Anexo 2

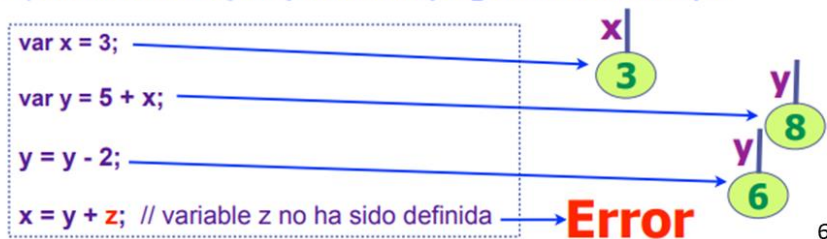
Parcial 2

Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Tema: Variables

Expresiones con variables

- ♦ Una variable representa el valor que contiene
 - Puede ser usada en expresiones como cualquier otro valor
- ♦ Una variable puede utilizarse en la expresión que se asigna a ella misma
 - La parte derecha usa el valor anterior a la ejecución de la sentencia
 - En $y = y - 2$; la variable y tiene el valor 8, por lo que se asigna a y un 6 (8-2)
- ♦ Usar una **variable no definida** en una expresión
 - provoca un **error** y la ejecución del **programa se interrumpe**

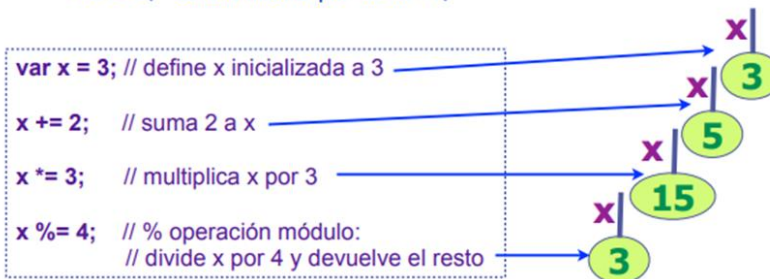


Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Tema: Variables

Asignación con operación

- ◆ Es muy común modificar el valor de una variable
 - sumando, restando, algún valor
 - Por ejemplo, $x = x + 7$; $y = y - 3$; $z = z * 8$;
- ◆ JavaScript tiene operadores especiales para estas operaciones
 - $+=$, $-=$, $*=$, $/=$, $\%=$,(aplica a operadores lógicos, desplazamiento, ..)
 - $x += 7$; será lo mismo que $x = x + 7$;

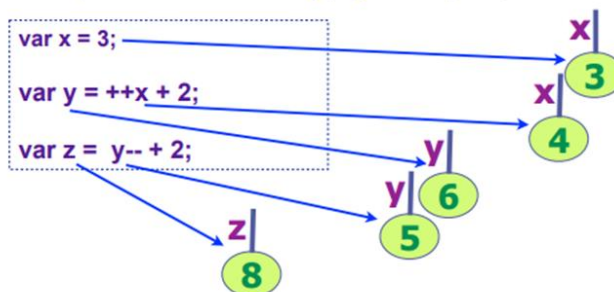


Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Tema: Variables

Pre y post incremento o decremento

- ◆ JavaScript posee los operadores ++ y -- de auto-incremento/decremento
 - auto-incremento (++) suma 1 a la variable a la que se aplica
 - auto-decremento (--) resta 1 a la variable a la que se aplica
- ◆ ++ y -- se aplican a la izquierda o derecha de una variable en una expresión
 - modificando el valor antes o después de evaluar la expresión
 - Al producir efectos laterales y programas crípticos, no se recomienda un uso limitado



Anexo 2

Parcial 2

Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Tema: Variables

Entrada/Salida y ejecución de expresiones

- ◆ La sentencia de **ejecución de expresiones** puede evaluar expresiones
 - como por ejemplo, **3+2;** o **alert("Texto");**
 - sin asignar el resultado, puede ser: **x = 3+2;** o **3+2;**
- ◆ Estas sentencias se utilizan habitualmente para comunicar con el exterior
 - p.e. **alert("Texto");** muestra una ventana desplegable al usuario
- ◆ Una expresión sin efecto lateral, solo genera un valor
 - Si ese valor no se guarda en una variable
 - La instrucción no tiene ningún efecto en el programa, solo consume recursos

```
alert("Texto");           // expresiones útiles, envían mensaje al exterior  
document.write("Texto");
```

```
var x = 3; // definición e inicialización de variable
```

```
x*5 - 2; // es una expresión correcta, pero inútil al no guardar el resultado
```

```
x = x*5 + 2; // asignación, es una expresión útil porque guarda el resultado
```


Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Funciones alert(), confirm() y prompt()

◆ Interacción sencilla basada en “pop-ups”

◆ alert(msj):

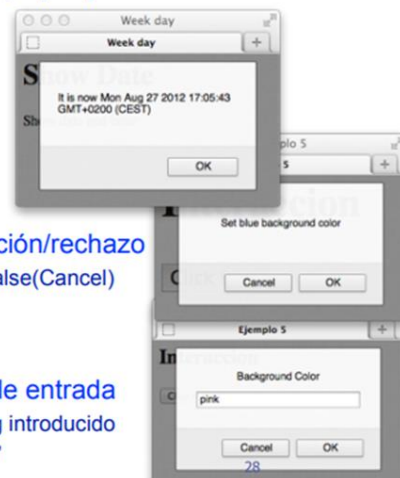
- presenta un mensaje al usuario
 - Retorna al pulsar OK

◆ confirm(msj):

- Presenta mensaje y pide confirmación/rechazo
 - Retorna al pulsar, devuelve true(Ok)/false(Cancel)

◆ prompt(msj):

- Presenta mensaje y pide un dato de entrada
 - Al pulsar OK, retorna y devuelve string introducido
 - Si se pulsa Cancel devuelve “null”



Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Práctica guiada 3

Script con expresión

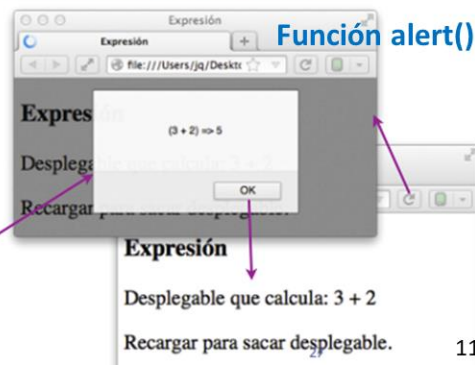
- ◆ Script: programa JavaScript insertado en una página HTML
 - Delimitado con la marca `<script>` con atributo `"type=text/javascript"`
 - Se ejecuta al cargar la página HTML en el navegador
- ◆ `alert("Texto")` muestra "Texto" en un desplegable
 - se utiliza para alertar al usuario sobre algún evento o resultado

```
<!DOCTYPE html>
<html><head><title>Expresión</title>
  <meta charset="UTF-8"></head>

<body>
<h3>Expresión</h3>

Desplegable que calcula: 3 + 2
<p>
Recargar para sacar desplegable.

<script type="text/javascript">
alert("(3 + 2) => " + (3+2));
</script>
</body>
</html>
```



Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

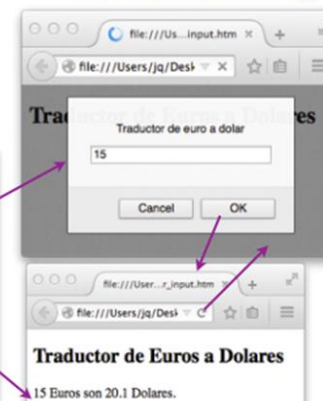
Práctica guiada 4

Script conversor

- ◆ El script pide el número de Euros a convertir a Dólares
 - con el desplegable de la función **prompt()**
- ◆ Cuando el usuario lo introduce
 - calcula el equivalente en dólares ($\times 1,34$)
 - y lo presenta con **document.write(...)**
- ◆ Recargando la página
 - se vuelve a ejecutar el programa

```
<!DOCTYPE html>
<html><body>
<h2>Traductor de Euros a Dolares</h2>
<script type="text/javascript">
  var x = prompt("Traductor de euro a dolar");
  document.write(x + " Euros son "
    + x*1.34 + " Dolares.");
</script>
</body>
</html>
```

Función confirm()



Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Operador Igual, idéntico o diferente



Operadores de identidad e igualdad

- ◆ Identidad o igualdad estricta: `<v1> === <v2>`
 - igualdad de tipo y valor:
 - ◆ aplicable a: **number**, **boolean** y **strings**
 - En objetos es **identidad de referencias**
- ◆ Desigualdad estricta: `<v1> !== <v2>`
 - negación de la igualdad estricta
- ◆ Igualdad y desigualdad débil: `==` y `!=`
 - **No utilizar!** Conversiones impredecibles!

// Identidad de tipos básicos

```

0 === 0           => true
0 === 0.0         => true
0 === 0.00        => true

0 === 1           => false
0 === false       => false

'2' === '2'       => true
'2' === '02'      => false

'' === ''         => true
'' === ' '        => false
  
```

Anexo 2

Parcial 2

Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Operadores Relacionales

Operadores de comparación

◆ JavaScript tiene 4 operadores de comparación

- Menor: `<`
- Menor o igual: `<=`
- Mayor: `>`
- Mayor o igual: `>=`

◆ Utilizar comparaciones solo con números (number)

- poseen una relación de orden bien definida

◆ No se recomienda utilizar con otros tipos: **string, boolean, object, ..**

- Las relación de orden en estos tipos existe, pero es muy poco intuitiva

`1.2 < 1.3` `=>` true

`1 < 1` `=>` false

`1 <= 1` `=>` true

`1 > 1` `=>` false

`1 >= 1` `=>` true

`false < true` `=>` true

`"a" < "b"` `=>` true

`"a" < "aa"` `=>` true

Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Operador y de JavaScript: &&

Operador Lógico AND

- ◆ Operador lógico y
 - operador binario: **<valor_1> y <valor_2>**
 - ♦ Verdadero solo si ambos valores son verdaderos
- ◆ **&&** se ha extendido y es más que un operador lógico
 - No convierte el resultado a booleano
 - ♦ Solo **interpreta <valor_1> como booleano** y según sea **false** o **true**
 - ♦ Devuelve como resultado **<valor_1>** o **<valor_2>** sin modificar
- ◆ Semántica del operador lógico y (and) de JavaScript: **<valor_1> && <valor_2>**
 - si **<valor_1>** se evalúa a **false**
 - ♦ devuelve **<valor_1>**, sino devuelve **<valor_2>**

```

true && true  => true
false && true => false
true && false => false
false && false => false
  
```

```

0 && true    => 0
1 && "5"     => "5"
  
```


Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Operador o de JavaScript: ||

Operador Lógico OR

- ◆ Operador lógico o
 - operador binario: **<valor_1> o <valor_2>**
 - ♦ Verdadero solo si ambos valores son verdaderos
- ◆ || se ha extendido y es más que un operador lógico
 - No convierte el resultado a booleano
 - ♦ Solo **interpreta <valor_1> como booleano** y según sea **false** o **true**
 - ♦ Devuelve como resultado **<valor_1>** o **<valor_2>** sin modificar
- ◆ Semántica del operador lógico o (or) de JavaScript: **<valor_1> || <valor_2>**
 - si **<valor_1>** se evalúa a **true**
 - ♦ devuelve **<valor_1>**, sino devuelve **<valor_2>**

```
true || true  => true
false || true => true
true  || false => true
false || false => false
```

```
undefined || 0  => 0
13 || 0         => 13
```

```
// Asignar valor por defecto
// si x es undefined o null
```

```
x = x || 0;
```

Anexo 2

Parcial 2

Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Operador de asignación condicional: "?:"

- ◆ El operador de asignación condicional
 - devuelve un valor en función de una **condición** lógica
 - ♦ La condición lógica va siempre entre paréntesis
- ◆ Semántica de la asignación condicional: **(condición) ? <valor_1> : <valor_2>**
 - si **condición** se evalúa a **true**
 - ♦ devuelve **<valor_1>**, sino devuelve **<valor_2>**

(true) ? 0 : 7	=> 0
(false) ? 0 : 7	=> 7

(7) ? 0 : 7	=> 0
("") ? 0 : 7	=> 7

Anexo 2

Parcial 2

Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.



