



Creación de Páginas Web: HTML 5



ICB
EDITORES

ÍNDICE

Créditos

UNIDAD 1: HTML 5 PARTE 1^a

Tema 1.1: Introducción a HTML y CSS

- 01. Introducción e Historia de Html
- 02. Fundamentos de Html
- 03. Fundamentos de Css
- 04. Lo Que Hemos Aprendido

Tema 1.2: Introducción a HTML5 y CSS3

- 01. Definición de Html5
- 02. Etiquetas de Html5
- 03. Apis para Html5
- 04. Novedades en Css3
- 05. Lo Que Hemos Aprendido

Tema 1.3: Creación de una Web Usando HTML y CSS

- 01. Archivo de Reset Css
- 02. Definición de la Estructura de la Web
- 03. Secciones y Estilos
- 04. Más Estilos
- 05. Prioridades en Css
- 06. Lo Que Hemos Aprendido

UNIDAD 2: HTML 5 PARTE 2^a

Tema 2.1: Etiquetas HTML5

- 01. Repaso a las Características de Html5
- 02. Transformación de Una Web a Html5
- 03. Compatibilidad con Todos los Navegadores
- 04. Lo Que Hemos Aprendido

Tema 2.2: El Canvas de HTML5

- 01. Introducción a Canvas
- 02. Fundamentos de Javascript
- 03. Utilización de Canvas
- 04. Lo Que Hemos Aprendido

Tema 2.3: Drag and Drop

- 01. Características de Drag And Drop
- 02. Ejemplo: Cesta de la Compra con Drag And Drop
- 03. Lo Que Hemos Aprendido

Glosario

Bibliografía

CREACIÓN DE PÁGINAS WEB: HTML 5

Editado por:

ICB. S.L. (Interconsulting Bureau S.L.)

Avda. Ortega y Gasset, 198

P.I. Alameda 29006 – Málaga. España

Tfno: (+34) 952 28 87 67

Web: www.icbeditores.com

Correo electrónico: info@icbeditores.com

Creación de Páginas Web: HTML 5

Autora: Ainoa Celaya Luna

Ingeniera de Telecomunicación, especialidad en Comunicaciones por la Universidad de Málaga. Colegiada N° 17782. Ingeniera Técnica de Telecomunicación, especialidad en Sonido e Imagen por la Universidad de Málaga.

Programadora de Aplicaciones y Páginas Web desde 2003.

Experta en Social Media, ha sido responsable de varias campañas de marketing digital a nivel nacional.

Fundadora de la empresa Lunamic Ingeniería y Comunicación. Obtuvo el Premio Emprendedor 2013 de la Asociación Andaluza de Ingenieros de Telecomunicación.

Ponente en el XX Congreso Iberoamericano de Seguridad Informática.

Locutora de sección de tecnología en Onda Litoral.

2ª Edición

© ICB, S.L. (Interconsulting Bureau S.L.), 2013/09, 1ª edición

© ICB, S.L. (Interconsulting Bureau S.L.), 2015/01, 2ª edición

Reservados todos los derechos de publicación en cualquier idioma.

Según el Código Penal vigente ninguna parte de este o cualquier otro libro puede ser reproducida, grabada en alguno de los sistemas de almacenamiento existentes o transmitida por cualquier procedimiento,

ya se electrónico, mecánico, reprográfico, magnético o cualquier otro, sin autorización previa y por escrito de Interconsulting Bureau S.L., su contenido está protegido por la Ley vigente que establece penas de prisión y/o multas a quienes intencionadamente reprodujeren o plagiaren, en todo o en parte, una obra literaria, artística o científica.

ISBN: 978-84-9021-384-1

ISBN EBOOK: 978-84-9021-647-7

CÓDIGO: MAIC001203

Impreso en España- Printed in Spain

Diríjase a CEDRO (Centro Español de Derechos Reprográficos, www.cedro.org) si necesita fotocopiar, escanear o hacer copias digitales de algún fragmento de esta obra.

Para descargar el contenido multimedia diríjase a la siguiente dirección:

<http://www.icbeditores.com/multimedia-internacional>

UNIDAD 1.1

HTML 5 PARTE 1^a

Introducción a HTML y CSS

- Introducción e Historia de HTML
- Fundamentos de HTML
- Fundamentos de CSS

INTRODUCCIÓN E HISTORIA DE HTML

HTML5 es la versión 5 de HTML, que corresponde a las siglas en inglés de HiperText Markup Language o Lenguaje de Marcas de Hipertexto.

HTML es un lenguaje de programación que utiliza una serie de códigos llamados etiquetas que van definiendo los elementos que componen una página web: texto, imágenes, etc. Esas etiquetas serán interpretadas por un programa navegador de internet (como por ejemplo Internet Explorer) que mostrará adecuadamente la página web al usuario.

Pero HTML5 no se limita a ser un lenguaje de etiquetas HTML que sólo permiten definir elementos básicos, sino que combina nuevas etiquetas de lenguaje HTML, propiedades CSS3, Javascript y algunas otras tecnologías. Todas ellas suponen una actualización de gran potencia al conjunto de herramientas ya existente, y con él se pueden crear páginas web más sofisticadas y útiles.

HISTORIA DE HTML

El origen de HTML se remonta al año 1980, cuando el físico Tim Berners-Lee, trabajador del CERN (Organización Europea para la Investigación Nuclear) propuso un nuevo sistema de "hipertexto" para compartir documentos.

El primer documento formal con la descripción de HTML se publicó en 1991 con el nombre HTML Tags (Etiquetas HTML) y todavía hoy puede ser consultado a modo de curiosidad.

La primera propuesta oficial para convertir HTML en un estándar se realizó en 1993 por parte del organismo IETF (Internet Engineering Task Force). Aunque se consiguieron avances significativos (en esta época se definieron las etiquetas para imágenes, tablas y formularios) ninguna de las dos propuestas de estándar, llamadas HTML y HTML+ consiguieron convertirse en estándar oficial.

En 1995, el organismo IETF organiza un grupo de trabajo de HTML y consigue publicar, el 22 de septiembre de ese mismo año, el estándar HTML 2.0. A pesar de su nombre, HTML 2.0 es el primer estándar oficial

de HTML. Se creó con objetivos divulgativos, orientado a la actividad académica, en el que el contenido de las páginas era más importante que el diseño.

Pero esta versión del HTML carecía de muchas herramientas que permitieran controlar el diseño de las páginas y añadir contenido multimedia, por lo que Netscape (cuyos navegadores eran los más utilizados por aquellos años) comenzó a incluir nuevas etiquetas que no existían en el estándar.

El comité encargado de establecer los estándares dentro de Internet, comenzó a trabajar en el borrador de una nueva versión de HTML, el borrador de HTML 3.0, pero este borrador resultó demasiado extenso, al intentar incluir numerosos nuevos atributos para etiquetas ya existentes, y la creación de otras muchas etiquetas nuevas. Por ello, no fue bien aceptado por el mercado y varias compañías se unieron para formar un nuevo comité encargado de establecer los estándares del HTML. Este comité pasó a llamarse W3C.

En enero de 1997 se aprobó el estándar HTML 3.2. Este nuevo estándar incluía las mejoras proporcionadas por los navegadores Internet Explorer y Netscape Navigator, que ya habían realizado extensiones sobre el estándar HTML 2.0.

En diciembre de 1997 se aprobó el estándar HTML 4.0, creado para estandarizar los marcos (frames), las hojas de estilo y los scripts.

HTML 4.0 es el lenguaje que conforma la base de la gran mayoría de las páginas web que se pueden ver hoy día. Los diseñadores y desarrolladores web han estado utilizando esta especificación combinándola con CSS para la definición de estilos y con JavaScript para añadir interactividad a los contenidos.

Tras la finalización de HTML 4.0, el W3C continuó sus trabajos siguiendo la evolución del mundo web, y comenzó con un lenguaje llamado XHTML. Uno de los objetivos de XHTML era crear un lenguaje que pudiera extenderse y resolver las necesidades de las tecnologías futuras, por ejemplo para los dispositivos móviles.

En 2004, los principales fabricantes de navegadores y un grupo de desarrolladores web formaron un grupo independiente llamado WHATWG (Web Hypertext Application Technology Working Group). Su

objetivo era crear una especificación de lenguaje HTML mejor, orientada a crear un nuevo tipo de aplicaciones web pero manteniendo la compatibilidad con los navegadores existentes.

En 2006, Tim Berners-Lee anunció que el W3C y el WHATWG se habían unido para la elaboración conjunta del nuevo estándar. La nueva especificación HTML se llamó HTML5.



En enero de 2010 el W3C presentó el logo de HTML5, que se puede descargar y utilizar libremente y de forma gratuita. Este logo se puede incorporar a los sitios web o cualquier otro contenido para indicar que se utiliza esta tecnología.

FUNDAMENTOS DE HTML

EDITORES HTML

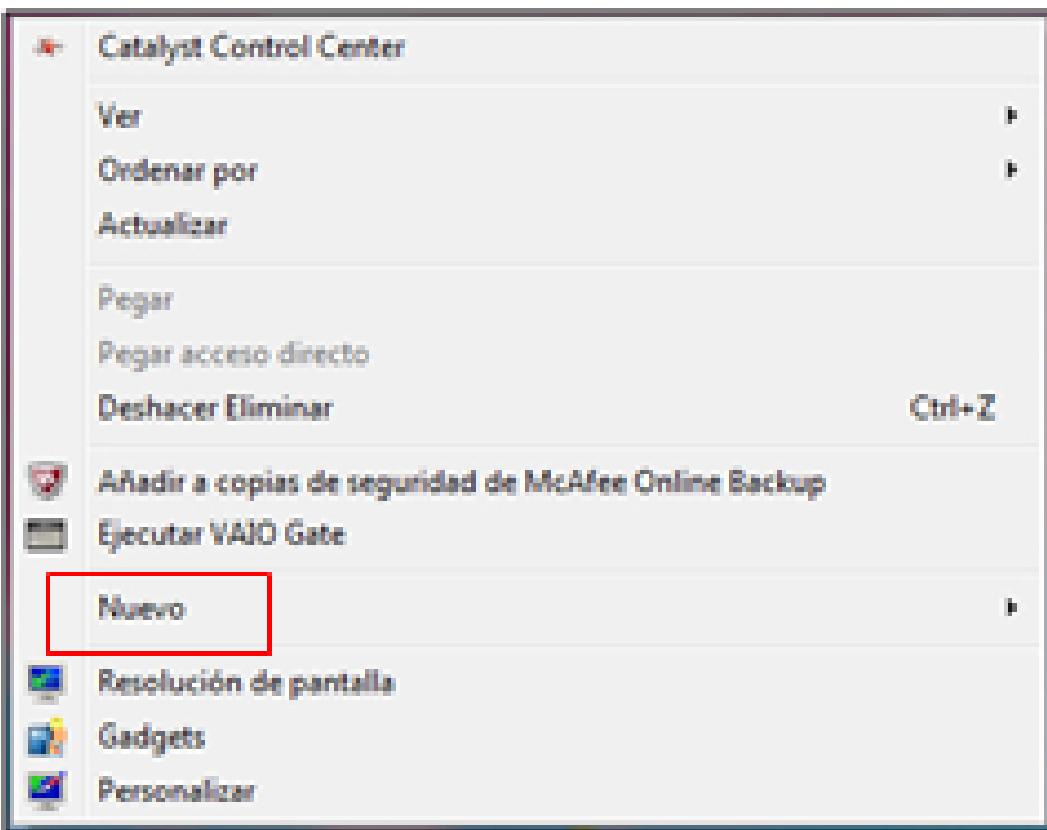
Un documento HTML no es más que un archivo de texto, por tanto para crear o modificar archivos HTML basta con utilizar un editor de texto simple, como el bloc de notas de Windows.

Existen editores más sofisticados como Frontpage o Dreamweaver. Estos editores presentan el código de forma más amigable, distinguiendo por ejemplo los diferentes tipos de etiquetas con colores o incorporando herramientas visuales para incorporar elementos, pero además de que se trata de herramientas de pago, no son recomendables cuando se está aprendiendo, debido a que en numerosas ocasiones introducen líneas de código automáticamente lo cual puede confundir al alumno.