The screenshot shows a web browser window with the title "15-if-else.htm". The browser displays the text "Sentencia if/else" and "0.5242976508023318 MAYOR que 0,5". A blue arrow points from the code in the background to the browser window. The code is as follows:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
</head>
<body>
<h3> Sentencia if/else </h3>

<script type="text/javascript">

    // Math.random() devuelve
    // número aleatorio entre 0 y 1.
    var numero = Math.random();

    if (numero <= 0.5){
        document.writeln(numero + ' MENOR que 0,5');
    }
    else {
        document.writeln(numero + ' MAYOR que 0,5');
    }
</script>
</body>
</html>
```

Ejemplo con sentencia if/else

Práctica guiada 5

Anexo 2 **Parcial 2** **Módulo III. Desarrolla aplicaciones Web.**
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Ejemplo de prompt()

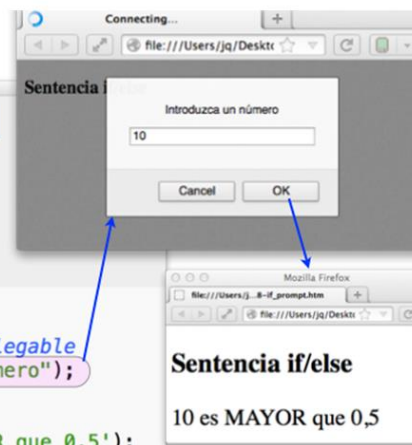
Práctica guiada 6

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
</head>
<body>
<h3> Sentencia if/else </h3>

<script type="text/javascript">

    // Prompt pide un dato con un desplegable
    var numero = prompt("Introduzca un número");

    if (numero <= 0,5){
        document.writeln(numero + ' es MENOR que 0,5');
    }
    else {
        document.writeln(numero + ' es MAYOR que 0,5');
    }
</script>
</body>
</html>
```



Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Clase String

- ◆ La clase String
 - incluye métodos y propiedades para procesar strings
 - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String
- ◆ Un string es un array de caracteres
 - un índice entre **0** y **número_de_caracteres-1** referencia cada carácter
- ◆ Propiedad con tamaño: **'ciudad'.length** **=> 6**
- ◆ Acceso como array: **'ciudad'[2]** **=> 'u'**
- ◆ Método: **'ciudad'.charCodeAt(2)** **=> 117**
 - devuelve código UNICODE de tercer carácter
- ◆ Método: **'ciudad'.indexOf('da')** **=> 3**
 - devuelve posición de substring
- ◆ Método: **'ciudad'.substring(2,5)** **=> 'uda'**
 - devuelve substring entre ambos índices

ciudad
[0] [1] [5]

Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Clase Number

- ◆ La clase Number encapsula números
 - como objetos equivalentes
- ◆ Number define algunos métodos útiles
 - **toFixed(n)** devuelve string
 - ♦ redondeando a n decimales
 - **toExponential(n)** devuelve string
 - ♦ redondeando mantisa a n decima.
 - **toPrecision(n)** devuelve string
 - ♦ redondeando a n dígitos
- ◆ JS convierte una expresión a objeto al
 - aplicar el método a una expresión
 - ♦ **Ojo!** literales dan error sintáctico

Mas info:

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Number

```
var x = 1.1;

x.toFixed(0)    => "1"
x.toFixed(2)    => "1.10"

(1).toFixed(2)  => "1.00"

1.toFixed(2)    => Error sintáctico

Math.PI.toFixed(4) => "3.1416"

(0.1).toExponential(2) => "1.00e-1"
x.toExponential(2)    => "1.10e+0"

(0.1).toPrecision(2)  => "0.10"
x.toPrecision(2)      => "1.1"
```


Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Números: tipo number

◆ Los números se representan con literales de

- **Enteros:** 32
 - ◆ Entero máximo: 9007199254740992
- **Decimales:** 32.23
- **Coma flotante:** 3.2e1 (3,2x10)
 - ◆ Rango real: 1,797x10³⁰⁸ --- 5x10⁻³²⁴

◆ Todos los números son del tipo **number**

◆ Todos los números se representan igual internamente

- coma flotante de doble precisión (64bits)

◆ El tipo number incluye 2 valores especiales

- **Infinity:** representa desbordamiento
- **NaN:** representa resultado no numérico

10 + 4	=> 14	// sumar
10 - 4	=> 6	// restar
10 * 4	=> 40	// multiplicar
10 / 4	=> 2.5	// dividir
10 % 4	=> 2	// operación resto

//decimales dan error de redondeo
0.1 + 0.2 => 0,3000000000004

3e2	=> 300
3e-2	=> 0,03

+10/0	=> Infinity	//desborda
-10/0	=> -Infinity	//desborda

5e500	=> Infinity	//desborda
-------	-------------	------------

Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Conversión a enteros

- ◆ Cuando JavaScript calcula expresiones
 - **convirtiendo tipos** según necesita
 - ♦ utiliza las prioridades de operadores
- ◆ Conversión a **entero (o real)**
 - **booleano**: true a 1, false a 0
 - **String**: Convierte número a valor o NaN
 - **null**: a 0, **undefined**: a NaN
- ◆ Convertir un **string** a un **número**
 - se denomina también "parsear" o analizar sintácticamente
 - ♦ es similar al análisis sintáctico realizado a los literales de números

'67' + 13	=> 6713
+'67' + 13	=> 80
+'6.7e1' + 13	=> 80
'xx' + 13	=> 'xx13'
+'xx' + 13	=> NaN
13 + true	=> 14
13 + false	=> 13

Anexo 2

Parcial 2

Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Modulo Math

- ◆ El Modulo Math contiene
 - constantes y funciones matemáticas
- ◆ Constantes
 - Números: E, PI, SQRT2, ...
 - ...
- ◆ Funciones
 - sin(x), cos(x), tan(x), asin(x),
 - log(x), exp(x), pow(x, y), sqrt(x),
 - abs(x), ceil(x), floor(x), round(x),
 - min(x,y,z,...), max (x,y,z,...), ...
 - random()

Mas info:

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Math

Math.PI => 3.141592653589793
Math.E => 2.718281828459045

// numero aleatorio entre 0 y 1
Math.random() => 0.7890234

Math.pow(3,2) => 9 // 3 al cuadrado
Math.sqrt(9) => 3 // raíz cuadrada de 3

Math.min(2,1,9,3) => 1 // número mínimo
Math.max(2,1,9,3) => 9 // número máximo

Math.floor(3.2) => 3
Math.ceil(3.2) => 4
Math.round(3.2) => 3

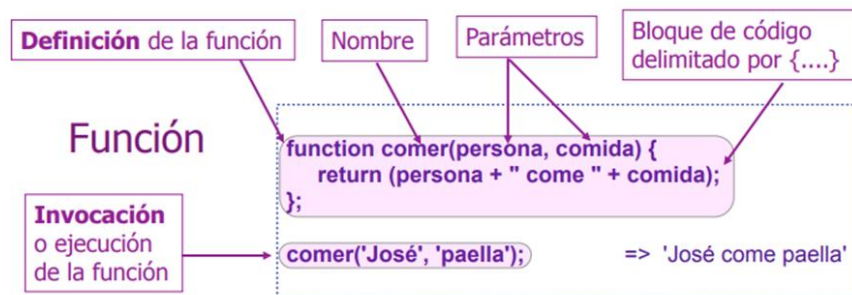
Math.sin(1) => 0.8414709848078965
Math.asin(0.8414709848078965) => 1

Anexo 2

Parcial 2

Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.



- ◆ **Función:**
 - bloque de código con parámetros, invocable (ejecutable) a través del nombre
 - La ejecución finaliza con la sentencia “**return expr**” o al **final** del bloque
 - Al acabar la ejecución, devuelve un resultado: **valor de retorno**
- ◆ **Valor de retorno**
 - resultado de evaluar **expr**, si se ejecuta la sentencia “**return expr**”
 - **undefined**, si se alcanza final del bloque sin haber ejecutado ningún **return**

Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Parámetros de una función

- ◆ Los parámetros de la función son variables utilizables en el cuerpo de la función
 - Al invocarlas se asignan los valores de la invocación
- ◆ La función se puede invocar con un **número variable de parámetros**
 - Un **parámetro inexistente** está **undefined**

```
function comer(persona, comida) {  
    return (persona + " come " + comida);  
};
```

```
comer('José', 'paella'); => 'José come paella'
```

```
comer('José', 'paella', 'carne'); => 'José come paella'  
comer('José'); => 'José come undefined'
```

Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

El array de argumentos

- ◆ Los parámetros de la función están accesibles también a través del
 - array de argumentos: **arguments[...]**
 - Cada parámetro es un elemento del array
- ◆ En: **comer('José', 'paella')**
 - **arguments[0]** => 'José'
 - **arguments[1]** => 'paella'

```
function comer() {  
    return (arguments[0] + " come " + arguments[1]);  
};
```

```
comer('José', 'paella');           => 'José come paella'
```

```
comer('José', 'paella', 'carne'); => 'José come paella'  
comer('José');                    => 'José come undefined'
```


Anexo 2 **Parcial 2** Módulo III. Desarrolla aplicaciones Web.
 Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Parámetros por defecto

- ◆ Funciones invocadas con un número variable de parámetros
 - Suelen definir parámetros por defecto con el operador ||
 - ♦ "x || <parámetro_por_defecto>"
- ◆ Si x es "undefined", será false y devolverá **parámetro por defecto**
- ◆ Los parámetros son variables y se les puede asignar un valor

```
function comer (persona, comida) {
  persona = (persona || 'Alguién');
  comida = (comida || 'algo');
  return (persona + " come " + comida);
};

comer('José');    => 'José come algo'
comer();          => 'Alguien come algo'
```

28

Aprendizajes esenciales o Competencias esenciales 3er parcial	Estrategias de Aprendizaje	Productos a Evaluar
Competencia Profesional El alumno Emplea JavaScript para control de eventos y el formulario de datos Contenido - Objetos - Propiedades - Clases y Herencia - Métodos de la clase	- Realiza la evaluación de seguimiento (Anexo 3 pág. A-C) - Realiza un mapa mental con la información proporcionada en la documentación adjunta. (Anexo 3 pág. 1-21) - Realiza práctica guiada 7 acceso a DOM (Anexo 3 pág. 16) - Realiza práctica guiada 8 varios Scripts en un documento HTML (Anexo 3 pág. 18) - Realiza un mapa mental con la información proporcionada en la documentación adjunta. (Anexo 3 pág. 22-36) - Realiza Practica Guiada 9 Ejemplo con Sentencia for/in (Anexo 3 pág. 24) - Realiza Ejercicio Practica Guiada 10 window.screen (Anexo 3 pág. 25) - Realiza Ejercicio Practica Guiada 11 Reloj (Anexo 3 pág. 27)	- Resuelve evaluación de seguimiento (Anexo 3 pág. A-C) . Responde de forma escrita en su libreta - Mapa Mental (Anexo 3 pág. 1-21) dibujado en su cuaderno - Código de la Práctica guiada 7 escrita en libreta o en su celular utilizando el editor de código propuesto. (Anexo 3 pág. 16)

5. Definición de un nuevo método. 6. Algunas clases predefinidas 7. Objetos anidados: árboles 8. Propiedades Dinámicas 9. Usar propiedades dinámicas 10. Referencias a objetos 11. Identidad de Objetos 12. Objeto window o this 13. Variables globales y el entorno de ejecución 14. DOM: Document Object Model 15. Acceso a DOM 16. Varios scripts en un mismo documento HTML 17. Location, history, screen 18. window.screen 19. funciones de Selección de elementos 20. Sentencia for/in (sintaxis) 21. Eventos periódicos con setInterval(...) 22. Eventos DOM 23. Eventos definidos directamente en Javascript 24. Eventos HTML 25. Evento como propiedad 26. Formularios	i. Realiza Ejercicio Practica Guiada 12 Eventos en HTML (Anexo 3 pág. 30) j. Realiza Ejercicio Practica Guiada 13 Evento Propiedad (Anexo 3 pág. 31) k. Realiza Ejercicio Practica Guiada 14 Evento Propiedad (Anexo 3 pág. 32) l. Realiza Ejercicio Practica Guiada 15 Evento Onload (Anexo 3 pág. 33) m. Realiza Ejercicio Practica Guiada 16 Formularios (Anexo 3 pág. 34) n. Realiza Ejercicio Practica Guiada 17 Validación de campos en el formulario (Anexo 3 pág. 36) SECCION DE EVALUACION Practica autónoma 3 Diseña una página web y escribe el código en JavaScript necesario para realizar los siguiente: ✓ Crea una galería fotográfica con los siguientes comportamientos: ➤ Al pasar el mouse por cada foto cambiar el color de su borde, al quitar el mouse regresar a su color inicial. ➤ Al dar clic sobre la imagen aumentar su tamaño. ➤ Al soltar el botón del mouse regresar a su tamaño original. ➤ Al dar doble clic sobre la imagen mostrar información informativa sobre la imagen.	d. Código de la Práctica guiada 8 escrita en libreta o en su celular utilizando el editor de código propuesto. (Anexo 3 pág. 18) e. Mapa Mental (Anexo 3 pág. 22-36) dibujado en su cuaderno f. Código de la Práctica guiada 9 escrita en libreta o en su celular utilizando el editor de código propuesto. (Anexo 3 pág. 24) g. Código de la Práctica guiada 10 escrita en libreta o en su celular utilizando el editor de código propuesto. (Anexo 3 pág. 25) h. Código de la Práctica guiada 11 escrita en libreta o en su celular utilizando el editor de código propuesto. (Anexo 3 pág. 27) i. Código de la Práctica guiada 12 escrita en libreta o en su celular utilizando el editor de código propuesto. (Anexo 3 pág. 30) j. Código de la Práctica guiada 13 escrita en libreta o en su celular utilizando el editor de código propuesto. (Anexo 3 pág. 31) k. Código de la Práctica guiada 14 escrita en libreta o en su
--	---	--

27. Validación de campos

Anexo 3 **Parcial 3** **Módulo III. Desarrolla aplicaciones Web.** **Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente**

Evaluación de seguimiento

1- ¿En qué lugar se ejecuta generalmente el código JavaScript?

- ☐ Servidor
- ☐ Cliente (en el propio navegador de internet)

2- ¿Cuáles de estas son marcas para la inserción del código JavaScript en las páginas HTML?

- ☐ `<javascript_code> y </javascript_code>`
- ☐ `<script> y </script>`
- ☐ `<?script> y <script?>`

3- La llamada al código Javascript debe colocarse en:

- ☐ La sección Body de la página
- ☐ Antes de la etiqueta HTML
- ☐ Puede colocarse en la sección Head o en Body

Pág. A

celular utilizando el editor de código propuesto. **(Anexo 3 pág. 32)**

- l. Código de la Práctica guiada 15 escrita en libreta o en su celular utilizando el editor de código propuesto. **(Anexo 3 pág. 33)**
- m. Código de la Práctica guiada 16 escrita en libreta o en su celular utilizando el editor de código propuesto. **(Anexo 3 pág. 34)**
- n. Código de la Práctica guiada 17 escrita en libreta o en su celular utilizando el editor de código propuesto. **(Anexo 3 pág. 36)**
- o. Código de la Práctica autónoma 3, Crea una galería fotográfica; escrita en libreta o en su celular utilizando el editor de código propuesto

Anexo 3 **Parcial 3** **Módulo III. Desarrolla aplicaciones Web.**
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Evaluación de seguimiento

4- En JavaScript, para darle el nombre a una variable, objeto o función, debemos tener en cuenta que:

- ☐ No se pueden usar mayúsculas
- ☐ JavaScript no distingue entre mayúsculas y minúsculas
- ☐ JavaScript diferencia entre mayúsculas y minúsculas

5-¿Cuál es la instrucción usada para devolver un valor en una función de JavaScript?

- ☐ Send
- ☐ Return
- ☐ Value

6- Para terminar las instrucciones en Javascript se utiliza:

- ☐ Un punto y coma
- ☐ Un punto y coma o un salto de línea
- ☐ La sentencia End.

Anexo 3 **Parcial 3** **Módulo III. Desarrolla aplicaciones Web.**
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

7- ¿Cuál de estas instrucciones está correctamente escrita en Javascript?

- ☐ `if (a==0) alert (a);`
- ☐ `if (a=0) print a;`
- ☐ `if (a==0) { print [a] }`
- ☐ `if (a==0): print a;`

Evaluación de seguimiento

8- Para concatenar cadenas de caracteres en Javascript se usa el carácter:

- ☐ `&` (ampersand)
- ☐ `+` (más)
- ☐ `.` (punto)
- ☐ `*` (por)

9- ¿Es posible hacer que se ejecute un formulario por JavaScript?

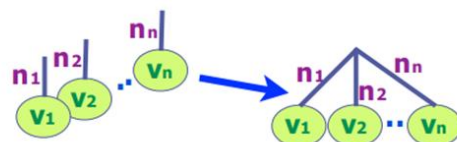
- ☐ No, esa función sólo puede realizarse mediante código PHP, y se ha de realizar por tanto en el servidor.
- ☐ Sí, de hecho los formularios se crean con código Javascript, por lo que es el propio Javascript el que los ejecuta.
- ☐ Sí, por ejemplo basta con pasarle a una función Javascript el identificador del formulario, y aplicarle el comando "`submit`" para ejecutar ese formulario

10- ¿Todo el código JavaScript debe estar por fuerza dentro del archivo `.html` de la página web?

- ☐ Sí, porque si no, no se podría ejecutar en el navegador
- ☐ No, es posible incluir código JavaScript en ficheros de extensión `.js` y hacer un "`include`" en la sección HEAD de la página HTML

Anexo 3 **Parcial 3** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Objetos



- ◆ Los objetos son colecciones de variables
 - agrupadas como un elemento estructurado que llamamos objeto
 - Las variables de un objeto se denominan **propiedades**
- ◆ Una **propiedad** es un par **nombre:valor** donde
 - los **nombres** deben ser **todos diferentes** en un mismo objeto
- ◆ Se definen con el literal: { **nombre:valor**, ... }
 - Por ejemplo: {titulo: 'Avatar', director: 'James Cameron'}
 - crea un objeto con 2 propiedades:
 - titulo:'Avatar'
 - director:'James Cameron'



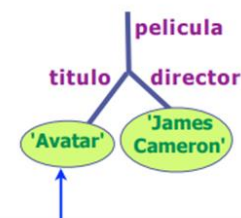
Anexo 3

Parcial 3

Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Propiedades

- ◆ El acceso a propiedades utiliza el operador punto
 - **obj.propiedad**
- ◆ Por ej. en: `var pelicula = {titulo: 'Avatar', director: 'James Cameron'}`
 - **pelicula.titulo** => "Avatar"
 - **pelicula.director** => "James Cameron"
 - **pelicula.fecha** => undefined // la propiedad fecha no existe
- ◆ Aplicar el operador punto sobre **undefined** o **null**
 - Provoca un **Error_de_ejecución** y aborta la ejecución del programa
- ◆ La notación punto solo acepta nombres de propiedades
 - Con la sintaxis de variables: **a, _method, \$1, ...**
 - ◆ No son utilizables: **"#43", "?a=1",**





Anexo 3 Parcial 3 Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Nombres de propiedades como variables

- ◆ La notación array permite acceder también a propiedades
 - cuyo nombre esta en una variable en forma de string
 - ♦ Esto no es posible con la notación punto

```
var x = {titulo: 'Avatar', director: 'James Cameron'}
```

```
x.titulo;      => 'Avatar'
```

```
x['titulo'];   => 'Avatar'
```

```
var p = 'titulo'; // inicializada con string 'titulo'
```

```
x[p];          => 'Avatar'
```

```
x.p;           => undefined
```

x tiene una propiedad de nombre **'titulo'**, que es el string que contiene **p**

El **objeto x** no tiene ninguna propiedad de nombre **p** y devuelve **undefined**

Anexo 3 **Parcial 3** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Clases y herencia

- ◆ Todos los objetos de JavaScript pertenecen a la **clase Object**
 - Javascript posee mas clases predefinidas que derivan de Object
 - **Date, Number, String, Array, Function,**
 - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Predefined_Core_Objects
 - Un objeto hereda los métodos y propiedades de su clase
- ◆ Un **método** es una operación (~función) invocable sobre un objeto
 - Se invoca con la notación punto: **objeto.metodo(..params..)**
- ◆ Todas las clases tienen un constructor con el nombre de la clase
 - que permite crear objetos con el operador **new**
 - Por ejemplo, **new Object()** crea un objeto vacío equivalente a **{}**

Anexo 3 Parcial 3

Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Métodos de la clase

- ◆ Un objeto **hereda** métodos de su **clase**, por ejemplo
 - los objetos de la clase **Date** heredan métodos como
 - ◆ **toString()**, **getDay()**, **getFullYear()**, **getHours()**, **getMinutes()**, (ver ejemplo)
 - ◆ https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Objetos_globales/Date
- ◆ Solo se puede invocar métodos heredados o definidos en un objeto
 - Invocar un método **no heredado ni definido en un objeto**
 - ◆ provoca un **error_de_ejecución**

```
var fecha = new Date();
```

```
fecha.toString()    => Fri Aug 08 2014 12:34:36 GMT+0200 (CEST)
fecha.getHours()    => 12
fecha.getMinutes()  => 34
fecha.getSeconds()  => 36
```

Anexo 3 Parcial 3 Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente

Definición de un nuevo método de un objeto

- ◆ Los métodos se pueden definir también directamente en un objeto
 - El nuevo método solo se define para ese objeto (no es de la clase)
- ◆ Invocar un método cambia el **entorno de ejecución** de JavaScript
 - pasando a ser el **objeto invocado**, que se referencia con **this**
 - ♦ **this.titulo** referencia la propiedad **titulo** del objeto **pelicula**

```
var pelicula = {  
  titulo:'Avatar',  
  director:'James Cameron',
```

```
  resumen:function () {  
    return "El director de " + this.titulo + " es " + this.director;  
  }  
}
```

```
pelicula.resumen() => "El director de Avatar es James Cameron"
```


Anexo 3 **Parcial 3** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Algunas Clases predefinidas

- ◆ **Object**
 - Clase raíz, suele usarse el literal: `{a:3, b:"que tal"}`
- ◆ **Array**
 - Colección indexable, suele usarse el literal: `[1, 2, 3]`
- ◆ **Date**
 - Hora y fecha extraída del reloj del sistema: `new Date()`
- ◆ **Function**
 - Encapsula código, suele usarse literal o def.: `function (x) {...}`
- ◆ **RegExp**
 - Expresiones regulares, suele usarse el literal: `/(hola)+$/`
- ◆ **Math**
 - Módulo con **constantes** y **funciones matemáticas**
- ◆ **Number, String y Boolean**
 - Clases que encapsulan valores de los tipos **number**, **string** y **boolean** como objetos
 - Sus métodos se aplican a los tipos directamente, la conversión a objetos es automática

Anexo 3 **Parcial 3** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Objetos anidados: árboles

```
var pelicula = {  
  titulo: 'Avatar',  
  director: {  
    nombre: 'James',  
    apellido: 'Cameron'  
  }  
};
```

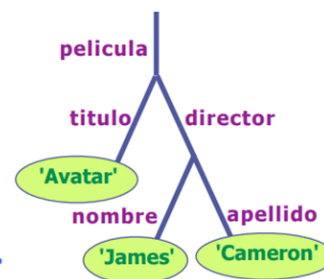
- ◆ Los objetos pueden **anidarse** entre si
 - Los objetos anidados representan **árboles**

- ◆ La notación punto o array puede **encadenarse**

- Representando un **camino en el árbol**

- Las siguientes expresiones se evalúan así:

- `pelicula.director.nombre` => 'James'
- `pelicula['director']['nombre']` => 'James'
- `pelicula['director'].apellido` => 'Cameron'
- `pelicula.estreno` => undefined
- `pelicula.estreno.año` => Error_de_ejecución



Anexo 3

Parcial 3

Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Propiedades dinámicas

- ◆ Las propiedades de objetos
 - Pueden **crearse**
 - Pueden **destruirse**
- ◆ Operaciones sobre **propiedades**
 - **x.c = 4** ¡¡OJO: sentencia compleja!!
 - ♦ si propiedad **x.c** existe, le asigna **4**;
 - ♦ si **x.c** no existe, crea **x.c** y le asigna **4**
 - **delete x.c**
 - ♦ si existe **y.c**, la elimina; si no existe, no hace nada
 - **"c" in x**
 - ♦ si **x.c** existe, devuelve **true**, sino devuelve, **false**

```
var x = { peras:3, fresas:20};  
x.peras = 7;  
x.kiwis = 5;  
delete x.fresas;
```



Anexo 3

Parcial 3

Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Usar propiedades dinámicas

- ◆ Las propiedades dinámicas de JavaScript
 - son muy útiles si se utilizan bien
- ◆ Un objeto solo debe definir las propiedades
 - que contengan información conocida
 - ♦ añadirá mas solo si son necesarias
- ◆ La información se puede consultar con
 - **prop1 && prop1.prop2**
 - ♦ para evitar errores de ejecución
 - ♦ si las propiedades no existen

// Dado un objeto **pel** definido con

```
var pel = {  
  titulo: 'Avatar',  
  director: 'James Cameron'  
};
```

// se puede añadir **pel.estreno** con

```
pel.estreno = {  
  año: '2009',  
  cine: 'Tivoli'  
}
```

// La expresión

pel.estreno && pel.estreno.año

// devuelve **pel.estreno** o **undefined**
// evitando **ErrorDeEjecución**, si
// **pel.estreno** no se hubiese creado

Anexo 3 Parcial 3

Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

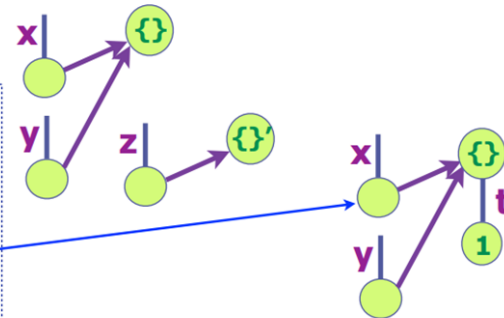
```
var x = {}; // x e y tienen la  
var y = x; // misma referencia
```

```
var z = {}; // la referencia a z  
            // es diferente de  
            // la de x e y
```

```
x.t = 1;
```

```
x.t  => 1 // x accede al mismo  
y.t  => 1 // objeto que y  
z.t  => undefined
```

Referencias a objetos



- ◆ Las variables que contienen objetos
 - solo contienen la referencia al objeto
- ◆ El objeto esta en otro lugar en memoria
 - indicado por la referencia
- ◆ Esto produce efectos laterales
 - como ilustra el ejemplo

Anexo 3

Parcial 3

Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Identidad de objetos

◆ Las referencias a objetos afectan a la identidad

- porque identidad de objetos
 - ◆ es identidad de referencias
- los objetos no se comparan
 - ◆ se comparan solo las referencias
- es poco útil si no se redefine

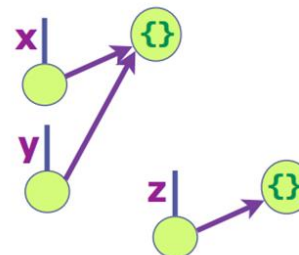
◆ Igualdad (débil) de objetos == y !=

- no tiene utilidad tampoco con objetos
 - ◆ no se debe utilizar

```
var x = {}; // x e y contienen la misma referencia
var y = x; // misma referencia
```

```
var z = {} // la referencia a z es diferente de x e y
```

```
x === y    => true
x === {}   => false
x === z    => false
```



Anexo 3 Parcial 3 Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Objeto window o this