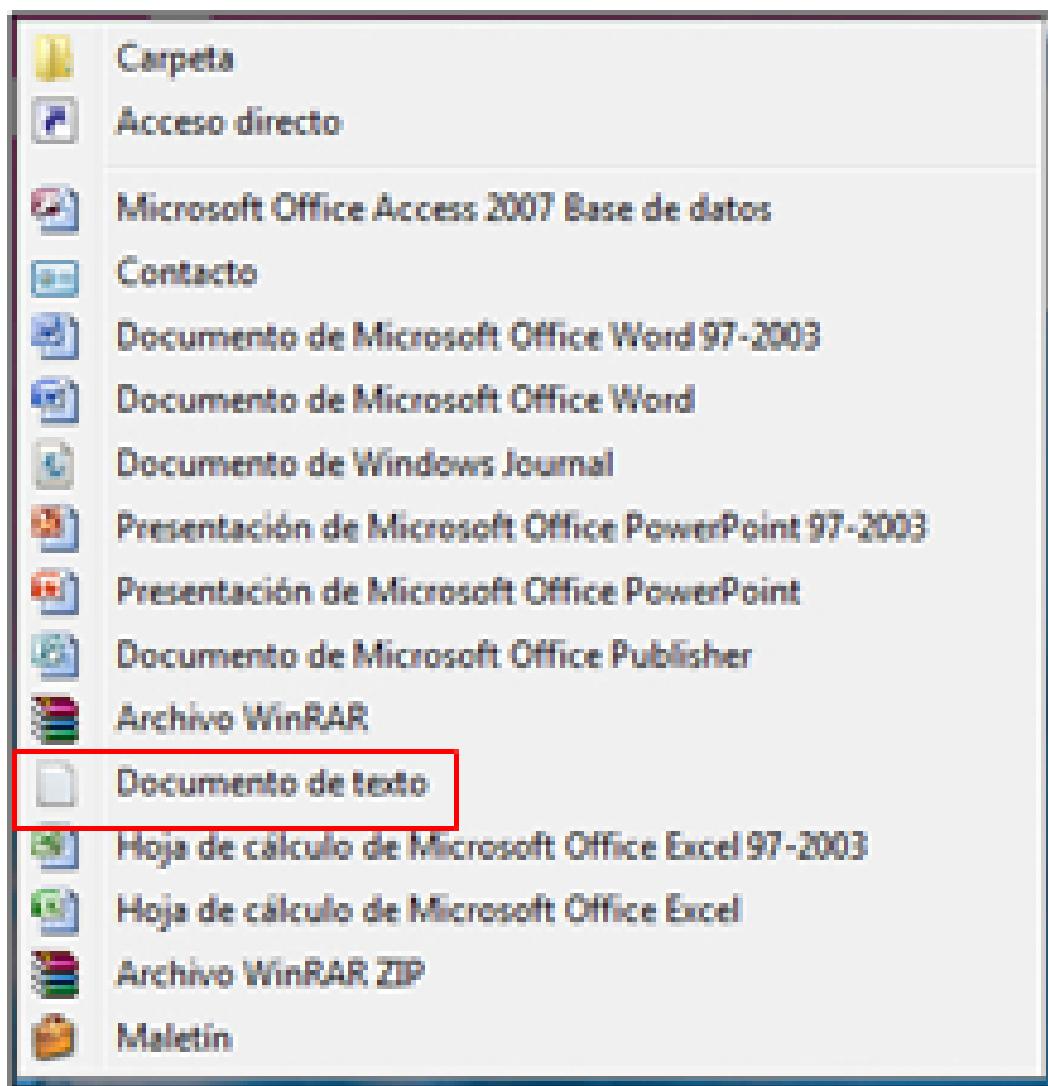
Para crear un nuevo documento HTML en un ordenador con sistema operativo Windows, por ejemplo en el escritorio:

Hacer clic en el botón derecho del ratón (en cualquier punto del escritorio)

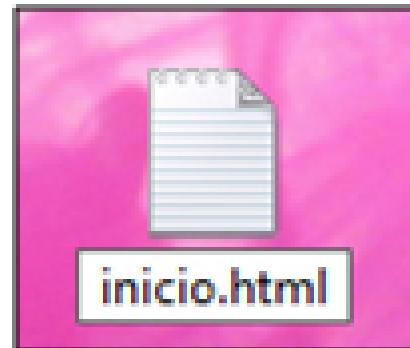
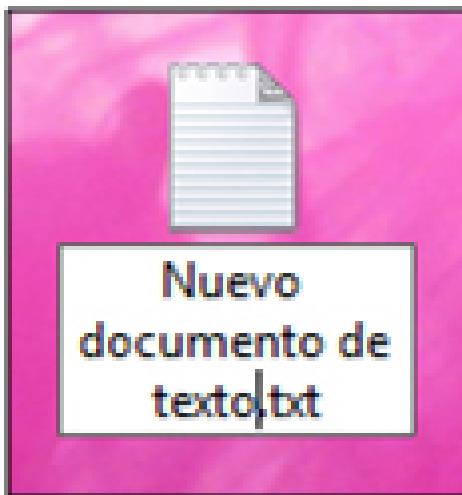
Pulsar Nuevo



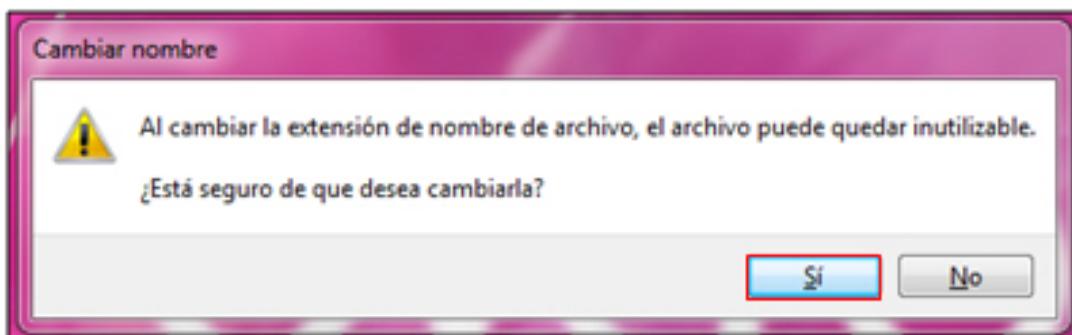
Seleccionar Documento de texto.



Renombrar El archivo de Nuevo documento de texto.txt a inicio.html



Pulsar Sí



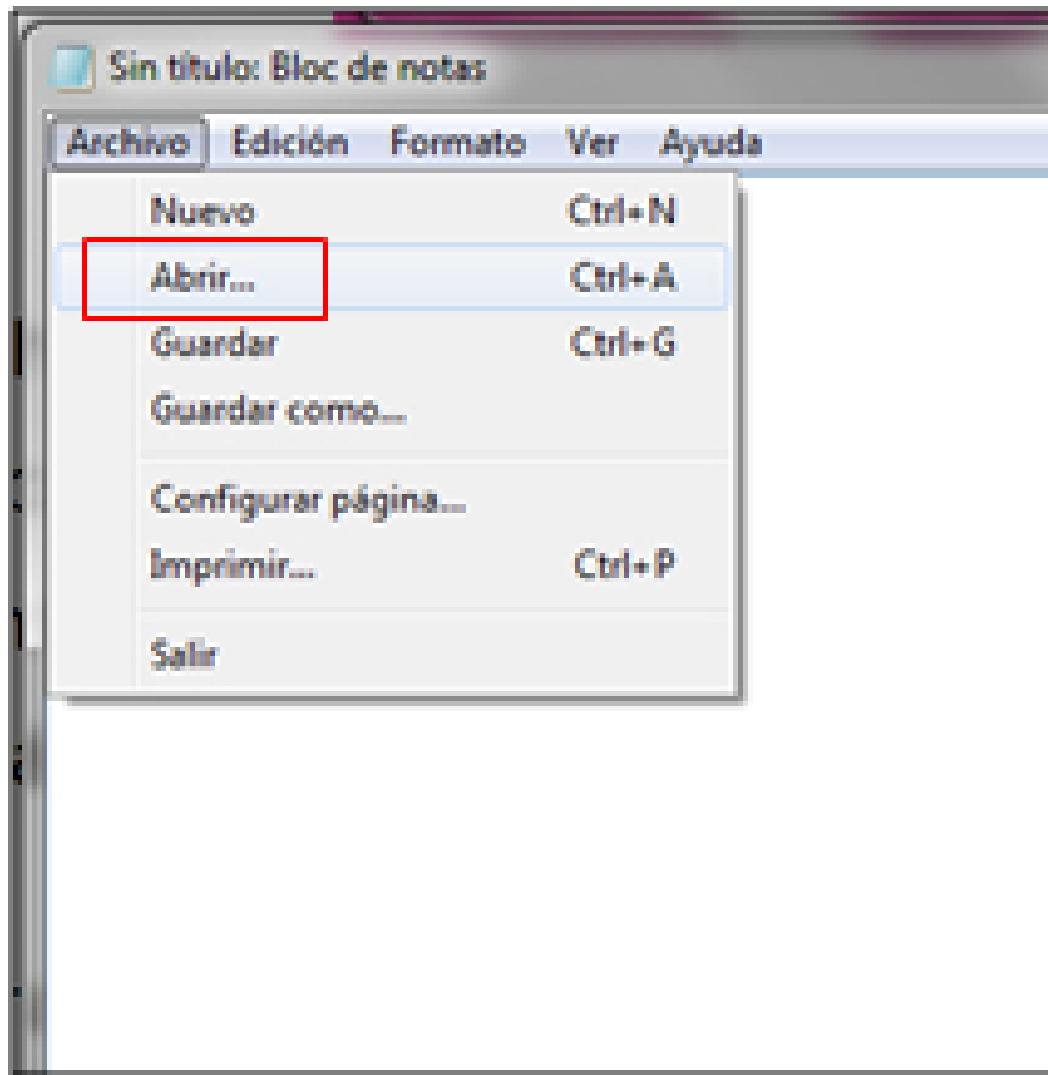
Al hacer doble clic sobre el archivo creado inicio.html, el sistema operativo Windows lo ejecutará, es decir, abrirá el navegador de internet e interpretará el código mostrándolo como una página web (en este caso mostrará una página en blanco ya que el archivo está vacío).

Para editar el archivo e introducir código:

Abrir El bloc de notas de Windows.

Desplegar menú Archivo

Pulsar Abrir



En la parte inferior de la ventana explorador que aparece seleccionar
Todos los archivos (*.*)



Seleccionar El archivo inicio.html.

Pulsar Abrir



Desde aquí ya se puede escribir el código HTML dentro del archivo.

EL CÓDIGO HTML

La especificación completa de la última versión de HTML 4.0 puede consultarse de forma gratuita en la web:

<http://www.w3.org/TR/html401>. Esta norma oficial está escrita de manera muy formal y algunas secciones son difíciles de comprender. Afortunadamente no es necesario leer las recomendaciones oficiales para aprender a diseñar páginas con HTML.

El lenguaje HTML, como ya se ha comentado, es un lenguaje de marcas o etiquetas. Esto quiere decir que los elementos que conforman la página web se describen mediante palabras especiales que marcan el inicio y el final de los mismos, estas palabras se denominan etiquetas.

Aunque existen algunas excepciones, en general las etiquetas se indican por pares de la siguiente forma:

- Etiqueta de apertura: carácter <, seguido del nombre de la etiqueta (sin espacios en blanco) y terminado con el carácter >.
- Etiqueta de cierre: carácter <, seguido del carácter /, seguido del nombre de la etiqueta (sin espacios en blanco) y terminado con el carácter >.

Por tanto, la estructura típica de las etiquetas HTML es:

<nombre_etiqueta>... </nombre_etiqueta>

Estructura de un Documento HTML

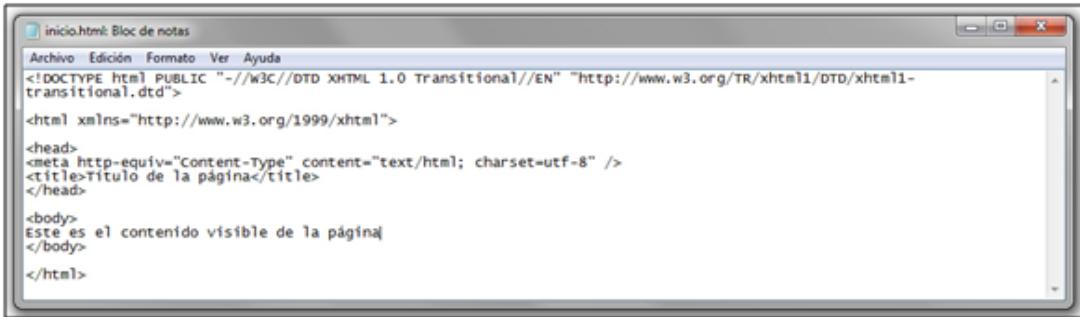
Todas las páginas HTML contienen los mismos elementos básicos:

- Doctype: Es la primera línea de código que tiene que estar en cualquier documento HTML. Esta línea indica al navegador qué especificación de HTML se está utilizando. Suele ser la siguiente:



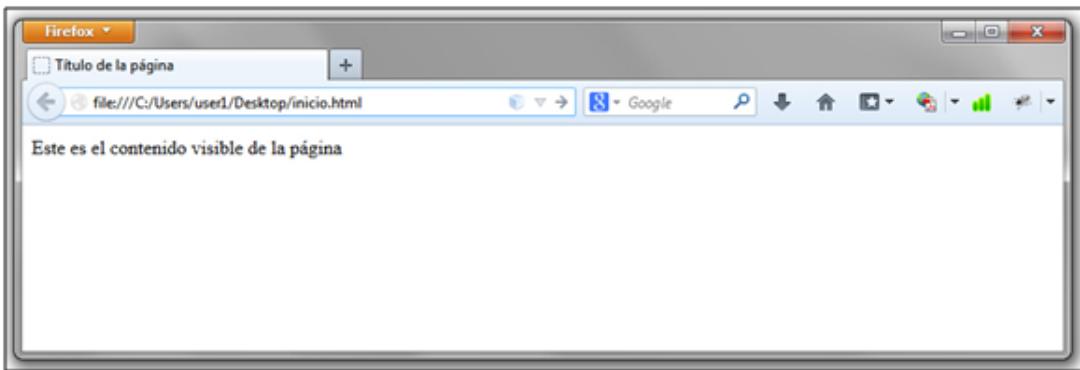
- HTML: El par de etiquetas `<html>` y `</html>` se encuentran al principio y al final de todo documento HTML y sirve para indicar a todos las aplicaciones que pueden analizar texto sin formato (no sólamente a los navegadores) que esa página utiliza HTML.
- Head: Las etiquetas `<head>` y `</head>` delimitan el contenido de la cabecera del documento, es decir, el título de la página y una información que no aparece en la pantalla.
- Title: El par de etiquetas `<title>` y `</title>` rodea el texto del título. El título aparece en la barra de título del navegador web cuando presenta la página. Normalmente va dentro del elemento `<head>`.
- Meta: `<meta>` permite aportar metainformación al documento, para su mejor identificación e indexación por los motores de búsqueda.
- Body: Las etiquetas `<body>` y `</body>` rodean el contenido visible de la página. También puede llevar incluida información sobre las propiedades de la página, por ejemplo `<body bgcolor="#RRGGBB">` define el color de fondo de la página.