- ◆ El entorno de ejecución de JavaScript es el **objeto global window**
 - El **objeto global window** tiene propiedades con información sobre
 - Objetos predefinidos de JavaScript, el **navegador**, el **documento HTML**,
- ◆ **window** se referencia también como **this** en el entorno global
 - La **propiedad document** de **window** se referencia como
 - `window.document`, `this.document` o `document`

Anexo 3 Parcial 3 Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Variables globales y el entorno de ejecución

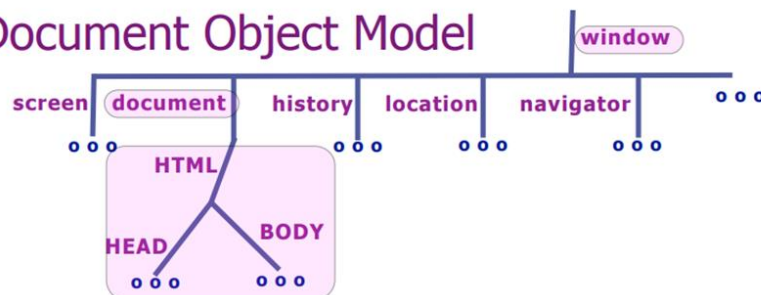


- ◆ Un programa JavaScript se ejecuta con el objeto **window** como entorno
 - una asignación a una variable no definida como **x = 1;**
 - ♦ Crea una nueva **propiedad de window** de nombre **x**, porque
 - **x = 1;** es equivalente a **this.x = 1;** y a **window.x = 1;**
- ◆ Olvidar definir una variable, es un error muy habitual
 - y al asignar un valor a la variable no definida, JavaScript no da error
 - ♦ sino que crea una nueva **propiedad de window**
 - Es un error de diseño de JavaScript y hay que tratar de evitarlo

Anexo 3 Parcial 3

Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

DOM: Document Object Model



- ◆ **Objeto DOM:** objeto JS asociado al documento HTML visualizado en el navegador
 - El navegador lo almacena en la propiedad **document** de **window**
 - ♦ Que contiene el árbol de objetos DOM de la página HTML
 - ♦ **Doc:** <https://developer.mozilla.org/en/docs/Web/API/Document>
- ◆ Los objetos DOM pueden buscarse por **atributos ("id", "class", ..) de HTML**
 - Por ej., el método **document.getElementById("idx")** devuelve el objeto DOM
 - ♦ asociado al elemento de la página HTML con **identificador "idx"**

Anexo 3 Parcial 3 Módulo III. Desarrolla aplicaciones Web.

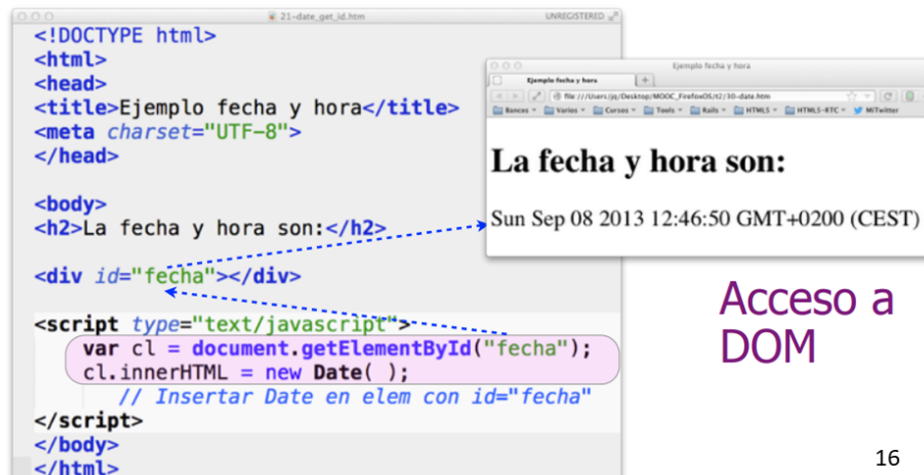
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

◆ `getElementById ("fecha")` devuelve el objeto DOM de `<div id="fecha"></div>`

◆ Propiedad `innerHTML` de un objeto DOM

- Permite extraer/insertar HTML en el elemento del documento

Práctica guiada 7



```
<!DOCTYPE html>
<html>
<head>
<title>Ejemplo fecha y hora</title>
<meta charset="UTF-8">
</head>

<body>
<h2>La fecha y hora son:</h2>
<div id="fecha"></div>
<script type="text/javascript">
  var cl = document.getElementById("fecha");
  cl.innerHTML = new Date( );
  // Insertar Date en elem con id="fecha"
</script>
</body>
</html>
```

La fecha y hora son:
Sun Sep 08 2013 12:46:50 GMT+0200 (CEST)

Acceso a DOM

Anexo 3 Parcial 3 Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

- ◆ La propiedad **fecha** de **window** contiene el objeto DOM de `<div id="fecha"></div>`
 - **No está recomendado** usar estas propiedades de window
 - ◆ Si hay colisión de nombres con otras propiedades y puede haber problemas
- ◆ Propiedad **innerHTML** de un objeto DOM
 - Permite extraer/insertar HTML en el elemento del documento



```
<!DOCTYPE html>
<html>
<head>
<title>Ejemplo fecha y hora</title>
<meta charset="UTF-8">
</head>

<body>
<h2>La fecha y hora son:</h2>
<div id="fecha"></div>

<script type="text/javascript">
  fecha.innerHTML = new Date( );
  // Insertar Date en elem con id="fecha"
</script>
</body>
</html>
```

Acceso a
DOM

Anexo 3 Parcial 3 Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Práctica guiada 8

Varios scripts

- ◆ Una página
 - con varios scripts
 - ◆ es un único programa
- ◆ Scripts se juntan siguiendo el orden de aparición en la página

```
<!DOCTYPE html>
<html>
<head>
<title>Ejemplo de función</title>
<meta charset="UTF-8">

<script type="text/javascript">
function mostrar_fecha( ) {
var cl = document.getElementById("fecha");
cl.innerHTML = new Date( );
}
</script>

</head>

<body>
<h2>Ejemplo con función</h2>

<div id="fecha"><div>

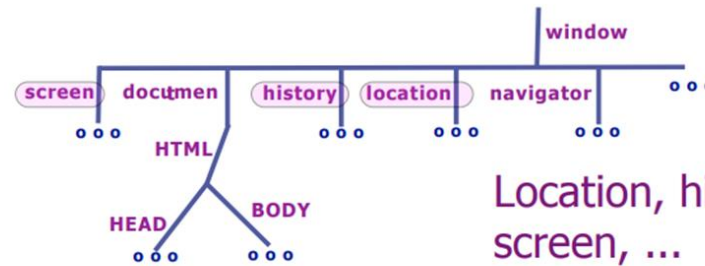
<script type="text/javascript">
mostrar_fecha( ); // Llamar función
</script>

</body>
</html>
```



- ◆ función mostrar_fecha()
 - Se define e invoca en scripts diferentes

Anexo 3 Parcial 3 Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.



Location, history, screen, ...

- ♦ **location**: propiedad que contiene el URL a la página en curso
 - `location = "http://www.upm.es"` // Carga la página en el navegador
 - ♦ `location.reload()` re-carga la página en curso
 - propiedades: `href` (url), `protocol`, `hostname`, `port`, `pathname`, `search` (query), ...
- ♦ **history**: propiedad con la historia de navegación
 - Métodos para navegar por la historia: `history.back()`, `history.forward()`, ...
- ♦ **screen**: dimensiones de la pantalla
 - `width`, `height`, `availWidth`, `availHeight`: para adaptar apps a pantallas móviles

Anexo 3 Parcial 3 Módulo III. Desarrolla aplicaciones Web Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente

Propiedades de window

La propiedad location.href contiene el URL:
 file:///Users/jq/Desktop/MOOC_FirefoxOS/s3/09-window_table.htm

Las dimensiones de mi pantalla (screen.width y screen.height) son: 2560 x 1440

Propiedades de window

La propiedad location.href contiene el URL:

 <p>
 Los pixels de mi pantalla (screen.width y screen.height) son:

<script>
 document.getElementById("i1").innerHTML = location.href;
 var p = document.getElementById("i2")
 p.innerHTML = screen.width + " x " + screen.height;
 </script>
 </body>
 </html>

Anexo 3 **Parcial 3** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente

Funciones de selección de elementos

◆ getElementById("my_id")

- Es el mas sencillo de utilizar porque devuelve
 - El objeto DOM con el identificador buscado o null si no lo encuentra
 - ¡Un identificador solo puede estar en un objeto de una página HTML!

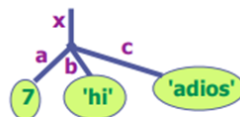
◆ getElementByName("my_name"), getElementsByTagName("my_tag"), getElementsByClassName("my_class"), querySelectorAll("CSS selector"),...

- Devuelven una matriz de objetos
 - Por ejemplo: getElementByName("my_name")[0]
 - referencia el primer elemento con atributo name="my_name"

Anexo 3 **Parcial 3** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Sentencia for/in

- ◆ **for (i in x) {..bloque de instrucciones..}**
 - itera en todas las propiedades del objeto **x**
- ◆ El **nombre** de propiedad y su **contenido** se referencian con **"i"** y **"x[i]"**
 - **"i"** contiene el nombre de la propiedad en cada iteración
 - **"x[i]"** representa el valor de la propiedad **"i"**
 - ♦ Dentro de la sentencia for debe utilizarse la notación array



Anexo 3 Parcial 3

Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Sintaxis de la sentencia for/in

- ◆ La sentencia comienza por **for**
- ◆ Sigue la condición (**i in obj**)
 - debe ir entre **paréntesis (...)**
- ◆ Los bloques de más de 1 sentencia
 - deben delimitarse con **{...}**
- ◆ Bloques de 1 sentencia
 - pueden omitir **{..}**, pero mejoran la legibilidad delimitados con **{..}**

```
// Utilizar notación array para
// acceder a propiedades: obj[i]

for (i in obj) {
  z = z + obj[i];
  obj[i] = "inspected";
}

// En bloques de solo 1 sentencia
// {...} es opcional
//   -> pero se recomienda usarlo

for (i in obj) {
  z = z + obj[i];
}

// Estas 2 formas son equivalentes
// pero menos legibles

for (i in obj) z = z + obj[i];

for (i in obj)
  z = z + obj[i];
```

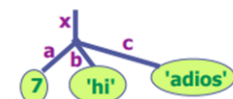

Anexo 3 Parcial 3 Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Sentencia for/in

Práctica guiada 9

- ◆ En el ejemplo se utiliza `for (i in x) {...}`
 - para mostrar en una página Web
 - ◆ el contenido de las propiedades de un objeto



```
<!DOCTYPE html><html>
<head><meta charset="UTF-8"></head>
<body>
<h3>Sentencia for/in:</h3>

<script type="text/javascript">
    var x = {a:7, b:'hi', c:'adios'};

    var i;
    for (i in x) {
        document.write("Propiedad " + i + " = " + x[i] + "<br>");
    }
</script>
</body>
</html>
```

Sentencia for/in:

Propiedad a = 7
Propiedad b = hi
Propiedad c = adios

Anexo 3

Parcial 3

Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Práctica guiada 10

window.screen

Screen

Propiedad	Valor
availWidth	= 2514
availHeight	= 1418
availTop	= 22
availLeft	= 46
pixelDepth	= 24
colorDepth	= 24
width	= 2560
height	= 1440

```

<!DOCTYPE html>
<html>
<head>
<title>DOM</title>
<meta charset="UTF-8">
</head>
<body>

<h2> Screen </h2>

  <!-- tabla con propiedades de screen -->
  <table id="tabla">
    <tr><th> Propiedad </th><th> Valor </th></tr>
  </table>

  <script>
  var i, tabla = document.getElementById("tabla");

  for (i in screen){ //cada iteración genera una fila de la tabla
    tabla.innerHTML+= "<tr><td>" + i + "</td><td> = " + screen[i] + "</td></tr>";
  }
  </script>
</body>
</html>
  
```

25

Anexo 3 **Parcial 3** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Eventos periódicos con setInterval(...)