Por ejemplo, el archivo inicio.html queda:



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Título de la página</title>
</head>
<body>
Este es el contenido visible de la página
</body>
</html>
```

Al ejecutarlo, haciendo doble clic sobre el archivo, el navegador de internet (en este caso Firefox) muestra:



Etiquetas HTML más Comunes

- Comentarios: <!-- comentarios -->: Para escribir anotaciones que sirven para ayudar a la comprensión del código. Lo que se escribe dentro de esta etiqueta es ignorado por el navegador y no se muestra en la página.

Formato de Textos

- ... : Aplica negrita al texto incluido entre las etiquetas (es equivalente usar ...)
- <i>...</i>: Aplica cursiva al texto incluido entre las etiquetas (es equivalente usar ...)
- <u> </u>: Aplica subrayado
- ...: Delimita un texto con un formato de fuente determinado definido por sus propiedades:

- ... : Indica el tamaño del texto.
- ... : Define el color del texto, donde cada letra RRGGBB es un valor hexadecimal (de 0 a F) que indica el color.
- ... : Determina el tipo de fuente de texto, es decir, la tipografía.

La etiqueta puede incluir los tres parámetros (tamaño, fuente y color):

- h1, h2, h3, h4, h5, h6: Indican 6 niveles de formato de encabezados, en los que <h1>...</h1> delimitaría el tipo de fuente de mayor tamaño.

Formato de Párrafos

-
: Introduce un salto de línea.
- <p>... </p>: Delimita un párrafo de texto.
 - <p align='center'>: Texto del párrafo con alineación centrada.
 - <p align='left'>: Párrafo alineado a la izquierda.
 - <p align='right'>: Párrafo alineado a la derecha.
 - <p align='justify'>: Texto del párrafo con alineación justificada.
- <hr>: Inserta una línea horizontal.
- <div>: Se utiliza para delimitar una sección dentro del documento, de forma que agrupe un número de elementos para luego añadirle un estilo determinado o realizar operaciones sobre ese bloque específico.

Creación de Listas

- Lista no numerada:

```
<ul> Inicio de la lista  
<li>primer elemento de la lista</li>  
<li>segundo elemento de la lista</li>  
<li>tercer elemento de la lista</li>  
</ul> Cierra la lista
```

- Lista numerada:

```
<ol> Inicio de la lista  
<li>primer elemento de la lista</li>  
<li>segundo elemento de la lista</li>  
</ol> cierra lista.
```

Imágenes

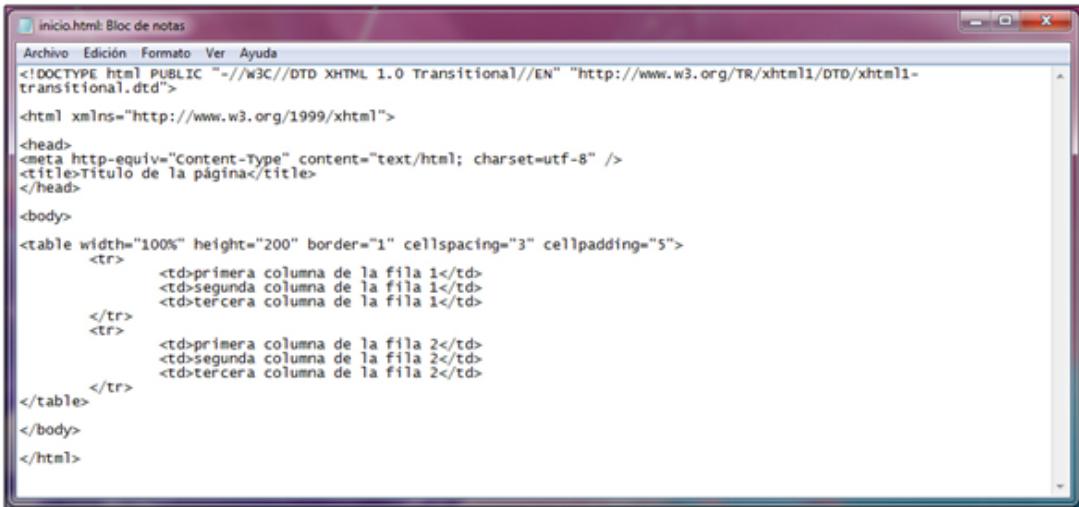
- : Inserta una imagen que se encuentra en la ruta indicada por "dirección de la imagen". También se pueden incluir una serie de propiedades:
 - : Establece un borde de X pixels en torno a la imagen.
 - : Establece un tamaño de la imagen, donde XX e YY son al altura y anchura en pixels respectivamente.
 - : Se muestra un texto al pasar el ratón sobre la imagen.
 - : Alineación inferior de la imagen respecto al texto.
 - : Alineación de la imagen en medio del texto.

- : Alineación superior de la imagen respecto al texto.
- : Alineación izquierda de la imagen en el párrafo.
- : Alineación derecha de la imagen en el párrafo.
- : Espacio horizontal en píxeles entre la imagen y el texto.
- : Espacio vertical en píxeles entre la imagen y el texto.

Tablas

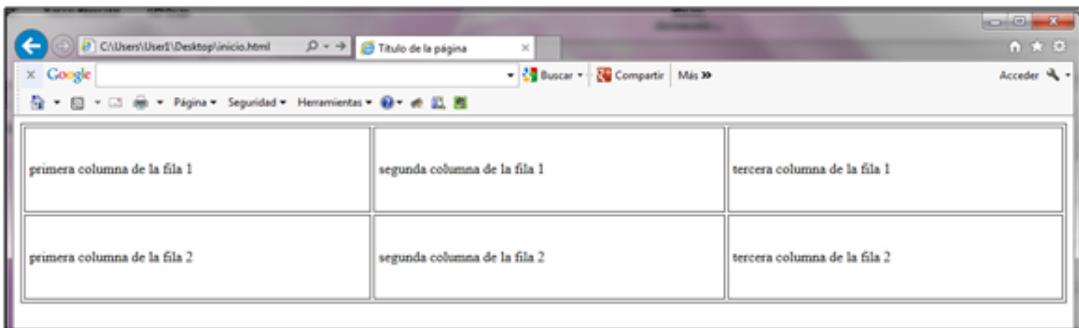
- <table>...</table>: Define dónde comienza y termina la tabla.
 - <table width="X">: Determina la anchura de la tabla. Puede darse en píxeles o en porcentaje.
 - <table height="X">: Determina la altura de la tabla en píxeles.
 - <table border="X">: Establece el grosor en píxeles del borde de la tabla
 - <table cellspacing="X">: Define el espacio en píxeles entre las celdas.
 - <table cellpadding="X">: Define el espacio en píxeles entre el borde y el texto.
- <tr>...</tr>: Indica el comienzo y el fin de cada una de las filas de la tabla.
- <td>...</td>: Indica el comienzo y el fin de cada una de las columnas o celdas dentro de las filas.

Ejemplo de tabla de 2 filas y 3 columnas:



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Titulo de la página</title>
</head>
<body>
<table width="100%" height="200" border="1" cellspacing="3" cellpadding="5">
<tr>
<td>primera columna de la fila 1</td>
<td>segunda columna de la fila 1</td>
<td>tercera columna de la fila 1</td>
</tr>
<tr>
<td>primera columna de la fila 2</td>
<td>segunda columna de la fila 2</td>
<td>tercera columna de la fila 2</td>
</tr>
</table>
</body>
</html>
```

El resultado en el navegador sería:



Dentro de cada celda se puede alinear el texto o cualquier contenido, cambiar el color de fondo, con las etiquetas habituales para texto, párrafos o imágenes.

Creación de Enlaces

- [Nombre del enlace](http://www.ejemplo.com): Cuando se pulsa sobre el texto "Nombre del enlace" en la web, se va al vínculo indicado en la dirección de href.
- [Nombre del enlace](mailto:ejemplo@ejemplo.com): Cuando se pulsa sobre el texto "Nombre del enlace" en la web, se abre una ventana para enviar un correo electrónico a la dirección indicada.

Página con Marcos (Frames)

Este tipo de páginas no llevan etiquetas <body>, sino que se utilizan los marcos o frames. En primer lugar se define un frameset o contenedor y dentro se incluyen los distintos frames, que subdividen la página en marcos.

Cada frame corresponderá a un archivo html que deberá ser programado aparte con el contenido correspondiente.

Por ejemplo:

<frameset cols="20%, 80%"> ... </frameset>: Divide la página en dos marcos en forma de columnas, cada una con su anchura correspondiente en porcentaje.

<frame src="menu.htm" name="navegacion">: Indica que el archivo menu.htm corresponde al marco de la izquierda, llamado “navegación” con 20% de anchura.

<frame src="principal.htm" name="contenidos">: Indica que el archivo principal.htm corresponde al marco de la derecha, llamado “contenidos”, 80% de anchura

</frameset>

- <frameset rows=" , ">: Divide la página en marcos horizontales.
- frameborder="NO": Evita que se vea el borde entre los marcos.
- framespacing="X": Establece X pixels de separación entre los marcos.
- scrolling="NO": Evita que aparezca una barra de scroll dentro del marco.
- scrolling="auto": Mostrará la barra de scroll sólo si es necesario.

Ejemplo de una página con tres marcos en forma de filas. La superior y la inferior tienen un tamaño fijo de 80 pixels; la del medio es adaptable. No se muestran los bordes entre los marcos. Los archivos cabecera.htm, principal.htm y piedepagina.htm se programarían aparte con el contenido de cada zona.



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Título de la página</title>
</head>
<frameset rows="80,*,80" frameborder="0" border="0" framespacing="0">
<frame src="cabecera.htm" name="topFrame" scrolling="No">
<frame src="principal.htm" name="mainFrame">
<frame src="piedepagina.htm" name="bottomFrame" scrolling="No">
</frameset>
</html>
```

Formularios

Un formulario es una sección dentro de un documento HTML que sirve para que los usuarios introduzcan una serie de datos y los envíen a un servidor web o a un servidor de correo. Las etiquetas que se usan principalmente en los formularios HTML son:

- <form>...</form>: Indican el comienzo y el fin del formulario.
- <label>...</label>: Delimitan una etiqueta de texto dentro del formulario.
- <input>...</input>: Indican el principio y final de algún control de entrada de datos por parte del usuario. Pueden ser de varios tipos, según se defina su atributo type:
 - text: Entrada de texto.
 - password: Igual que text pero el texto introducido se presentan de tal modo que se oculten los caracteres, normalmente mediante un serie de asteriscos. Este tipo de control suele utilizarse para introducir contraseñas.
 - checkbox: Casilla de verificación o checkbox.
 - radio: Elemento de selección entre varias opciones de tipo radio.
 - submit: Botón para enviar el formulario.

- image: Botón para enviar el formulario pero con una imagen de fondo.
 - reset: Botón para borrar todo el contenido de datos del formulario.
 - hidden: campo oculto, envía una información que no se muestra en el formulario.
 - file: Permite adjuntar un archivo desde el ordenador.
- <select>...</select>: Delimita el principio y el fin de una lista desplegable de opciones. Cada opción se indica mediante la etiqueta <option>.
 - <textarea>...</textarea>: Permite introducir un texto largo.

Ejemplo de formulario HTML

```

<form action="mailto:ejemplo@ejemplo.com" method="post" enctype="text/plain">

    Nombre: <input type="text" name="nombre" size="25" maxlength="50"><br><br>
    Apellidos: <input type="text" name="apellidos" size="35" maxlength="100"><br><br>
    Email: <input type="text" value="" name="correo" size="40" maxlength="100"><br><br>
    Localidad: <input type="text" name="poblacion" size="15" maxlength="50"><br><br>

    <table width="50%" border="0" cellspacing="0" cellpadding="10">
        <tr><td>Edad: <br>
            <input type="radio" name="edad" value="020"> 0-20 <br>
            <input type="radio" name="edad" value="2040" checked> 20-40 <br>
            <input type="radio" name="edad" value="4060"> 40-60 <br>
            <input type="radio" name="edad" value="60100"> 60-100 </td>
        </tr>
        <td>cual es tu animal favorito?<br>
            <input type="checkbox" name="animal"> Perro<br>
            <input type="checkbox" name="animal"> Gato<br>
            <input type="checkbox" name="animal"> Caballo<br>
            <input type="checkbox" name="animal"> Otros</td>
        </tr>
    </table>