- ◆ JavaScript tiene una función **setInterval (..)**
 - para programar eventos periódicos
- ◆ **setInterval (manejador, periodo_en_milisegundos)**
 - tiene 2 parámetros
 - ♦ **manejador**: función que se ejecuta al ocurrir el evento
 - ♦ **periodo_en_milisegundos**: tiempo entre eventos periódicos

Anexo 3

Parcial 3

Módulo III. Desarrolla aplicaciones Web
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente

Práctica guiada 11

Reloj

- ◆ Utilizamos la función
 - **setInterval(manejador, T)**
 - ◆ para crear un reloj
- ◆ Cada segundo se muestra
 - El valor de reloj del sistema



```
<!DOCTYPE html>
<html>
<head><title>Reloj</title>
  <meta charset="UTF-8">

<script type="text/javascript">

function mostrar_fecha( ) {
  var cl = document.getElementById("fecha");
  cl.innerHTML = new Date( );
}
</script>

</head>

<body>
<h2>Reloj</h2>

<div id="fecha"><div>

<script type="text/javascript">
  mostrar_fecha();// muestra fecha al cargar
                  // actualiza cada segundo
  setInterval(mostrar_fecha, 1000);
</script>
</body>
</html>
```

Anexo 3 **Parcial 3** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Eventos DOM

◆ Los eventos DOM se asocian a elementos HTML

- como atributos: 'onclick', 'ondblclick', 'onload',
 - ♦ donde el manejador es el valor asignado al atributo

◆ Ejemplo:

- ``
 - ♦ Código del manejador: `"this.src='img2.png'"` (valor del atributo)
 - `this` referencia el objeto DOM asociado al manejador

◆ Tutorial:

- https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Event_attributes

Anexo 3 **Parcial 3** Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Eventos definidos directamente en Javascript

- ◆ Los manejadores de eventos se pueden definir con
 - **objeto.addEventListener(evento, manejador)**
- ◆ También se pueden definir como propiedades
 - **objeto.evento = manejador**
 - ◆ objeto: objeto DOM al que se asocia el evento
 - ◆ evento: nombre (onload, onclick, onmouseover, etc.)
 - ◆ manejador: función ejecutada al ocurrir un evento
- ◆ Ejemplos
 - `img.addEventListener("onclick", function() {... código ...})`
 - `img.onclick=function() {... código...}`

Anexo 3

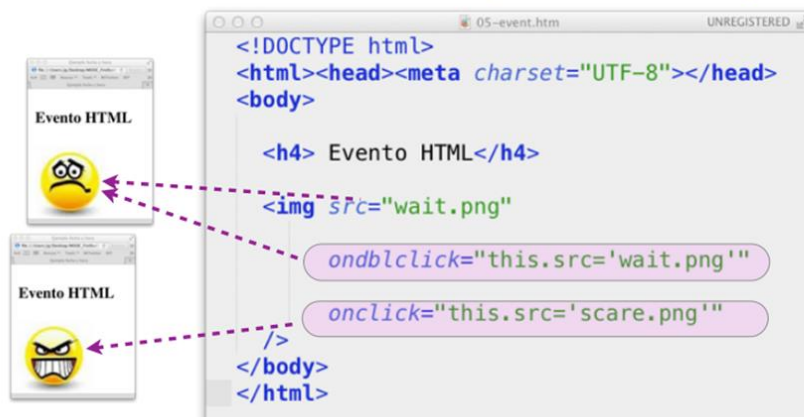
Parcial 3

Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente

Práctica guiada 12

Eventos en HTML

- ◆ Definimos 2 manejadores de evento en elem. `<img ....>`
 - Atributo **onclick**: muestra el icono enfadado
 - Atributo **ondblclick**: muestra el icono pasivo
- ◆ **this.src** referencia atributo **src** de `<img ..>`
 - **this** referencia objeto DOM asociado: `<img ..>`



Anexo 3 Parcial 3 Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Práctica guiada 13

- ◆ Los manejadores de evento se definen ahora en un script separado
 - El objeto `<img ...>` se identifica desde JavaScript con `getElementById(..)`
 - manejador se añade con método: `object.addEventListener(event, manejador)`
 - Actualmente se recomienda usar siempre `addEventListener(...)`

Evento como propiedad



```
<!DOCTYPE html>
<html>
<head><meta charset="UTF-8"></head>
<body>

  <h4>Evento JS</h4>

  <script type="text/javascript">
    var i=document.getElementById('i1');

    i.addEventListener('dblclick', function(){i.src='wait.png'});

    i.addEventListener('click', function(){i.src='scare.png'});

  </script>
</body>
</html>
```

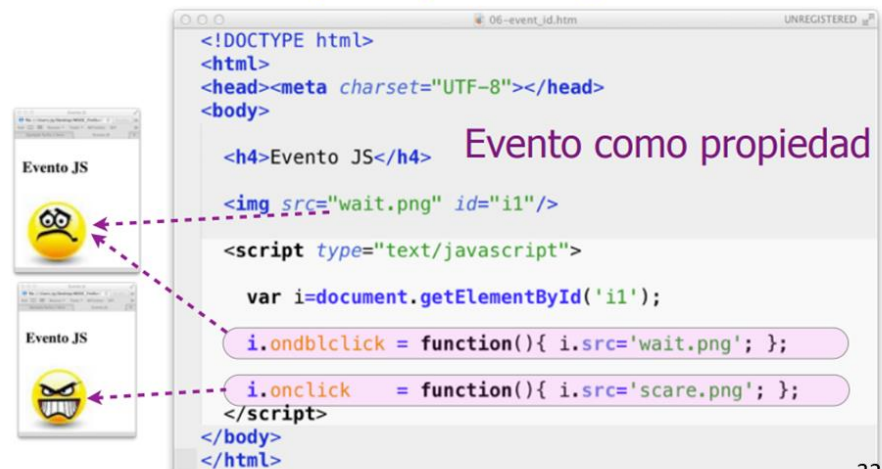
31

Anexo 3 Parcial 3 Módulo III. Desarrolla aplicaciones Web.

Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Práctica guiada 14

- ◆ Los manejadores de evento se definen también así en un script separado
 - El objeto `<img ...>` se identifica desde JavaScript con `getElementById(..)`
 - Sintaxis de los manejadores: `object.event= manejador`



Anexo 3 Parcial 3 Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Práctica guiada 15

- El script pasa a la cabecera, se **separa del documento HTML**
 - El código se mete en la función **inicializar()**, que se ejecuta al ocurrir **onload**
 - onload** ocurre con la página HTML ya cargada y el objeto DOM construido

Evento onload



Evento onload



```

<!DOCTYPE html>
<html>
<head><title>Evento onload</title><meta charset="UTF-8">

<script type="text/javascript">
  function inicializar() {
    var i=document.getElementById('i1');
    i.ondbclick = function () {i.src='wait.png'};
    i.onclick = function () {i.src='scare.png'};
  }
</script>
</head>

<!-- El arbol DOM ya esta construido al ocurrir onload -->
<body onload="inicializar()">

  <h4>Evento onload</h4>

</body>
</html>

```

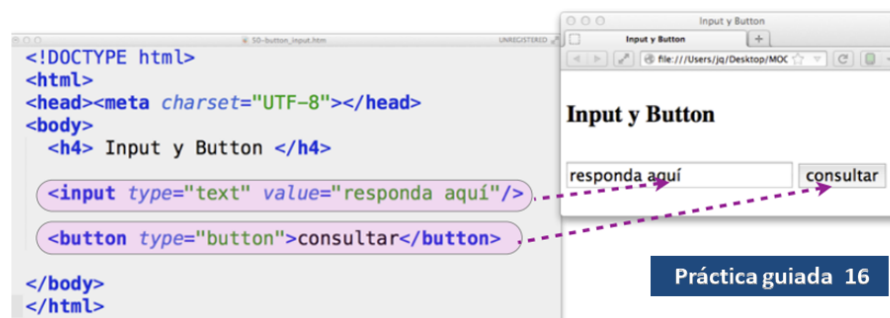
Evento onload

33

Anexo 3 **Parcial 3** Módulo III. Desarrolla aplicaciones Web.
Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.

Entradas y botones Formularios

- ◆ **Entrada:** un cajetín en pantalla para introducir texto en una aplicación
 - Se define con `<input type=text ...>`
 - ♦ el atributo `value="texto"` representa en texto dentro del cajetín
- ◆ **Botón:** elemento gráfico que invita a hacer clic
 - Se define con `<button type=button ...>nombre</button>`



The image shows two side-by-side windows. The left window is an HTML editor titled '5D-button_input.htm' showing the following code:

```
<!DOCTYPE html>
<html>
<head><meta charset="UTF-8"></head>
<body>
<h4> Input y Button </h4>
<input type="text" value="responda aquí"/>
<button type="button">consultar</button>
</body>
</html>
```

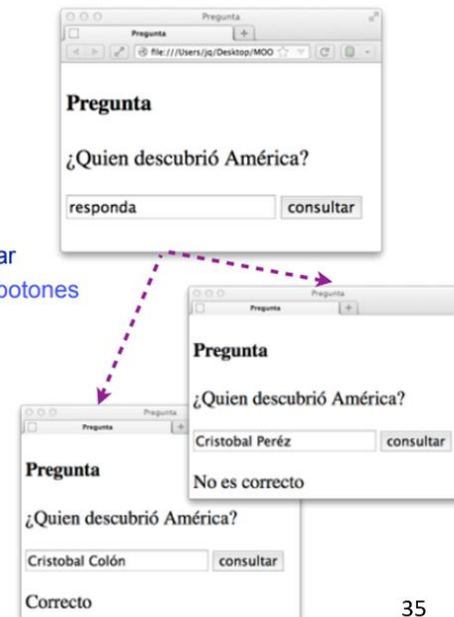
The right window is a web browser titled 'Input y Button' showing the rendered output. It displays the text 'Input y Button' followed by a text input field containing 'responda aquí' and a button labeled 'consultar'. Dashed purple arrows connect the code in the editor to the corresponding elements in the browser: one arrow points from the `<input type="text" value="responda aquí"/>` line to the input field, and another points from the `<button type="button">consultar</button>` line to the button.

Práctica guiada 16

Anexo 3 **Parcial 3** **Módulo III. Desarrolla aplicaciones Web.** **Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente.**

Ejemplo Pregunta

- ◆ Esta WebApp plantea la pregunta
 - ¿Quien descubrió América?
 - ♦ para ilustrar como interaccionar
 - a través de formularios y botones
- ◆ Escribir la respuesta en el cajetín
 - y pulsar el boton **“consultar”**
 - ♦ para saber si es correcto
- ◆ Según sea la respuesta se responde
 - **“Correcto”** o **“No es correcto”**



Pregunta
 ¿Quien descubrió América?
 responda consultar

Pregunta
 ¿Quien descubrió América?
 Cristobal Pérez consultar

Pregunta
 ¿Quien descubrió América?
 Cristobal Colón consultar

Correcto

Anexo 3 Parcial 3 Módulo III. Desarrolla aplicaciones Web. Submódulo 2. Desarrolla aplicaciones que se ejecutan en el cliente

```

<!DOCTYPE html>
<html><head><title>Pregunta</title><meta charset="UTF-8">
<script type="text/javascript">

function res() {
  var respuesta = document.getElementById('respuesta');
  var resultado = document.getElementById('resultado');

  if (respuesta.value === "Cristobal Colón")
    resultado.innerHTML = "Correcto";
  else
    resultado.innerHTML = "No es correcto";
}

</script>
</head>
<body>
  <h4> Pregunta </h4>
  <p> ¿Quien descubrió América? </p>

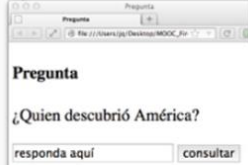
  <input type="text" id="respuesta" value="responda aquí">
  <button type="button" onclick="res()">consultar</button>

  <p><div id="resultado" /></p>

</body>
</html>

```

Pregunta




Correcto

Validación de campos

Práctica guiada 17



EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

Subsecretaría de Educación Media Superior
Dirección General de Educación Tecnológica Industrial y de Servicios
Dirección Académica e Innovación Educativa
Subdirección de Innovación Académica

Aprendizajes esenciales			
Carrera:	Programación	Semestre:	Cuarto
Módulo/Submódulo:	Módulo III. Desarrolla aplicaciones Web. Submódulo 3. Desarrolla aplicaciones que se ejecutan en el servidor.		
Aprendizajes esenciales o Competencias esenciales 1er parcial	Estrategias de Aprendizaje	Productos a Evaluar	
- Ejecuta su examen de diagnóstico - Conoce la instalación del servidor web apache-Xampp. - Selecciona el lenguaje de programación PHP. - Realiza ejemplos propuestos de problemas cotidianos en programación en PHP.	- Contestar correctamente el examen de diagnóstico (Anexo 1) - Revisa la guía para la instalación del servidor web apache- Xampp, reflexiona y con su propia interpretación genera un procedimiento, que incluya la instalación, sugerencias y conclusiones. (Anexo 2) - Identifica las características y la sintaxis del lenguaje php. (Anexo 3) - Analiza los ejemplos propuestos en php, y resuelve los ejercicios planteados.	- Examen escrito en su cuaderno. - Reporte escrito. - Resumen escrito. - Ejercicios realizados en su cuaderno.	
Aprendizajes esenciales o Competencias esenciales 2º parcial	Estrategias de Aprendizaje	Productos a Evaluar	
Identifica la metodología de programación en php.	✓ Seleccionar las Estructuras de control, para la solución de problemas cotidianos de acuerdo a la situación. (Anexo 4) ✓ Las condiciones: If Switch ✓ Los bucles For While Do while Foreach	- Resumen - Ejercicios realizados	
Aprendizajes esenciales o Competencias esenciales 3er parcial	Estrategias de Aprendizaje	Productos a Evaluar	

• Diseño de proyecto final mediante las etapas: • Creación de BD en servidor • Diseño Front-end (formularios) • Diseño Back-end (conexión con bd y ejecución en servidor).	Con base en el ejemplo mostrado para Altas de registros, bajas de registros, consulta de registros, modificación de registros y listado de registros. El alumno deberá realizar en su libreta un sistema semejante considerando: 1.- La base de datos tendrá los campos de: a) Matricula b) Apellido paterno c) Apellido materno d) Nombres e) Semestre f) Especialidad 2.- Dibuja en tu libreta las interfaces de cada uno de los programas de Altas, bajas, consulta, modificación y listado de registros. 3.- Escribe el código en tu libreta de cada uno de los programas de cada una de las interfaces (Formularios). Se anexa documento con el código básico de cada uno de los programas de Altas, Bajas, Consultas, Modificaciones y listado de registro de ejemplo. (Anexo 5).	Diseño y Codificación de las interfaces en su libreta de los programas de: • Altas de Registros • Bajas de Registros • Consulta de Registros • Modificación de Registros • Listados de Registros
---	--	---

Anexos del Submódulo 3

Anexo 1 Examen de diagnóstico

Examen de diagnóstico.