    Escribe tu opinion sobre nuestra pagina web<br>
    <textarea cols="40" rows="5" name="opinion">Escribe tu opinion...</textarea> <br><br>
    Tienes alguna sugerencia?: <br>
    <textarea cols="40" rows="5" name="sugerencias">Escribe aqui tus sugerencias...</textarea> <br><br>
    Cuanto navegas por internet? <br>
    <select name="frecuencia" size="2">
        <option value="1">2 horas al dia.
        <option value="2">4 horas al dia.
        <option value="3">10 horas a la semana.
        <option value="4">20 horas al mes.
    </select> <br> <br>

    <input type="submit" value="Enviar formulario">

    <input type="Reset" value="Borrar formulario">

</form>

```

El resultado en el navegador sería:

Nombre:

Apellidos:

Email: @

Localidad:

Edad:

- 0-20
- 20-40
- 40-60
- 60-100

Cual es tu animal favorito?

- Perro
- Gato
- Caballo
- Oros

Escribe tu opinion sobre nuestra pagina web

Escribe tu opinion...

Tienes alguna sugerencia..

Escribe aqui tus sugerencias...

Cuanto navegas por internet?

- 2 horas al dia.
- 4 horas al dia.

FUNDAMENTOS DE CSS

CSS (Cascading Style Sheets u Hojas de Estilo en Cascada) es un lenguaje de programación muy parecido a HTML que permite aplicar estilos a los distintos elementos de las páginas web, de modo que los títulos, listas y párrafos pueden verse igual en todas y cada una de las páginas.

SINTAXIS BÁSICA DE CSS

Cada estilo CSS se compone de tres partes: un selector, una propiedad y un valor, y su sintaxis es:

```
selector {propiedad:valor;}
```

- Selector: Indica a qué etiqueta HTML se aplicará el estilo.
- Propiedad: Indica a qué propiedad de la etiqueta se aplicará el estilo, por ejemplo al color de fondo, al tamaño de la letra, etc.
- Valor: Valor de la propiedad.

Por ejemplo, el estilo:

```
body {background-color: #FF0000;}
```

aplicará el color rojo al fondo de la página.

Se pueden definir propiedades múltiples, separándolas por punto y coma, por ejemplo:

```
h2{text-align:justify; color:blue;}
```

Aplicará alineación justificada y color azul a los textos de encabezado de tipo h2.

AGRUPACIÓN DE ELEMENTOS CON EL ATRIBUTO CLASS

Se pueden definir estilos diferentes para un mismo tipo de etiqueta HTML, creando clases mediante el atributo class y luego indicando en CSS las propiedades de cada clase particular.

Por ejemplo, en una página hay dos listas de enlaces, una de productos de ropa y otra de complementos:

```
<p>Ropa:</p>
<ul>
<li><a href="camisetas.htm">Camisetas</a></li>
<li><a href="pantalones.htm">Pantalones</a></li>
<li><a href="camisas.htm">Camisas</a></li>
</ul>

<p>Complementos:</p>
<ul>
<li><a href="sombreros.htm">Sombreros</a></li>
<li><a href="collares.htm">Collares</a></li>
<li><a href="pendientes.htm">Pendientes</a></li>
</ul>
```

Se desea que los enlaces de la lista de ropa aparezcan en verde y los de complementos en rojo. El resto de enlaces de la página deberán aparecer en azul. Se utiliza el atributo class para definir dos clases: ropa y complementos, como una propiedad dentro de la etiqueta <a> que define los enlaces:

```
<p>Ropa:</p>
<ul>
<li><a href="camisetas.htm" class="ropa">Camisetas</a></li>
<li><a href="pantalones.htm" class="ropa">Pantalones</a></li>
<li><a href="camisas.htm" class="ropa">Camisas</a></li>
</ul>

<p>Complementos:</p>
<ul>
<li><a href="sombreros.htm" class="complementos">Sombreros</a></li>
<li><a href="collares.htm" class="complementos">Collares</a></li>
<li><a href="pendientes.htm" class="complementos">Pendientes</a></li>
</ul>
```

Para asignar las propiedades a cada clase, se utiliza la sintaxis css:

```
a{color: blue;}
```

```
a.ropa{color: green;}
```

```
a.complementos{color: red;}
```

IDENTIFICACIÓN DE UN ELEMENTO USANDO EL ATRIBUTO ID

Dentro de cada etiqueta HTML se puede incluir un identificador mediante el atributo id. No pueden existir dos elementos dentro del mismo documento con el mismo id.

Este id puede servir para aplicar un estilo especial solamente a un elemento. En CSS se indica que un estilo se asigna a un identificador concreto mediante el carácter almohadilla #.

Por ejemplo, se tiene la siguiente disposición de líneas de texto de tipo h1 y h2, con los nombres de los capítulos de un libro. Cada línea lleva un id diferente.

```
<h1 id="c1">Capítulo 1</h1>
  ...
    <h2 id="c1-1">Capítulo 1.1</h2>
    ...
      <h2 id="c1-2">Capítulo 1.2</h2>
<h1 id="c2">Capítulo 2</h1>
  ...
    <h2 id="c2-1">Capítulo 2.1</h2>
    ...
      <h2 id="c2-2">Capítulo 2.2</h2>
<h1 id="c3">Capítulo 3</h1>
  ...
    <h2 id="c3-1">Capítulo 3.1</h2>
    ...
      <h2 id="c3-2">Capítulo 3.2</h2>
```

Si se quisiera aplicar un formato especial, por ejemplo color rojo solamente al capítulo 3-1, se escribiría la sintaxis CSS:

```
#c3-1 {color: red;}
```

CÓMO SE APLICA CSS A UN DOCUMENTO HTML

Se puede hacer de tres formas diferentes, pero la más recomendada es la tercera.

1. Mediante el atributo style:

El atributo style es de CSS pero se inserta dentro de las etiquetas HTML directamente. Por ejemplo, para aplicar el color rojo al fondo de la página, en el documento HTML se escribiría, dentro de la etiqueta body:

```
<html>
  <head>
    <title>Ejemplo</title>
  </head>
  <body style="background-color: #FF0000;">
    <p>Esta es una página con fondo rojo</p>
  </body>
</html>
```

2. Mediante la etiqueta style:

También de esta forma se insertaría código CSS dentro del archivo HTML, en esta ocasión mediante una etiqueta HTML llamada style. Por ejemplo, para conseguir el mismo efecto que en el caso anterior, se escribiría:

```
<html>
  <head>
    <title>Ejemplo</title>
    <style type="text/css">
      body {background-color: #FF0000; }
    </style>
  </head>
  <body>
    <p>Esta es una página con fondo rojo</p>
  </body>
</html>
```

3. Mediante un archivo CSS externo.

Esta es la forma más recomendada de definir los estilos, ya que puede ahorrar mucho trabajo. Si se desea cambiar, por ejemplo, el color de fondo de un sitio web compuesto por 100 páginas, se puede llevar a cabo en unos segundos modificando parte del código de la hoja de estilo principal. Con una hoja de estilos externa se evita el tener que cambiar de forma manual los 100 documentos HTML.

Una hoja de estilos CSS no es más que un archivo de texto, igual que un archivo HTML, solo que con la extensión .css. Por tanto la forma de crearlas y editarlas es exactamente igual que con los archivos HTML, y se puede usar también el bloc de notas.

En ese archivo se escriben todas las reglas de estilos en lenguaje CSS como se ha descrito anteriormente, guardando el archivo con el nombre por ejemplo estilos.css.

Después en cada archivo HTML al que se le quiera aplicar la hoja de estilos, se inserta una línea de código dentro del encabezado, indicando la ruta del archivo CSS, de la siguiente forma:

```
<html>
  <head>
    <title>Mi documento</title>
    <link rel="stylesheet" type="text/css" href="estilos.css" />
  </head>
<body>
  ...

```

LO QUE HEMOS APRENDIDO

- HTML5 (HyperText Markup Language) es un lenguaje de marcas para programar páginas web que combina etiquetas HTML, propiedades CSS, Javascript y otras tecnologías, permitiendo crear webs útiles y sofisticadas.
- Un documento HTML no es más que un archivo de texto sencillo, que puede editarse por ejemplo con el bloc de notas de Windows. Su extensión debe ser .html o .htm.
- El código HTML se basa en la definición de los elementos de la web a través de etiquetas, que normalmente se indican por pares de la forma <etiqueta> contenido </etiqueta>.
- CSS (Cascade Style Sheets) es un lenguaje que permite aplicar estilos a los elementos de páginas web creadas en HTML.
- Cada estilo CSS se define mediante un selector, una propiedad y un valor, mediante una sintaxis determinada.
- La forma más recomendable de aplicar estilos CSS a una página HTML es mediante un archivo CSS externo, que es un archivo de texto sencillo con la extensión .css.

UNIDAD 1.2

HTML 5 PARTE 1^a

Introducción a HTML5 y CSS3

- Definición de HTML5
- Etiquetas de HTML5
- APIs para HTML5
- Novedades en CSS3

DEFINICIÓN DE HTML5

Como ya se ha comentado, HTML5 es una combinación de nuevas etiquetas de HTML, propiedades CSS3, JavaScript y algunas tecnologías complementarias de apoyo, pero que técnicamente son independientes de la propia especificación HTML5. Por ello se distingue entre la especificación HTML5 en sí y la familia HTML5:

- La especificación HTML5: son nuevos elementos de sintaxis, utilizados para crear páginas web junto con las etiquetas utilizadas anteriormente. Estas nuevas etiquetas suponen herramientas más avanzadas y se traducen en mejores resultados para el usuario.
- La familia HTML5: incluye éstas nuevas etiquetas y además tecnologías como CSS3, Geolocalización, Almacenamiento Web, Drag and Drop, etc.

Los nuevos navegadores web ya incorporan herramientas para responder a las expectativas de los consumidores lo que es consecuencia de la evolución natural de la tecnología. Con los lenguajes de programación anteriores, existían muchas limitaciones y no todas las funcionalidades se podían incorporar de manera sencilla. HTML5 aporta nuevas funciones con el fin de conseguir que los sitios web sean más interesantes, atractivos y útiles.

Las versiones anteriores de HTML no tenían capacidad para reproducir contenidos multimedia, como audio o vídeo, sin un complemento o plugin como por ejemplo Flash. Tampoco tenían capacidad para almacenar datos en el ordenador del usuario, esto se hacía mediante otras tecnologías. Además en HTML 4 no existe ningún formato de dibujo, sino que los gráficos y animaciones se tenían que mostrar a través de archivos de imagen.

ETIQUETAS DE HTML5

A continuación se describirán algunas de las nuevas etiquetas que incorpora HTML5 además de las que ya tenían las anteriores versiones de HTML, que por supuesto se pueden seguir utilizando.

ETIQUETAS SEMÁNTICAS

HTML5 incorpora nuevas etiquetas pensadas para hacer que la estructura de la página web sea más lógica y funcional. Antes de HTML5, la estructura de una página se basaba fundamentalmente en etiquetas `<div>`, generalmente asociadas a una clase o a un id.

Por ejemplo, en HTML 4.0 para definir la cabecera de una página web se suele utilizar algo como:

```
<div id="header" > Nombre de la cabecera </div>
```

De esta forma, para definir por ejemplo la anchura, altura y color de fondo de la cabecera en el código CSS se utilizaría por ejemplo:

```
#header {width:960px; height:100px; background-color:blue;}
```

En la especificación HTML5 existe ya una etiqueta llamada `<header>` que viene a sustituir al elemento `<div>` de forma que la sintaxis es mucho más lógica y coherente, se trata de un tipo de etiqueta llamada semántica porque da una noción del tipo de contenido que englobará:

```
<header> Nombre de la cabecera </header>
```

En este ejemplo ya se pueden añadir directamente las propiedades de estilo (ancho, alto, color de fondo, etc.) en una regla para el nuevo elemento header de CSS:

```
header {width:960px; height:100px; background-color:blue;}
```

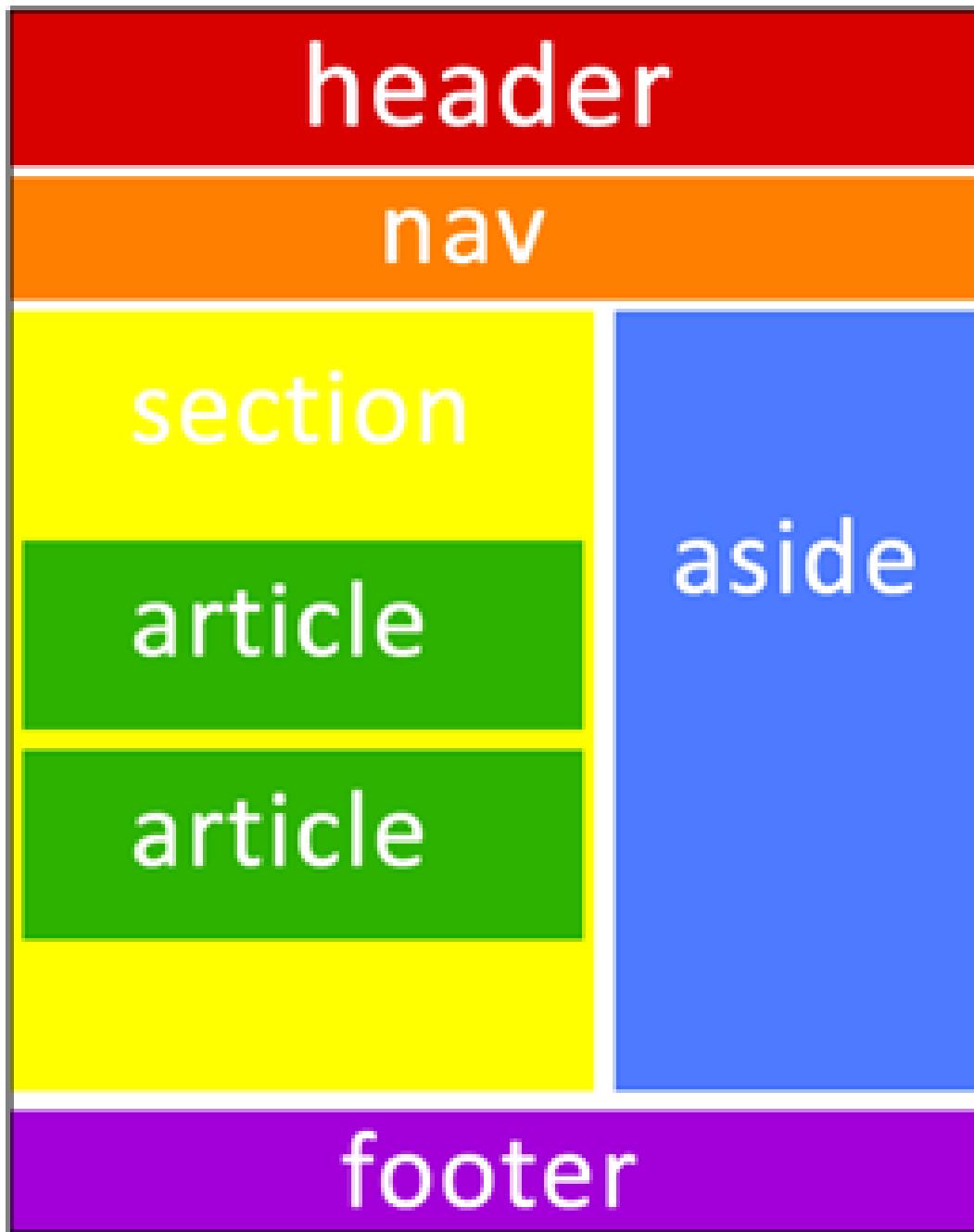
En definitiva, en HTML5 se incorporan las etiquetas semánticas:

- `<header>...</header>`: Para definir la cabecera de la página.
- `<footer>...</footer>`: Para definir el pie de página. Normalmente contiene datos sobre quien ha creado la página, datos del

copyright, etc.

- <nav>...</nav>: Permite definir un elemento de navegación de la página web, como el menú principal o menús secundarios.
- <section>...</section>: Se utilizan para encerrar el código correspondiente a una sección genérica dentro de un documento.
- <aside>...</aside>: Sirve para delimitar todo aquel contenido que no es directamente contenido principal de esa página en concreto. Puede usarse por tanto para todos aquellos elementos secundarios como bloques publicitarios, enlaces externos, citas, un calendario, etc.
- <article>...</article>: Se utiliza para definir artículos como noticias, entradas de un blog, etc. Es probablemente uno de los más importantes de HTML5, ya que permite indicar cuál es el contenido de una página web.

El siguiente ejemplo podría ser una visualización esquemática de la estructura de una página web utilizando las nuevas etiquetas semánticas de HTML5:



El objetivo de los nuevos elementos HTML5 es evitar el exceso de etiquetas `<div>` sustituyéndolas por una estructura de página más consistente y legible. HTML5 no sustituye ningún elemento de sintaxis de HTML, sino que simplemente añade nuevos elementos de vocabulario a la lista existente. Dicho de otra forma, se puede seguir utilizando la etiqueta `<div>`, pero esta etiqueta ya no necesariamente

tiene que ser la base que soporte el diseño visual de toda una página web.

Doctype

El Doctype, es como la primera línea de código que tiene que estar en cualquier documento HTML y que indica al navegador qué especificación de HTML se está utilizando. En HTML5 el Doctype, se simplifica enormemente, ya que basta con escribir:

```
<!DOCTYPE HTML>
```

ETIQUETAS DE CONTENIDO MULTIMEDIA

La especificación HTML5 incluye etiquetas que permiten integrar contenidos multimedia sin necesidad de complementos de navegador. Las más importantes son:

Insertar Audio y Video

- <video>...</video>: Sirven para integrar vídeo en las páginas web de la misma forma que se hace con los archivos de imagen utilizando la etiqueta . Por ejemplo: <video src="mivideo.mp4"></video>.
- <audio>...</audio>: Inserta un archivo de audio mp3 en la página de forma idéntica a como se hace con el vídeo.

En el caso del audio y el vídeo pueden añadirse también los atributos:

- controls: Si se incluye dentro de las etiquetas <audio> o <video> se activa un conjunto de controles de reproducción integrados. Normalmente incluye reproducción, pausa, buscar y ajustar volumen.
- height: Establece el alto del reproductor en píxeles.
- width: Establece el ancho del reproductor en píxeles.
- preload: Define cuánto almacenamiento en búfer es necesario para comenzar la reproducción.

- **autoplay**: Inicia la reproducción automáticamente cuando el reproductor tiene suficiente contenido almacenado en el búfer.
- **poster**: Indica una imagen que se mostrará cuando el reproductor no está disponible o bien mientras se está cargando el contenido.
- **loop**: Reproduce el contenido repetidamente hasta que se presiona el botón de pausa en los controles.

El ejemplo siguiente muestra la imagen poster.jpg hasta que se carga el contenido y reproduce un vídeo repetidamente. También se incluyen los controles de reproducción:

```
<video src="demo.mp4" controls autoplay loop preload="auto"
poster="poster.jpg"></video>
```

Insertar Gráficos

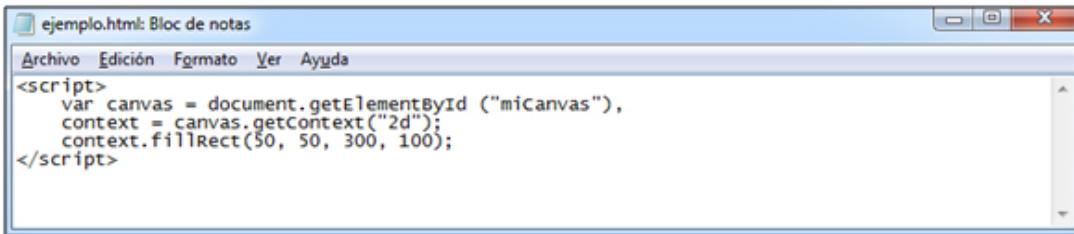
La etiqueta `<canvas>` dota al lenguaje HTML5 de un formato nativo para el dibujo y la animación. Esta etiqueta puede servir, además, como plataforma alternativa para los gráficos y animaciones que se pueden ver en películas con tecnología Flash.

El elemento `<canvas>` funciona a modo de superficie de dibujo dentro de una página web. Dentro de esta superficie de dibujo se pueden crear formas con colores, patrones de relleno, etc. Se pueden manipular los píxeles de forma interactiva en pantalla, mostrar textos y exportar los contenidos a archivos de imagen como PNG. Mediante JavaScript o las nuevas funciones de animación de CSS3 los objetos creados podrán moverse, desaparecer, cambiar de tamaño, etc.

Para incorporar un elemento canvas a una página se utiliza:

```
<canvas id="miCanvas"></canvas>
```

Aparte se escribe el código JavaScript necesario para dibujar. Por ejemplo, para crear un rectángulo en la posición x=50, y=50, ancho=300, alto=100 el código necesario debería ser:



```
<script>
    var canvas = document.getElementById ("miCanvas"),
        context = canvas.getContext("2d");
    context.fillRect(50, 50, 300, 100);
</script>
```

FORMULARIOS

HTML5 incorpora nuevas etiquetas, atributos y tipos de entradas que facilitan el trabajo con formularios.

Nuevos Tipos de Controles de Entradas

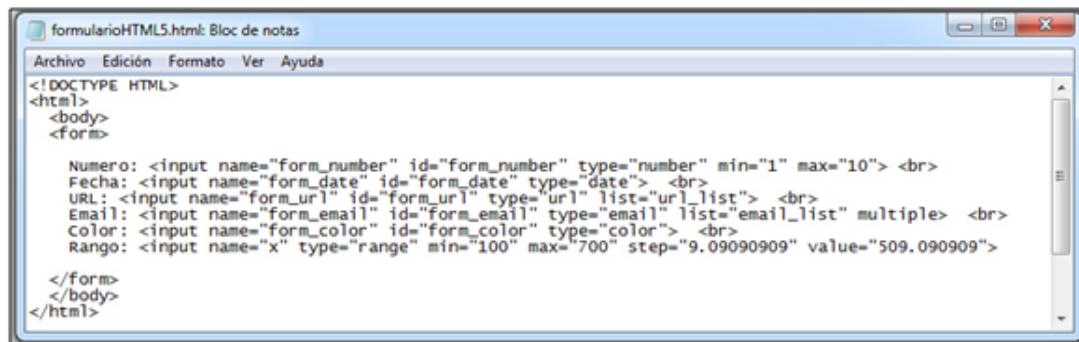
- <input type="email" />: Para introducir una dirección de correo electrónico. El sistema validará que la sintaxis es correcta.
- <input type="url" />: Para introducir una dirección web o url. Muestra un mensaje de error por ejemplo si se introducen caracteres no válidos.
- <input type="date" />: Permite introducir una fecha, incorpora un calendario desplegable para elegirla.
- <input type="number" />: Para introducir un número, se puede indicar un rango máximo y mínimo.
- <input type="range" />: Presenta una barra con un rango para arrastrar con el ratón.
- <input type="search" />: Para introducir un término de búsqueda en la web.
- <input type="color" />: Para elegir un color. Despliega una carta de colores.

Nuevos Atributos para las Etiquetas

Entre los nuevos atributos para las etiquetas de tipo <input> se encuentran:

- autocomplete: La función de autocompletar el texto cuando se rellena un formulario es una característica del navegador. Sin embargo, HTML5 ofrece la posibilidad de activar o desactivar esta característica, dando el valor "on" u "off" a este atributo.
- autofocus: Indica si se colocará el cursor del ratón por defecto en el formulario al entrar en la página.
- min: Permite indicar un valor mínimo para un rango, por ejemplo en el input de tipo number.
- max: Permite indicar un valor máximo para un rango, por ejemplo en el input de tipo number.
- placeholder: Permite definir un texto que se desea que aparezca dentro del campo del formulario a modo de ayuda. Este texto se resalta en un tono diferente, de forma que, cuando se pone el cursor sobre el campo, el texto desaparece y el campo vuelve a su color habitual.
- required: Si se coloca este atributo dentro de una etiqueta input, el sistema comprobará que el campo ha sido llenado antes incluso de pulsar el botón de envío. Si el usuario deja este campo en blanco, se mostrará un mensaje de error o simplemente se colocará el cursor de escritura sobre el primer campo vacío.

Ejemplo de formulario con las nuevas etiquetas HTML5:



```
<!DOCTYPE HTML>
<html>
<body>
<form>

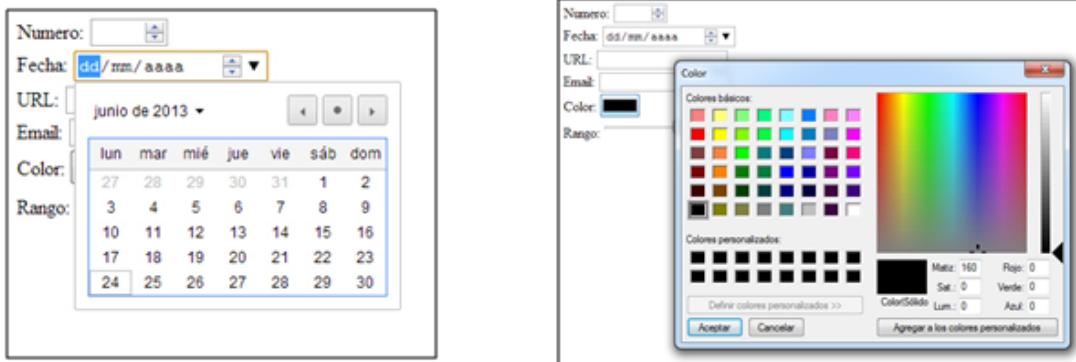
    Número: <input name="form_number" id="form_number" type="number" min="1" max="10"> <br>
    Fecha: <input name="form_date" id="form_date" type="date"> <br>
    URL: <input name="form_url" id="form_url" type="url" list="url_list"> <br>
    Email: <input name="form_email" id="form_email" type="email" list="email_list" multiple> <br>
    Color: <input name="form_color" id="form_color" type="color"> <br>
    Rango: <input name="x" type="range" min="100" max="700" step="9.090909" value="509.090909">

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

El resultado en el navegador sería:

Número:	<input type="text"/>
Fecha:	<input type="text"/> dd/mm/aaaa <input type="button"/>
URL:	<input type="text"/>
Email:	<input type="text"/>
Color:	<input type="color"/>
Rango:	<input type="range"/>

De tal forma que, por ejemplo, al pulsar sobre la flecha junto a el campo Fecha, se desplegaría un calendario y al pulsar sobre el color una carta de colores para elegir:



Hay que destacar que no todos los navegadores en la actualidad son compatibles con este tipo de campos en formularios. La especificación HTML5 está programada para que en el caso de ejecutarse el documento en un navegador no compatible, los tipos de <input> nuevos, se reconocen como de tipo texto, por lo que el formulario seguiría funcionando correctamente.

OTRAS ETIQUETAS HTML5

- <figure>...</figure>: Delimita una zona para incluir una imagen.
- <figcaption>...</figcaption>: Enmarca el título de una imagen dentro de la etiqueta <figure>.
- <hgroup>...</hgroup>: Sirve para agrupar títulos con subtítulos. Por ejemplo, poner un título con h1 seguido de un subtítulo con h2.
- <mark>...</mark>: Produce un efecto de resaltado de color sobre el texto que rodea.
- <time>...</time>: representa una hora o una fecha del calendario Gregoriano.

APIS PARA HTML5

Una API (del inglés Application Programming Interface), es una biblioteca de utilidades que añade funcionalidades a la programación web.

HTML5 permite la incorporación de varias APIs que van a permitir llevar a cabo actividades desde el navegador web y dispositivos móviles compatibles que no eran posibles con las versiones anteriores de HTML.

Uno de los objetivos principales de una API es el de normalizar el modo de trabajo de ciertos mecanismos y simplificar tareas de programación, que de lo contrario, serían bastante complejas. Las APIs son un aspecto muy importante dentro del entorno de HTML5 y hay una serie de ellas que conviene conocer y que se describen a continuación.

LA API DE GEOLOCALIZACIÓN

La API Geolocation permite que un navegador web pueda determinar la ubicación geográfica de un usuario, esta ubicación se da en coordenadas de Latitud y Longitud. Utiliza tecnología Javascript, que se inserta dentro de la página HTML5.

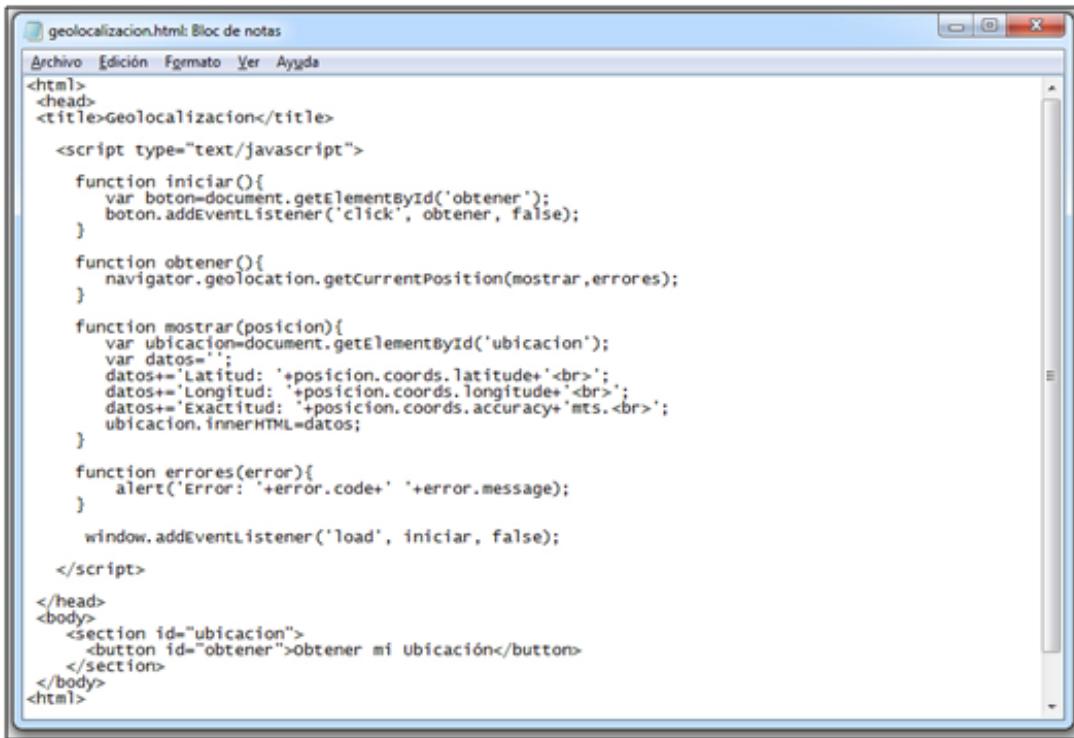
El navegador utilizará diferentes métodos para obtener la localización del visitante. Algunas de ellas son el GPS, la dirección IP, triangulación de antenas de redes móviles o bien la localización de la red WiFi que esté utilizando. Dependiendo del dispositivo y navegador, se utilizan unos u otros métodos para conseguir la geolocalización, por lo que el resultado puede no ser el mismo pero sí muy aproximado. De todos modos, la precisión es muy alta y nos permite conocer los datos más importantes, como por ejemplo, la ciudad. El valor “accuracy” muestra la precisión que tiene esa información.

La API define tres métodos:

- `getCurrentPosition()`: Recupera la posición actual y se ejecuta una sola vez.
- `watchPosition()`: Recupera y actualiza la posición actual a medida que cambia de posición.

- `clearWatch()`: Detiene la actualización de la posición.

El siguiente ejemplo de código hace que se muestre un botón en la pantalla con el nombre "Obtener mi ubicación". Al ser pulsado, se muestran en pantalla la longitud, latitud y precisión actuales.



```

geolocalizacion.html: Bloc de notas
Archivo Edición Formato Ver Ayuda
<html>
<head>
<title>Geolocalización</title>
<script type="text/javascript">
    function iniciar(){
        var boton=document.getElementById('obtener');
        boton.addEventListener('click', obtener, false);
    }
    function obtener(){
        navigator.geolocation.getCurrentPosition(mostrar, errores);
    }
    function mostrar(posicion){
        var ubicacion=document.getElementById('ubicacion');
        var datos="";
        datos+=`Latitud: ${posicion.coords.latitude}<br>`;
        datos+=`Longitud: ${posicion.coords.longitude}<br>`;
        datos+=`Exactitud: ${posicion.coords.accuracy} mts.<br>`;
        ubicacion.innerHTML=datos;
    }
    function errores(error){
        alert(`Error: ${error.code} ${error.message}`);
    }
    window.addEventListener('load', iniciar, false);
</script>
</head>
<body>
<section id="ubicacion">
<button id="obtener">Obtener mi ubicación</button>
</section>
</body>
<html>

```

Se observa que el código Javascript se escribe dentro de la zona `<head>` del documento HTML, y se delimita con las etiquetas `<script>...`
`</script>`.

Dentro del script se crean 4 funciones:

- Iniciar: crea una variable llamada botón y la programa como botón para que al pulsarlo se llame a la función obtener.
- Obtener: llama a la función de la API `getCurrentPosition`, los datos se almacenarán en la variable mostrar.
- Mostrar: toma los datos de la ubicación, y almacena los que se van a mostrar (latitud, longitud y exactitud) en una variable llamada datos. Mediante `innerHTML` se relacionan los datos y el documento HTML.

- Errores: Hace un control de los posibles errores y alertas que se pueden dar.

A continuación, dentro de las etiquetas <body>...</body> se crea la sección ubicación que contendrá los datos de geolocalización y el botón en HTML.

Más adelante se verán más funciones de Javascript, de momento sólo es necesario familiarizarse con el código.

LA API WEB STORAGE

La API Web Storage es básicamente una mejora de las Cookies. Las cookies permiten que los sitios web puedan guardar una pequeña información en los equipos de los usuarios, datos que normalmente sirven para volver a utilizarlos en momentos posteriores, y de esta manera, por ejemplo, ciertos sitios web recuerdan la información del usuario desde el último acceso.

Las cookies son una tecnología bastante limitada y no resulta fácil su utilización por parte de los diseñadores web. En HTML5, Web Storage actualiza este modelo para que las aplicaciones web puedan almacenar una cantidad de datos muy superior y que su acceso y utilización sea mucho más fácil y eficiente.

Web Storage ofrece dos maneras de guardar datos:

- sessionStorage: conservará los datos disponibles sólo durante la duración de la sesión de una página. De hecho, a diferencia de sesiones reales, la información almacenada a través de este mecanismo sólo es accesible desde una única ventana o pestaña y es preservada hasta que la ventana es cerrada.
- localStorage: Este mecanismo trabaja de forma similar a un sistema de almacenamiento para aplicaciones de escritorio. Los datos son grabados de forma permanente y se encuentran siempre disponibles para la aplicación que los creó.

Ambos mecanismos trabajan a través de la misma interfaz, compartiendo los mismos métodos y propiedades. En ambos casos la

información está disponible solo a través del sitio web o la aplicación que los creó.

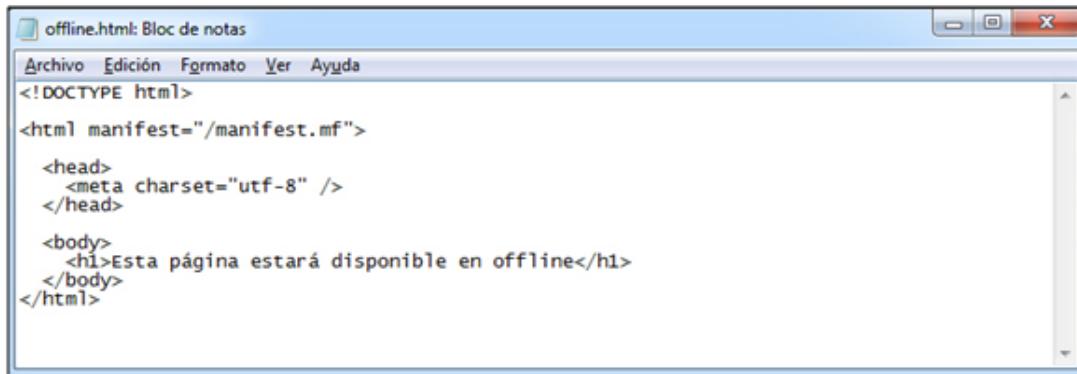
API DE TRABAJO OFFLINE

La API Offline Application Catching permite que una página web pueda ser consultada aún cuando no se tenga conexión a internet, ya que todo o parte de su contenido quedará almacenado en la memoria caché del ordenador.

Esto conlleva una serie de ventajas:

- Navegación sin conexión: los usuarios pueden explorar todo el sitio web sin conexión.
- Velocidad: los recursos almacenados en caché son locales y, por tanto, se cargan más rápido.
- Reducción de carga del servidor: el navegador sólo descarga los recursos del servidor que hayan cambiado.

Para habilitar la caché de aplicación para una web, se incluye el atributo "manifest" en la etiqueta html del documento, indicando dónde se encuentra un fichero llamado Caché Manifest:

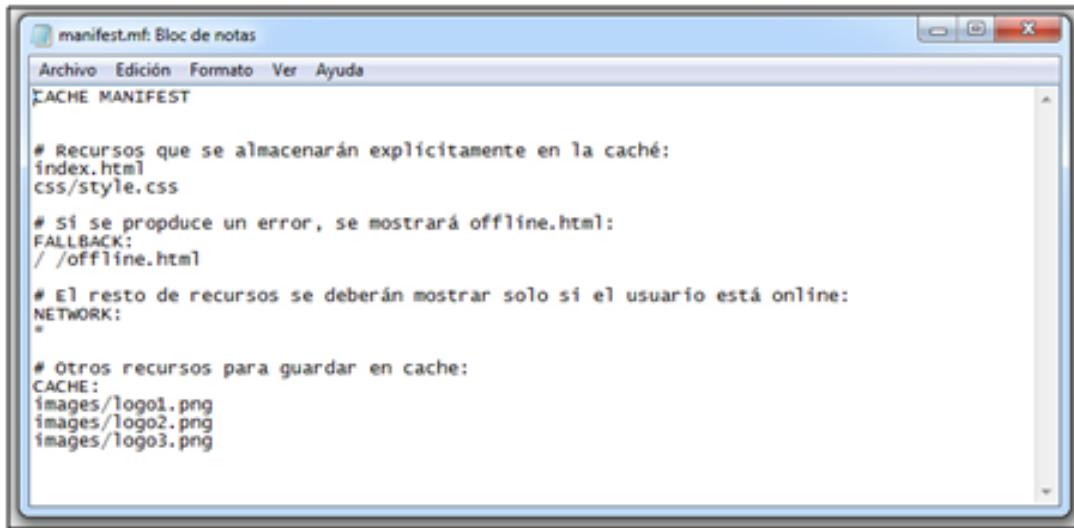


```
offline.html: Bloc de notas
Archivo Edición Formato Ver Ayuda
<!DOCTYPE html>
<html manifest="/manifest.mf">
  <head>
    <meta charset="utf-8" />
  </head>
  <body>
    <h1>Esta página estará disponible en offline</h1>
  </body>
</html>
```

El fichero Cache Manifest es un archivo de texto con cualquier extensión (se suele usar .mf), que deberá comenzar siempre con el texto CACHE MANIFEST en la primera línea. Un archivo de manifest puede incluir tres tipos de secciones en cualquier orden y que pueden aparecer varias veces:

- CACHE: Esta es la sección predeterminada para las entradas. Los archivos incluidos en esta sección (o inmediatamente después de CACHE MANIFEST) se almacenarán en caché explícitamente después de descargarse por primera vez.
- NETWORK: Los archivos incluidos en esta sección son recursos permitidos que requieren conexión al servidor. En todas las solicitudes enviadas a estos recursos se omite la caché, incluso si el usuario está trabajando sin conexión.
- Fallback: Se trata de una sección opcional en la que se especifican páginas alternativas en caso de no poder acceder a un recurso. La primera URL corresponde al recurso y la segunda, a la página alternativa.