Alumno: _____ Semestre: _____ Grupo: _____

Submódulo 3: Desarrolla Aplicaciones que se ejecutan en el servidor Fecha: _____

FACILITADOR: _____

I.- CONTESTA BREVEMENTE LAS SIGUIENTES INTERROGACIONES

1. ¿Escribe la definición de Internet?
2. ¿Explica que es la WEB?
3. ¿Define Qué es una página WEB? (Página estática: Página dinámica:)
4. ¿Cuáles son los servicios que brinda Internet?
5. Menciona por lo menos una característica de la WEB 1.0, 2.0, 3.0 y 4.0
6. ¿Qué es una dirección IP?
7. ¿Define el concepto de un servidor web?

8.- Contesta brevemente a qué se refieren los siguientes métodos:

\$_SERVER
\$_GET

\$_POST

9.- Como se declaran las variables en PHP

10.- Cuales son las sentencias que se utilizan en PHP para escribir mensajes en la pantalla

II.- DISEÑAR EL SIGUIENTE FORMULARIO: Y CALCULAR EN PHP: ¿cuánto pagaría un cliente, si elige un plan de pagos de acuerdo a los siguientes datos?

12 meses = 12 % de interés anual

24 meses = 17 % de interés anual.

Menor de 12 meses = no paga interés.

Solo se acepta un máximo plan de 24 meses

Anexo 2

Desarrollo de sitios web con PHP



Tema 1: Introducción

1. Introducción a PHP
2. Instalación de Apache
3. Instalación de PHP
4. Instalación de una distribución de Apache: XAMPP
5. Entornos de desarrollo para PHP
6. Recursos de PHP

Introducción a PHP

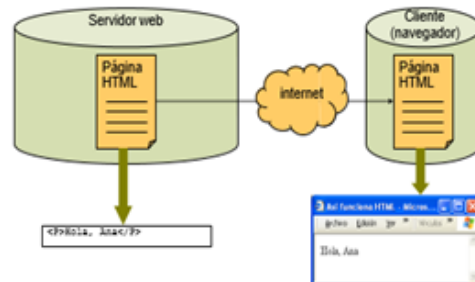
- Lenguajes de script

- PHP es un lenguaje de script del lado del servidor. Otros lenguajes similares son ASP, JSP o ColdFusion
- Los scripts PHP están incrustados en los documentos HTML y el servidor los interpreta y ejecuta antes de servir las páginas al cliente
- El cliente no ve el código PHP sino los resultados que produce



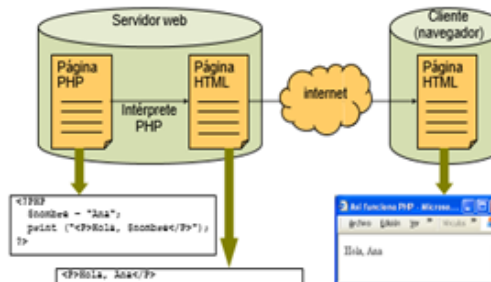
Introducción a PHP

- ¿Cómo funciona PHP? (1)



Introducción a PHP

- ¿Cómo funciona PHP? (2)



Introducción a PHP

- Breve historia de PHP

- Creado por Rasmus Lerdorf para uso personal en 1994
- PHP = PHP: Hypertext Preprocessor
- Versión actual: PHP 5
- Es un módulo que se añade al servidor web y fue concebido inicialmente para Apache

- ¿Por qué PHP?

- Por sus ventajas: es potente, fácil de aprender, de libre distribución, permite el acceso a bases de datos y otras funcionalidades orientadas a la red
- Dispone de abundante soporte en la Web

Introducción a PHP

- **Requisitos**
 - Servidor web Apache Xampp
 - [tutoriales / www.apachefriends.org/es/download.html](http://tutoriales.apachefriends.org/es/download.html)
 - con el módulo PHP (www.php.net) si se desea crear y la base de datos MySQL (www.mysql.com) páginas dinámicas
 - Herramientas para la gestión de MySQL, como PHPMyAdmin (www.phpmyadmin.net)
 - Editores de PHP como DevPHP (www.sourceforge.net), Eclipse (www.eclipse.org) o Aptana Studio (www.aptana.com)
 - Manuales de PHP y MySQL
- **Otras utilidades**



7

Instalación de XAMPP

- ¿Qué es XAMPP?
- XAMPP es una distribución de Apache que incluye MySQL, PHP y otras herramientas para el desarrollo de aplicaciones web, como phpMyAdmin
- XAMPP es gratuito y fácil de instalar: basta con descargar el archivo y extraerlo
- XAMPP es multiplataforma: existen versiones para Windows, Linux y Mac OS
- Precaución: la configuración por defecto de XAMPP no es segura y no es adecuada para un entorno de producción. El paquete incluye una herramienta para obtener una configuración más segura



8

Instalación de XAMPP

- **Instalación y configuración de XAMPP**
 - Pasos:
 - Descargar
 - Instalar
 - Probar



9

Instalación de XAMPP

- **Instalación y configuración de XAMPP. 1: descargar**
 - Conectarse a <http://www.apachefriends.org/es/xampp.html>
 - Seleccionar la plataforma adecuada
 - En el caso de Windows existen dos versiones: la normal (XAMPP) y la reducida (XAMPP)
 - Para instalar la versión XAMPP para Windows,
 - Seleccionar Download > XAMPP
 - Elegir el archivo EXE



10

Instalación de XAMPP

- **Instalación y configuración de XAMPP. : probar**
 - Ejecutar XAMPP haciendo doble clic sobre el icono xampp_control
 - Arrancar los módulos Apache y MySQL. Aparece el rótulo Running al lado de ambos
 - Pulsar el botón Admin. Aparecerá la página de inicio del servidor
 - Para parar XAMPP, pulsar el botón Exit



11

Instalación de XAMPP

- **Configuración segura de XAMPP**
 - Arrancar XAMPP y cargar la página de inicio
 - Seleccionar la opción Chequeo de seguridad
 - Pulsar el enlace recomendado para solucionar los problemas de seguridad
 - Establecer una contraseña para el administrador (root) de MySQL (por defecto está en blanco)
 - Crear un usuario con contraseña para proteger el acceso a la carpeta de XAMPP



12



Entornos de desarrollo para PHP

■ ¿Cómo desarrollar un proyecto en PHP?

- Los archivos PHP son ficheros de texto y se pueden crear con cualquier editor de texto plano, como e notepad de Windows
- Es mucho más conveniente utilizar entornos de desarrollo que permiten editar el código más cómodamente, y además proporcionan funciones como la detección y corrección de errores, visualización de las páginas en el navegador, ayuda sensible al contexto y gestión de todos los recursos asociados al proyecto



13

Tema 1: Introducción

1. Introducción a PHP
2. Instalación de Apache
3. Instalación de PHP
4. Instalación de una distribución de Apache: XAMPP
5. Entornos de desarrollo para PHP
6. Recursos de PHP

14

Desarrollo de sitios web con PHP y MySQL



Introducción

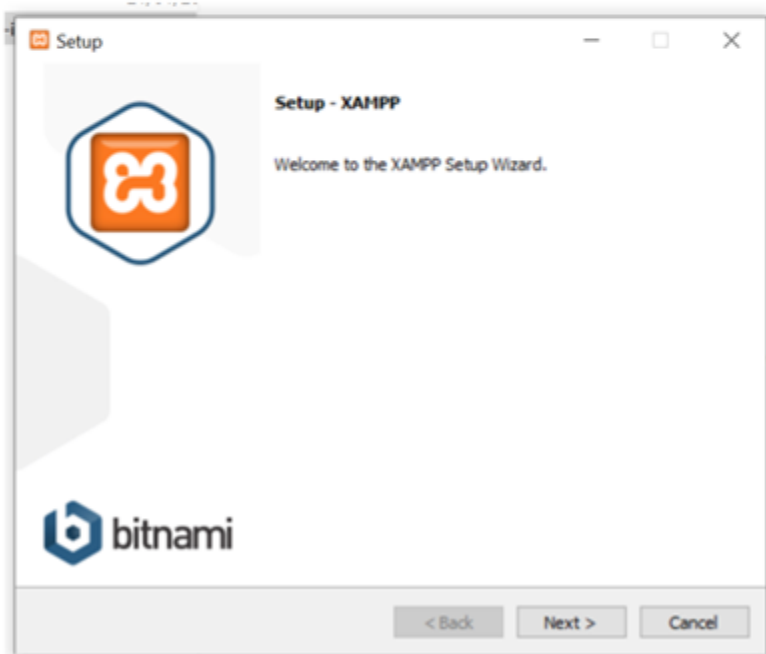
15

GUIA DE INSTALACION SERVIDOR WEB

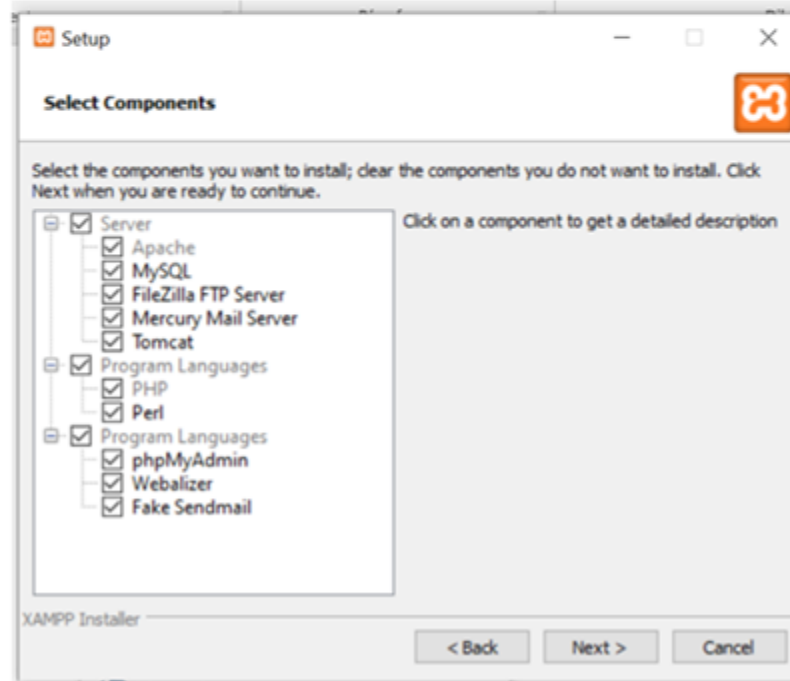
1.- EJECUTAR LA SIGUIENTE APLICACION

 xampp-windows-x64-7.4.11-0-VC15-installer

PASO 1



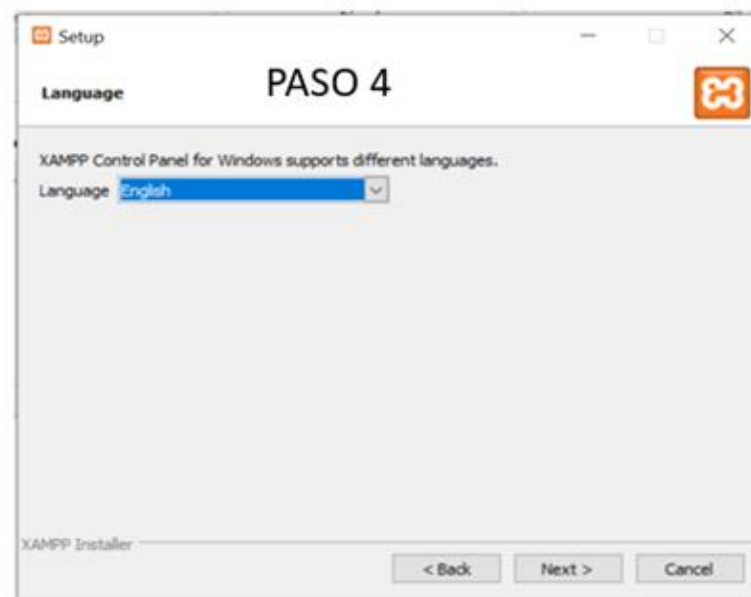
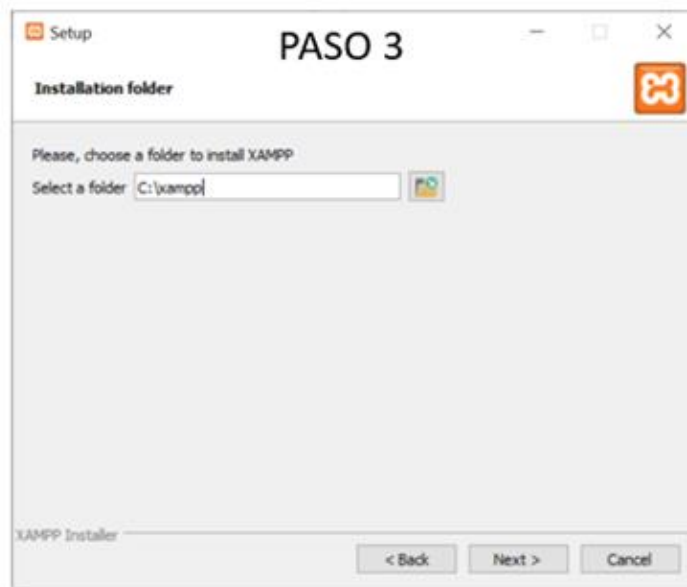
PASO 2