Ejemplo de archivo de manifest:



```

manifest.manifest: Bloc de notas
Archivo Edición Formato Ver Ayuda
CACHE MANIFEST

# Recursos que se almacenarán explícitamente en la caché:
index.html
css/style.css

# Si se produce un error, se mostrará offline.html:
FALLBACK:
/ /offline.html

# El resto de recursos se deberán mostrar solo si el usuario está online:
NETWORK:
*

# Otros recursos para guardar en cache:
CACHE:
images/logo1.png
images/logo2.png
images/logo3.png

```

Las líneas precedidas con # son comentarios y no se tienen en cuenta.

El carácter * se llama carácter comodín, y significa que engloba todos los recursos que no estén en las otras secciones.

LA API DRAG AND DROP

Drag and Drop significa Arrastrar y Soltar, es decir, mediante esta API se puede arrastrar y mover prácticamente cualquier tipo de elemento de un elemento de una página web. Esta funcionalidad puede ser muy útil en

infinidad de aplicaciones web, por ejemplo, en un juego de ajedrez, para poder mover con el ratón las piezas por el tablero.

NOVEDADES EN CSS3

Muchos de los aspectos que se engloban bajo el nombre ‘HTML5’ son en realidad una combinación de las tecnologías HTML5 descritas antes junto con JavaScript o CSS. La última versión de CSS es la CSS 3.0, que ha ido evolucionando en paralelo con la especificación de HTML5. A continuación se describen brevemente algunas de las novedades de CSS3.

TRANSFORMACIONES

La propiedad transform de CSS3 permite rotar, cambiar la escala o sesgar un elemento de una página web. Por ejemplo, se puede girar levemente una foto dentro de una página para conseguir un efecto atractivo. También se puede dar animación a las transformaciones: por ejemplo, mediante una animación aplicada a la propiedad scale se puede conseguir un efecto de ampliación o reducción del tamaño de cualquier elemento. Se puede también añadir la propiedad perspective al efecto de transformación para simular la visión de un objeto en un espacio tridimensional, estático o en movimiento.

EFFECTOS VISUALES

Con CSS3 es posible aplicar mejoras muy interesantes a la apariencia de una página. Un ejemplo es la nueva propiedad ‘border-radius’, con la que se pueden poner esquinas redondeadas a los objetos rectangulares.

También se pueden crear muchos efectos nuevos, como gradientes de color o sombreados. Algunos efectos tradicionales, como background-image y la propiedad border han sido mejorados en CSS3. Por ejemplo, se puede utilizar la propiedad border-image para utilizar imágenes como bordes de objetos, o añadir varias imágenes de fondo a un mismo contenedor, evitando así la limitación actual a una sola imagen que permite la propiedad background-image.

La descripción del espacio de color RGBa es otra novedad de CSS3, ya que la "a" representa el grado de transparencia o alpha. Con la notación

RGBa de color ahora se pueden aplicar efectos de transparencia a cualquier objeto de la página.

FUENTES DE LETRA WEB

Cada vez está más extendido el soporte para añadir tipos de letra especiales a las páginas web mediante la propiedad @font-face, que permite especificar una fuente concreta y un enlace desde el cual el navegador pueda descargarla. Esta característica puede cambiar de forma radical el aspecto de las páginas web.

MEDIA QUERIES

Las Media Queries o consultas de tipos de medios, son un nuevo tipo de directiva CSS, y consisten en lanzar una consulta al navegador para determinar el tipo de pantalla en el cual se va a visualizar la página y a partir de esta información, enviar un estilo específico, optimizado para esas dimensiones. Por ejemplo, la misma página web, en un monitor con 2.000 pixels de ancho podría mostrarse con cuatro o incluso cinco columnas, pero en pantallas de 320 pixels, podría enviar una plantilla de estilos que solo utilice una columna.

TRANSICIONES

Una transición permite variar el valor de una propiedad CSS de manera continua a lo largo de un intervalo de tiempo definido. Por ejemplo, un botón con un fondo de color determinado puede cambiar de forma progresiva y suave a un color distinto cuando el usuario pasa el cursor sobre él.

Actualmente se puede conseguir este tipo de transiciones utilizando JavaScript y Flash, pero, al igual que sucede con otros muchos elementos de CSS3, las transiciones ofrecen a los diseñadores web una alternativa para conseguir los mismos resultados sin necesidad de convertirse en programadores expertos.

LO QUE HEMOS APRENDIDO

- HTML5 incorpora nuevas etiquetas HTML para hacer que la estructura de la página web sea más lógica y funcional.
- Donde antes sólo se podían utilizar etiquetas `<div>` para separar los contenidos, ahora existen `<header>`, `<nav>`, `<section>`, `<article>`, `<footer>`, etc.
- Con HTML5 se pueden insertar directamente reproductores de audio y vídeo, de forma similar a como se hacía con las imágenes.
- La etiqueta `<canvas>` dota a HTML5 de un formato de dibujo específico, y funciona como una superficie de dibujo dentro de la página web.
- HTML5 también incorpora nuevos tipos de etiquetas y atributos para formularios.
- Las APIs son bibliotecas de utilidades que añaden funcionalidades a la programación web, como por ejemplo geolocalización, almacenamiento web, trabajo offline, o arrastrar y soltar.
- La última versión de CSS, es decir CSS3, ha ido evolucionando paralelamente a la especificación HTML5, y permite aplicar estilos cada vez más sofisticados a las páginas web.

UNIDAD 1.3

HTML 5 PARTE 1^a

Creación de una Web Usando HTML y CSS

- Archivo de Reset CSS
- Definición de la Estructura de la Web
- Secciones y Estilos
- Más Estilos
- Prioridades en CSS

ARCHIVO DE RESET CSS

Todos los elementos HTML de una página web se muestran en pantalla utilizando los estilos por defecto que aplica cada navegador, es decir, los estilos que no se definen explícitamente en el propio código HTML o en algún archivo de estilo CSS, se definen en función de lo que tenga establecido cada tipo de navegador.

Por ejemplo, todos los elementos HTML incluyen márgenes por defecto. El estilo por defecto para un elemento `<h1>` incluye unos márgenes de 10 píxeles por encima y por debajo del texto.

Pero además, como se ha comentado, no todos los navegadores aplicarían estos valores por defecto, sino que un margen por defecto de 10 pixeles en un navegador, puede transformarse en 15 píxeles en otro navegador diferente.

Estas diferencias generan resultados no homogéneos a la hora de ver las páginas en distintos sistemas.

Si se quiere modificar el estilo de `h1` para que no lleve márgenes, habrá que indicar de manera explícita unas reglas que conviertan dichos valores a cero, por ejemplo para las etiquetas `<h1>`:

```
h1{  
margin-top:0px;  
margin-bottom:0px;  
}
```

Es por esto que existe una técnica para homogeneizar las presentaciones en pantalla, independientemente del navegador con el que se visualicen.

Esta técnica consiste en utilizar lo que se llama un archivo CSS de reset que elimina los estilos por defecto aplicados a los elementos HTML utilizados con más frecuencia. Con este reseteo de valores se consigue una base fiable y coherente sobre la cual construir los nuevos estilos.

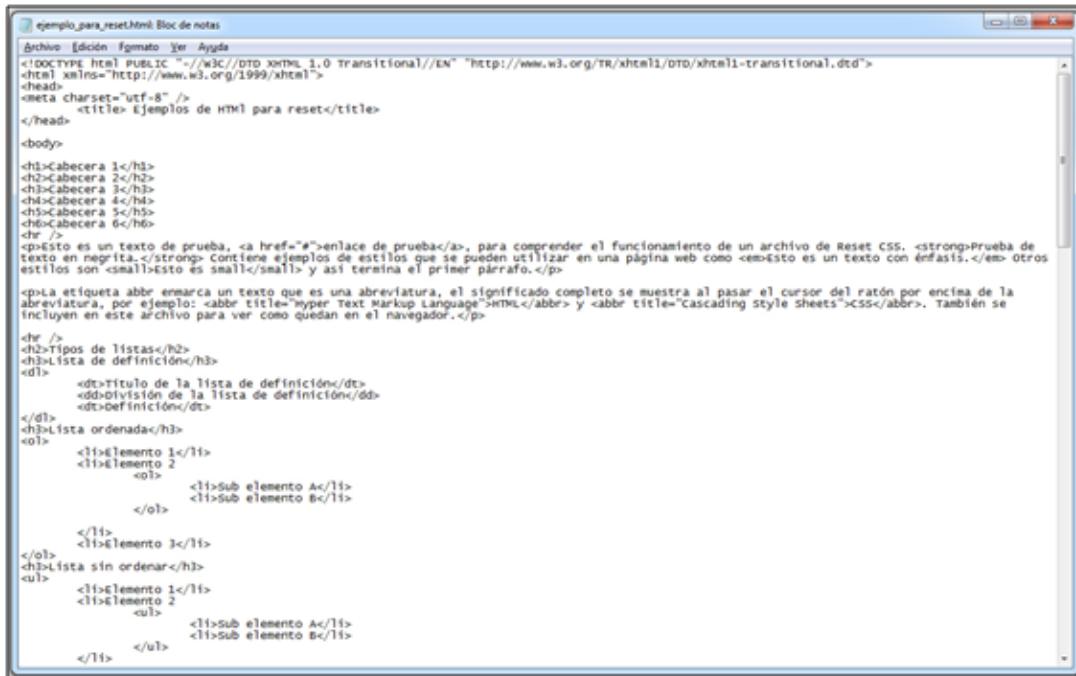
UTILIZACIÓN DE UN ARCHIVO DE RESET

A continuación se mostrará, mediante un ejercicio, cómo se utiliza un archivo de reset CSS:

Abrir El archivo ejemplo_para_reset.html (con el bloc de notas)

Observar El contenido del archivo, contiene una serie de elementos HTML genéricos, como títulos, párrafos, listas y formularios

No incluye estilos CSS.



```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta charset="utf-8" />
<title> Ejemplos de HTML para reset</title>
</head>
<body>
<h1>cabecera 1</h1>
<h2>cabecera 2</h2>
<h3>cabecera 3</h3>
<h4>cabecera 4</h4>
<h5>cabecera 5</h5>
<h6>cabecera 6</h6>
<br>
<p>Este es un texto de prueba, <a href="#">enlace de prueba</a>, para comprender el funcionamiento de un archivo de Reset CSS. <strong>Prueba de texto en negrita.</strong> Contiene ejemplos de estilos que se pueden utilizar en una página web como <em>Este es un texto con énfasis.</em> Otros estilos son <small>Este es small</small> y así termina el primer párrafo.</p>
<p>La etiqueta abbr encara un texto que es una abreviatura, el significado completo se muestra al pasar el cursor del ratón por encima de la abreviatura, por ejemplo: <abbr title="Hyper Text Markup Language">HTML</abbr> y <abbr title="Cascading Style Sheets">CSS</abbr>. También se incluyen en este archivo para ver como quedan en el navegador.</p>
<h2>Tipos de Listas</h2>
<h3>Lista de definición</h3>
<dl>
<dt>Título de la lista de definición</dt>
<dd>División de la lista de definición</dd>
<dt>Definición</dt>
</dl>
<h3>Lista ordenada</h3>
<ol>
<li>Elemento 1</li>
<li>Elemento 2
    <ol>
        <li>Sub elemento A</li>
        <li>Sub elemento B</li>
    </ol>
<li>Elemento 3</li>
</ol>
<h3>Lista sin ordenar</h3>
<ul>
<li>Elemento 1</li>
<li>Elemento 2
    <ul>
        <li>Sub elemento A</li>
        <li>Sub elemento B</li>
    </ul>
</li>
</ul>
```

Hacer doble clic En el archivo ejemplo_para_reset.html, para ejecutarlo

Observar Lo que aparece en el navegador de internet. En este caso se ha abierto con Internet Explorer 10

Cabecera 1

Cabecera 2

Cabecera 3

Cabecera 4

Cabecera 5

Cabecera 6

Este es un texto de prueba, [enlace de prueba](#), para comprender el funcionamiento de un archivo de Reset CSS. **Prueba de texto en negrita**. Contiene ejemplos de estilos que se pueden utilizar en una página web como *Este es un texto con diseño*. Otros estilos son *Este es small* y así termina el primer párrafo.