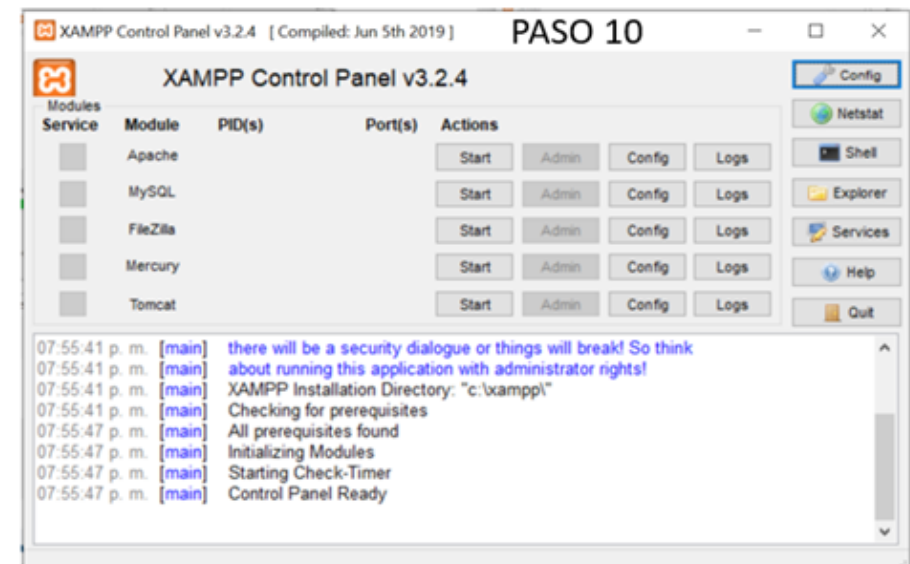
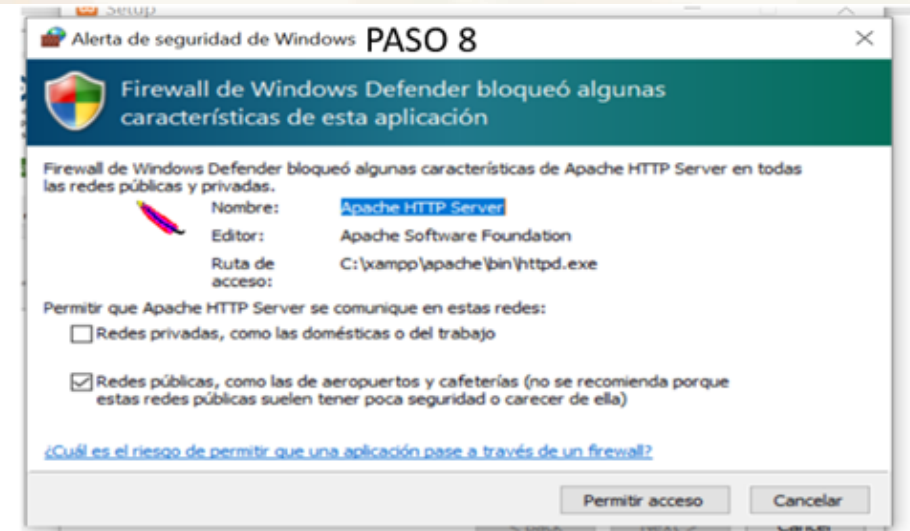




EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

Subsecretaría de Educación Media Superior
Dirección General de Educación Tecnológica Industrial y de Servicios
Dirección Académica e Innovación Educativa
Subdirección de Innovación Académica







PASO 11 Y ULTIMO.

XAMPP Control Panel v3.2.4 [Compiled: Jun 5th 2019]

XAMPP Control Panel v3.2.4

Service	Module	PID(s)	Port(s)	Actions
☐	Apache	6344 9172	80, 443	Stop Admin Config Logs
☐	MySQL	3008	3306	Stop Admin Config Logs
☐	FileZilla			Start Admin Config Logs
☐	Mercury			Start Admin Config Logs
☐	Tomcat			Start Admin Config Logs

Config Netstat Shell Explorer Services Help Quit

07:55:47 p. m. [main] All prerequisites found
07:55:47 p. m. [main] Initializing Modules
07:55:47 p. m. [main] Starting Check-Timer
07:55:47 p. m. [main] Control Panel Ready
07:56:50 p. m. [Apache] Attempting to start Apache app...
07:56:50 p. m. [Apache] Status change detected: running
07:56:52 p. m. [mysql] Attempting to start MySQL app...
07:56:52 p. m. [mysql] Status change detected: running

Anexo 3

PHP

El lenguaje PHP (cuyo nombre es acrónimo de PHP: Hypertext Preprocessor) es un lenguaje interpretado con una sintaxis similar a la de C++ o JAVA. Aunque el lenguaje se puede usar para realizar cualquier tipo de programa, es en la generación dinámica de páginas web donde ha alcanzado su máxima popularidad. En concreto, suele incluirse incrustado en páginas HTML (o XHTML), siendo el servidor web el encargado de ejecutarlo.

1. Sintaxis básica

XHTML (Extensible Hypertext Markup Language) es un lenguaje de etiquetas. Es el sucesor de HTML y se basa en la sintaxis de XML. Asegura la compatibilidad tanto en equipos clásicos como en smartphones.

Ya conoce las etiquetas <html>, <body>, <head>...

Escriba PHP entre dos etiquetas. Se definen de la siguiente manera:

<?php: indica el comienzo del código PHP

?>: indica el final del código PHP

Una instrucción siempre termina con un punto y coma.

Ejemplo

```
<?php  
    echo '<p>Hola!</p>';  
?>
```

También puede escribir este código en una sola línea:

```
<?php echo '<p>Hola!</p>'; ?>
```

2. Inserción de etiquetas PHP en el código XHTML

Puedes insertar un código PHP en cualquier ubicación del código XHTML.

```
<html>  
<head>
```

```
<title>Ejemplo de página PHP</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
</head>
<body>
  Hola, hace <?php echo 'como estas'; ?>
</body>
</html>
```

3. Envío de datos al servidor Web

Existen varias instrucciones para enviar datos al servidor, es decir, para insertar código HTML en una página Web. La primera instrucción es echo y se escribe de la siguiente manera:

```
<?php echo 'texto'; ?>
```

También puede escribir este código:

```
<?php echo "texto"; ?>
```

O bien:

```
<?php echo('texto'); ?>
```

La segunda instrucción es print y se escribe de la siguiente manera:

```
<?php print('texto'); ?>
```

Por tanto, print equivale a echo.

Existen otras variantes de print:

printf(): muestra una cadena de caracteres formateada.

sprintf(): devuelve una cadena formateada.

vprintf(): muestra una cadena formateada.

sscanf(): analiza una cadena con ayuda de un formato.

fscanf(): analiza un archivo en función del formato.
flush(): vacía los búferes de salida.

También puede escribir varias instrucciones en la misma línea, siempre y cuando vayan separadas por punto y coma.

```
<?php echo 'texto'; ?> equivale a <?php echo 'tex'; echo 'to'; ?> y a  
<?php echo 'tex';  
    echo 'to';  
?>
```

4. Inserción del código XHTML con la instrucción echo

La función echo permite insertar cualquier código HTML, por ejemplo:

```
<?php echo '<table><tr><td>texto</td></tr></table>'; ?>
```

Y como resultado inserta una tabla HTML.

También puede insertar una imagen de la siguiente manera:

```
<?php echo ''; ?>
```

Por tanto, puede escribir una página Web completa con la instrucción echo.

```
<?php  
  
echo '<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//ES" , "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">';  
echo '<html xmlns="http://www.w3.org/1999/xhtml">';  
echo '<head>';  
echo '<title>PHP </title>';  
echo '</head>';  
echo '<body>';  
echo '<p>';  
echo "Hola \$nombre !<br />";
```



```
echo 'La fecha es 20 de Enero del 2021 .'; //  
echo '</p>';  
echo '</body>';  
echo '</html>';
```

?>

Las variables

1. Asignación

Una variable es una información que se almacena temporalmente en la memoria, es decir, es una zona de la memoria que almacena información en una página PHP y que se destruye automáticamente cuando la página ya no se ejecuta.

Una variable PHP comienza siempre con \$, seguida de una letra y de una secuencia de letras, cifras o del signo _.

Por ejemplo, \$edad.

Atención: PHP distingue entre mayúsculas y minúsculas, por lo que \$nombre es distinto de \$Nombre.

Una variable siempre tiene un nombre y un valor.

Por ejemplo, \$edad = 25, el valor 25 se asigna a la variable \$edad gracias al signo =.

No es necesario definir y buscar el tipo de variable. Se hace automáticamente.

\$dia = 24; //Se declara una variable de tipo integer.

\$sueldo = 758.43; //Se declara una variable de tipo double.

\$nombre = "juan"; //Se declara una variable de tipo string (cadena).

\$salida = true; //Se declara una variable boolean.

De este modo, puede escribir:

```
<?php
```

```
$edad = 25; //variable de tipo numérico
```

```
//después
```

```
$edad = '25'; //variable de tipo texto
```

?>

1. Operadores aritméticos

+ Suma dos valores

- Resta dos valores (o pasa a negativo un valor)

* Multiplica dos valores

/ Divide dos valores

% Resto de dividir dos valores

2. Tipos de variables

Hay dos categorías de variables:

Escalar:

Los números enteros llamados integer son 1, 2, 3... y los números negativos, -1, -2, -3...

Los números decimales llamados float son los números positivos o negativos con comas (1.35665).

Atención: el punto se utiliza como separador.

La cadena de caracteres string: cualquiera con dobles comillas ("hola") o comillas simples ('hola').

Los booleanos: solo tienen dos tipos de valores: verdadero o falso, clasificados como true o false.

Compuesta:

3. La concatenación

Es un conjunto de cadena de caracteres. PHP permite la concatenación usando la coma o el punto.

```
<?php  
echo 'hola '. 'lee esta ayuda';  
?>
```

Equivale a:

```
<?php  
echo 'hola ',lee esta ayuda';  
?>
```

Da como resultado:

Hola lee esta ayuda

Si quiere concatenar la cadena "hola" y "aquí hay un apóstrofo '", no podrá escribir:

```
<?php  
echo 'hola '. 'aquí hay un apóstrofo ''';  
?>      marcaría error.
```

Ejercicios

- 1.- Diseña un código donde muestre tu nombre completo.
- 2.- Diseña un código que indique los siguientes datos : calle, numero de calle, colonia, municipio y teléfono.
- 3.- Diseña un código para almacenar 2 números y realice una suma.

Anexo 4

ESTRUCTURAS DE CONTROL

1. Operadores de comparación(relacionales)

- ==** Comprueba si dos números son iguales
- !=** Comprueba si dos números son distintos
- >** Mayor que, devuelve true en caso afirmativo
- <** Menor que, devuelve true en caso afirmativo

>= Mayor o igual
<= Menor o igual

1. Operadores lógicos

! Operador NO o negación. Si era true pasa a false y viceversa
And Operador Y, si ambos son verdaderos vale verdadero
or Operador O, vale verdadero si alguno de los dos es verdadero
xor Verdadero si alguno de los dos es true pero nunca ambos
&& True si ambos lo son
|| True si alguno lo es

Quando se pretende que el script tome un camino concreto en determinados casos y otro diferente si las condiciones de ejecución difieren, se utiliza el conjunto de instrucciones: if, else y elseif.

La estructura base es la siguiente:

```
if (Condición) {  
    Instrucción 1;  
    Instrucción 2;  
}  
else {  
    Instrucción a;  
    Instrucción b;  
}
```

EJEMPLO.

If

```
<?php  
$tmp = 1;  
$variable = $tmp>5 ? 'Es mayor a 5' : 'No es mayor a 5';  
echo $variable;  
?>
```

```
<?php
$nombre = 'Miguel'; //declaración de la variable $nombre
if ($nombre == 'Velez') //comprueba la variable $nombre
{
    echo 'Bienvenido';
}
else
{
    echo 'denegado';
}
?>
```

Resultado es igual a “denegado”

```
<?php
$edad = 18; //declaración de la variable $edad
if ($edad >= 18) //comprueba la variable $edad
{
    echo 'Eres mayor de edad ya puedes votar';
}
else
{
    echo 'Eres menor de edad';
}
?>
```

Resultado es igual a “Eres mayo de edad ya puedes votar”

Condicionales compuestas if elseif

if (Condicion 1)

```
{
    Instrucción a;
}
```

elseif (Condicion 2)

```
{
```

Instrucción b;

}

else

{

Instrucción c;

}

<?php

\$nombre = 'Miguel'; //declaración de la variable \$nombre

if (\$nombre == 'velez') //comprueba la variable \$nombre

{

 echo 'Bienvenido';

}

else if (\$nombre == 'Miguel') //comprueba la variable \$nombre

{

 echo 'Hola';

}

else

{

 echo 'Denegado';

}

?>

Resultado es igual a “Hola”

Sentencia Switch

1. Un switch busca dentro de los “case” el valor de la variable o expresión evaluada (generalmente se evalúan variables).
2. Si lo encuentra, ejecuta el código correspondiente.
3. Si no lo encuentra, ejecuta el código por default.
4. *Observaciones:* El case por default es opcional.
5. Es importante poner “break;” al final de cada bloque de código dentro de cada case para que el switch no siga comparando al valor de la variable con los case que le siguen al correcto.



Sintaxis:

```
switch(expresión) {  
  
    case "a": //Código a ejecutar;  
        break;  
    case "b": //Código a ejecutar;  
        break;  
    case "c": //Código a ejecutar;  
        break;  
    default: //Código a ejecutar por default.  
}
```

Ejemplo

```
<?php  
$nombre = 'miguel'; //declaración de la variable $nombre  
switch ($nombre) //comprueba la variable $nombre  
{  
    case 'miguel':  
        echo 'Hola';  
        break;  
    case 'Juan':  
        echo 'Hasta pronto';  
        break;  
}  
?>
```

Resultado = Hola

La instrucción `break` provoca la salida del `switch` y si `$nombre` es igual a "miguel" el código ejecutará `echo "Hola"` y `break`, y saldrá del `switch` sin comprobar "Juan".

```
<?php  
$edad = 25; //declaración de la variable $edad
```



```
switch ($edad) //comprueba la variable $edad
{
    case 20:
        echo "Tiene 25 años.";
        break;
    case 25:
        echo "Tiene 25 años.";
        break;
    default:
        echo "No tiene 20 ni 25 años.";
}
?>
```

Resultado = tiene 25 años

Los bucles

1. For

Un bucle permite repetir n veces la ejecución de un código.

Por ejemplo, si quiere mostrar diez veces «Hola», solo tiene que escribir el bucle for.

```
<?php
for ($i = 1; $i <= 10; $i++)
{
```

```
    echo 'Hola <br />';  
}  
?>
```

La variable \$i representa el contador del bucle.

Por tanto, la sintaxis es:

```
for ($i=número inicial; $i <= número final; aumento)  
{  
    instrucciones  
}
```

\$i++ es igual a \$i=\$i+1 y representa el aumento de \$i. Puede escribir \$i=\$i+2 para aumento \$i=\$i-1 para disminuir.

Por ejemplo, puede escribir los números de 100 a 150 con el siguiente código:

```
<?php
```

```
for ($i = 100; $i <= 150; $i++)  
{  
    echo $i.'<br />';  
}  
?>
```

La instrucción echo \$i.'
'; se repite 50 veces y \$i aumenta en 1 cada vez.

 permite saltar una línea entre cada número para no tener que mostrarlos todos.

La instrucción break permite detener el bucle.

Por ejemplo, si quiere mostrar cinco veces «Hola», solo debe escribir un bucle for:

```
<?php
```

```
for ($i = 1; $i <= 10; $i++)  
{  
    echo 'Hola <br />';  
    if ($i == 5) {  
        break;  
    }  
}
```

```
?>
```

Da como resultado:

Hola
Hola
Hola
Hola
Hola
Hola

El bucle se detiene cuando \$i es igual a 5 (y no a 10).

2. While

El bucle while significa «mientras que», es decir, el bucle se ejecutará siempre y cuando una condición sea verdadera.

Por ejemplo, para mostrar diez veces «Hola», solo debe escribir un bucle while:

```
<?php
$i = 1;
while ($i <= 10)
{
    $i=$i+1;
    echo 'Hola <br />';
}
?>
```

La variable \$i representa el contador del bucle. Pero mientras \$i sea inferior o igual a 10, se repetirá el bucle.

Por lo tanto, la sintaxis es:

```
$i=número inicial
while ($i <= número final)
{
    aumento
    instrucciones
}
```

No olvide poner el aumento de \$i en las instrucciones de while; de lo contrario \$i nunca valdrá 10 y tendrá un bucle infinito.

Considera que el valor de salida de \$i se pone antes del bucle y que este valor debe respetar la condición del bucle (\$i <= número final) para entrar en el bucle.

Si escribe:

```
<?php
$i = 11;
while ($i <= 10)
{
    $i=$i+1;
    echo 'Hola <br />';
}
?>
```

Nunca pasará en el bucle porque \$i vale 11 en un principio, no se satisface la condición del bucle.

El bucle while es igual al bucle for; en algunas ocasiones le resultará muy útil si desconoce el número de veces que va a ejecutar un bucle, sobre todo si va a leer el bucle while en la base de datos y la condición de salida del bucle depende del valor leído en la base de datos.

3. Do while

El bucle Do while significa «hacer mientras», es decir, el bucle se ejecutará siempre y cuando una condición sea verdadera. Se diferencia del bucle while en que la expresión se ejecuta al menos una vez.

Por ejemplo, para mostrar diez veces "Hola", debe escribir el bucle Do while:

```
<?php
$i = 1;
do
{
    $i=$i+1;
    echo 'Hola <br />';
} while ($i <= 10)
?>
```

La variable \$i representa el contador del bucle. Pero esta vez debe leer: ejecutar el bucle si \$i es inferior o igual a 10.

Por tanto, la sintaxis es:

```
$i=número inicial
do
{
    aumento
```

instrucciones

} while (\$i <= número final)

```
<?php
$i = 1;
do
{
    $i=$i+1;
    echo 'Hola <br />';
} while ($i <= 10)
?>
```

EJERCICIOS PROPUESTOS

- 1.- Diseña un código para revisar el nombre y contraseña de un usuario para simular el ingreso a un sistema. El mensaje puede ser “acceso autorizado” o “acceso denegado”.
- 2.- Diseña un código para promediar tres calificaciones y verificar si su promedio es aprobado con la mínima de 6 si no es reprobado.
- 3.- Diseñar un código para registrar tres edades y verificar quien es el mayor, el menor o son iguales.
- 4.- Diseñar un código para registrar un día de la semana y que muestre que día de la semana es.
- 5.- Diseñar un código para que muestre 15 veces tu nombre completo, utiliza el bucle que prefieras.

Anexo 5

CODIGOS
SISTEMA PRÁCTICA PHP



ARCHIVO: altas.php

```
<html>
<head>
<title>alumnos</title>
</head>
<body>
<h1>Alta de Alumnos</h1>
<form action="altas2.php" method="post">

Ingrese nombre:
<input type="text" name="nombre"><br><br>
Ingrese direccion:
<input type="text" name="direccion"><br><br>
Ingrese telefono:
<input type="text" name="telefono"><br><br>
<br><br>
<input type="submit" value="Registrar">
</form>
</body>
</html>
```

ARCHIVO: altas2.php

```
<html>
<head>
<title>insertar base de datos</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","12345") or die("Problemas en la conexion");

mysql_select_db("uno",$conexion) or die("Problemas en la seleccion de la base de datos");

mysql_query("insert into alumnos(nombre,direccion,telefono) values
('$REQUEST[nombre]','$REQUEST[direccion]','$REQUEST[telefono]')",
$conexion) or die("Problemas en el select".mysql_error());

mysql_close($conexion);

echo "El alumno fue dado de alta.";
?>
</body>
</html>
```



ARCHIVO: bajas.php

```
<html>
<head>
<title>bajas</title>
</head>
<body>
<form action="borrar2.php" method="post">
Ingrese el nombre del alumno a borrar:
<input type="text" name="nombre">
<br>
<input type="submit" value="borrar">
</form>
</body>
</html>
```

ARCHIVO: borrar2.php

```
<html>
<head>
<title>BAJAS</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","12345") or die("Problemas en la conexion");
mysql_select_db("uno",$conexion) or die("Problemas en la selección de la base de datos");
$registros=mysql_query("select nombre from alumnos where
nombre='".$_REQUEST[nombre]'", $conexion) or
die("Problemas en el select:".mysql_error());
if ($reg=mysql_fetch_array($registros))
{
mysql_query("delete from alumnos where nombre='".$_REQUEST[nombre]'", $conexion) or
die("Problemas en el select:".mysql_error());
echo "Se efectuó el borrado del alumno con dicho nombre.";
}
else
{
echo "No existe un alumno con ese nombre.";
}
mysql_close($conexion);
?>
</body>
</html>
```



ARCHIVO:consulta.php

```
<html>
<head>
<title>consultas</title>
</head>
<body>
<form action="consultas2.php" method="post">
Ingrese el nombre del alumno a consultar:
<input type="text" name="nombre">
<br>
<input type="submit" value="consultar">
</form>
</body>
</html>
```

ARCHIVO: consultas2.php

```
<html>
<head>
<title>consulta2</title>
</head>
<body>
<?php
$conexion=mysql_connect("localhost","root","12345") or die("Problemas en la conexion");

mysql_select_db("uno",$conexion) or die("Problemas en la selección de la base de datos");

$registros=mysql_query("select nombre, direccion, telefono from alumnos where
nombre='$_REQUEST[nombre]",$conexion) or
die("Problemas en el select:".mysql_error());
if ($reg=mysql_fetch_array($registros))
{
    echo "Nombre:". $reg['nombre']."<br>";
    echo "direccion:". $reg['direccion']."<br>";
    echo "telefono:". $reg['telefono']."<br>";
}
else
{
    echo "No existe un alumno con ese nombre.";
}
mysql_close($conexion);
?>
</body>
</html>
```



ARCHIVO: editar2.php

```
<HTML>
<FORM ACTION=editar2.php METHOD=post>
DAME EL NOMBRE A EDITAR:<INPUT TYPE=text NAME=NOMBRE><BR>
<INPUT TYPE=submit NAME=OK VALUE="BUSCAR"><BR>
</FORM>
</HTML>
<?php

if ($OK == "BUSCAR") {
// conexion al servidor de bases de datos
$dbh=mysql_connect ("localhost", "root", "12345") or die ('problema conectando porque :'.
mysql_error());
// seleccionado la base de datos
mysql_select_db ("uno",$dbh);
// preparando la instruccion sql

$q = "Select * from alumnos where nombre= '$NOMBRE'";
```




```
// ejecutando el query select regresa un rowset
$alumnos = mysql_query("$q", $dbh) or die ("problemon con query") ;
// regresando renglon con registro
if($reg = mysql_fetch_row($alumnos))
{
// construyendo forma dinamica
echo "<FORM ACTION=editar2.php METHOD=post>";
// recordar que strings se encadenan con .
echo "NOMBRE:<INPUT TYPE=text NAME=NOMBRE value= \"\".$reg[0].\"\"><BR>";
echo "DIRECCION:<INPUT TYPE=text NAME=DIRECCION value= \"\".$reg[1].\"\"><BR>";
echo "TELEFONO:<INPUT TYPE=text NAME=TELEFONO value= \"\".$reg[2].\"\"><BR>";
//echo "<input type=hidden name=NOMBRE value=$reg[0]>";
echo "<INPUT TYPE=submit NAME=OK VALUE=editar><BR>";
echo "</FORM>";
}
else
{
echo "No existe un alumno con ese nombre.";
}
}
if ($OK == "editar")
{
// conexion al servidor de bases de datos
```



```
$dbh=mysql_connect ("localhost", "root", "12345") or die ('problema conectando porque :'.  
mysql_error());  
// seleccionado la base de datos  
mysql_select_db ("uno",$dbh);  
// preparando la instruccion sql  
$q = "UPDATE alumnos set nombre='$NOMBRE', direccion='$DIRECCION', telefono='$TELEFONO'  
where nombre='$NOMBRE'";  
// ejecutando el query  
mysql_query("$q", $dbh) or die ("problemita con query");  
// avisando  
echo "REGISTRO EDITADO";  
}
```

?>



ARCHIVO: listado.php

```
<html>  
<body>  
<?php  
$link = mysql_connect("localhost", "root", "12345");  
mysql_select_db("uno", $link);  
$result = mysql_query("SELECT * from alumnos ORDER BY nombre", $link);  
// comienza un bucle que leerá todos los registros existentes  
while($row = mysql_fetch_array($result)) {  
// $row es un array con todos los campos existentes en la tabla  
echo "<hr>";  
echo "Nombre: ".$row['nombre']."<br>";  
echo "Apellidos: ".$row['direccion']."<br>";  
echo "Dirección: ".$row['telefono']."<br>";  
} // fin del bucle de instrucciones  
mysql_free_result($result); // Liberamos los registros  
mysql_close($link); // Cerramos la conexion con la base de datos  
echo "<hr>";  
?>  
</body>
```

ARCHIVO: listado1.php

<?

```
//creamos el link de conexion a nuestro servidor
```

```
$link=mysql_connect("localhost","root","12345");
```

```
//seleccionamos la base de datos
```

```
mysql_select_db("uno",$link);
```

```
//ejecutamos la consulta a la base de datos para extraer los registros
```

```
$rows=mysql_query("select * from alumnos ORDER BY nombre");
```

?

| |
|--|
| |
|--|

| | |
 NOMBRE | DIRECCION | TELEFONO |

<?

```
//esta parte es opcional.
```

```
//aqui solo extraemos el total de registros que nos devolvio nuestra consulta
```

```
$totalregistros=mysql_num_rows($rows);
```

```
//iniciamos el recorrido de nuestros registros
```

```
while($row=mysql_fetch_array($rows)){
```

➤

|
 <? echo \$row['nombre'];?> | <? echo \$row['direccion'];?> | <? echo \$row['telefono'];?> |

<?

```
//este es el fin de nuestro ciclo while
```

}

?

?





EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

Subsecretaría de Educación Media Superior
Dirección General de Educación Tecnológica Industrial y de Servicios
Dirección Académica e Innovación Educativa
Subdirección de Innovación Académica

¡Éxito!