La etiqueta abre enmarca un texto que es una abreviatura, el significado completo se muestra al pasar el cursor del ratón por encima de la abreviatura, por ejemplo: HTML y CSS. También se incluyen en este archivo para ver como quedan en el navegador.

Tipos de listas

Lista de definición

Título de la lista de definición

División de la lista de definición

Definición

Lista ordenada

1. Elemento 1
2. Elemento 2
 - 1. Sub elemento A
 - 2. Sub elemento B
3. Elemento 3

Lista sin ordenar

- Elemento 1
- Elemento 2
 - Sub elemento A
 - Sub elemento B
- Elemento 3

Según el tipo de navegador que se esté utilizando, se aplican los estilos que vienen definidos por defecto para cada tipo de elemento. Si se abre el archivo por ejemplo con Firefox, estos estilos serán ligeramente diferentes, sobre todo en la parte de formularios.

Asignar El archivo reset.css como archivo de estilos para el documento HTML, para ello, escribir la siguiente línea de código en el archivo ejemplo_para_reset.html:



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta charset="utf-8" />
<title>Ejemplos de HTML para reset</title>
<link href="reset.css" rel="stylesheet" type="text/css">
</head>
<body>
```

Desplegar menú Archivo

Seleccionar Guardar como



Guardar El archivo con el nombre ejemplo_para_reset_con_css.html.

Ejecutar El archivo ejemplo_para_reset_con_css.html.

Observar Lo que ha ocurrido, la apariencia ha cambiado, y se han reseteado los estilos

Cabecera 1
Cabecera 2
Cabecera 3
Cabecera 4
Cabecera 5
Cabecera 6

Este es un texto de prueba, [enlace de prueba](#), para comprender el funcionamiento de un archivo de Reset CSS. **Prueba de texto en negrita**. Contiene ejemplos de estilos que se pueden utilizar en una página web como *Este es un texto con énfasis*. Otros estilos son *Este es small* y así termina el primer párrafo.
La etiqueta abre enmarca un texto que es una abreviatura, el significado completo se muestra al pasar el cursor del ratón por encima de la abreviatura, por ejemplo: HTML y CSS. También se incluyen en este archivo para ver como quedan en el navegador.

Tipos de listas
Lista de definición
Título de la lista de definición
División de la lista de definición
Definición
Lista ordenada
Elemento 1
Elemento 2
Sub elemento A
Sub elemento B
Elemento 3
Lista sin ordenar
Elemento 1
Elemento 2
Sub elemento A
Sub elemento B
Elemento 3

Tabla
La etiqueta caption define el contenido sobre la tabla
Cabecera de tabla 1 Cabecera de tabla 2 Cabecera de tabla 3
Celda 1-1 Celda 1-2 Celda 1-3
Celda 2-1 Celda 2-2 Celda 2-3
Celda 3-1 Celda 3-2 Celda 3-3

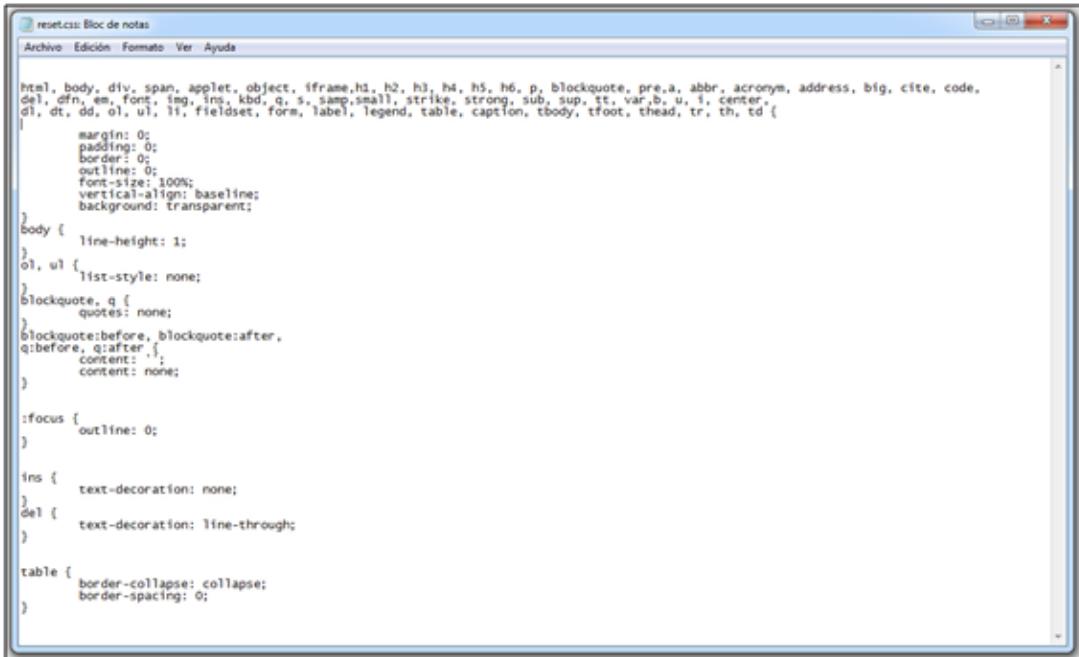
Texto preformatado
En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero.

Citas en Bloque (Blockquotes)
Confía en el tiempo, que suele dar dulces salidas a muchas amargas dificultades.
Miguel de Cervantes

Se han eliminado los márgenes y distancias al borde. Se han reseteado también otros muchos estilos, como por ejemplo las listas de datos, donde se han eliminado los puntos de las líneas en las listas no ordenadas, así como los números de orden en las listas ordenadas.

Editar El archivo reset.css con el bloc de notas

Observar Los estilos aplicados a cada tipo de elemento HTML



```
reset.css: Bloc de notas
Archivo Edición Formato Ver Ayuda

html, body, div, span, applet, object, iframe, h1, h2, h3, h4, h5, h6, p, blockquote, pre, a, abbr, acronym, address, big, cite, code, del, dfn, em, font, img, ins, kbd, q, s, samp, small, strike, strong, sub, sup, tt, var, b, u, i, center, dl, dt, dd, ol, ul, li, fieldset, form, label, legend, table, caption, tbody, tfoot, thead, tr, th, td {
    margin: 0;
    padding: 0;
    border: 0;
    outline: 0;
    font-size: 100%;
    vertical-align: baseline;
    background: transparent;
}

body {
    line-height: 1;
}

ol, ul {
    list-style: none;
}

blockquote, q {
    quotes: none;
}

blockquote:before, blockquote:after,
q:before, q:after {
    content: '';
    content: none;
}

:focus {
    outline: 0;
}

ins {
    text-decoration: none;
}

del {
    text-decoration: line-through;
}

table {
    border-collapse: collapse;
    border-spacing: 0;
}
```

Las hojas de estilos de reset son opcionales. Ayudan a normalizar la presentación en los distintos navegadores. También algunos diseñadores incluyen en ellas sus propios estilos de uso frecuente para establecer los valores iniciales de los elementos, como por ejemplo el tipo de letra que se usa con más frecuencia, el espacio de separación entre párrafos, etc. De esta forma se dispone de una serie de reglas básicas predefinidas que se pueden utilizar para empezar cualquier proyecto web.

DEFINICIÓN DE LA ESTRUCTURA DE LA WEB

El primer paso a la hora de comenzar con un nuevo proyecto de página web es determinar cómo se estructurarán los contenidos dentro de la misma.

ANCHO FIJO VS VARIABLE

Hay dos categorías principales de estructuras en base a su anchura:

- Estructuras de ancho fijo: Todos los elementos de la página se anidan dentro de un contenedor de tipo <div> que tiene definido un ancho especificado. Un valor habitual son 960 píxeles de ancho. Es una opción muy cómoda para el diseñador ya que permite ubicar exactamente los distintos elementos como cabeceras, barras laterales y pie de página.
- Estructuras de ancho variable: Están pensadas para adaptarse a la anchura de la ventana del navegador en cualquier momento. Este tipo de disposiciones es útil cuando los usuarios acceden a la web desde pantallas con resoluciones muy variadas ya que permiten el reposicionamiento en pantalla de textos e imágenes.

Puesto que los dispositivos móviles suponen actualmente una proporción importante de los navegadores en uso en el mercado, este tipo de estructuras por lo general se prefieren para ofrecer un mejor resultado frente a las estructuras de ancho fijo, que habitualmente tienen problemas por falta de espacio (sobre todo en Smartphone). Es habitual referirse a este tipo de estructuras como diseño Responsive. Las estructuras de ancho variable son más difíciles de programar y obligan al diseñador a resolver una serie de asuntos adicionales.

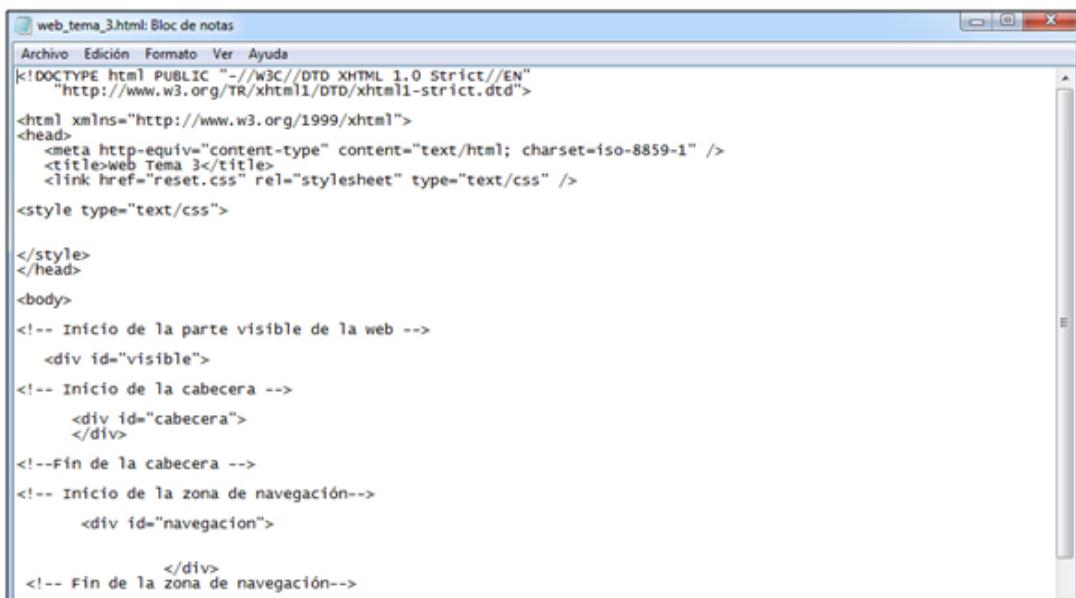
ESTRUCTURA DE LOS ELEMENTOS DE LA WEB

Para comenzar con el aprendizaje se creará una web con estructura de ancho fijo, compuesta por una cabecera, dos columnas y un pie de

página.

Esta página empleará una serie de etiquetas <div> para generar su estructura. Cada elemento <div> será un contenedor genérico en el que se colocarán otros elementos, incluso otros contenedores <div> y llevará asociado un atributo id, lo cual permitirá hacer referencia a él para aplicar a cada contenedor estilos CSS propios.

Editar El archivo web_tema_3.html con el bloc de notas



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1" />
<title>web Tema 3</title>
<link href="reset.css" rel="stylesheet" type="text/css" />
<style type="text/css">

</style>
</head>
<body>
<!-- Inicio de la parte visible de la web -->
<div id="visible">
<!-- Inicio de la cabecera -->
<div id="cabecera">
</div>
<!-- Fin de la cabecera -->
<!-- Inicio de la zona de navegación-->
<div id="navegacion">
</div>
<!-- Fin de la zona de navegación-->

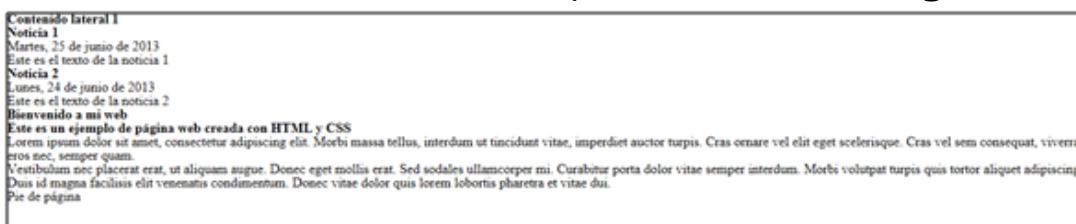
```

Se observa que contiene la estructura de la página prediseñada, a base de etiquetas <div>...</div>, y algunos contenidos. También se incluyen líneas de comentarios para facilitar la comprensión del código.

En la parte superior se observa que dentro de la etiqueta <head> se ha asignado el archivo de estilos reset.css.

La etiqueta <div id="visible"> es el comienzo de la sección de la página que contiene el resto de elementos.

Ejecutar El archivo web_tema_3.html para verlo en el navegador



Contenido lateral 1
Noticia 1
Martes, 25 de junio de 2013
Este es el texto de la noticia 1
Noticia 2
Lunes, 24 de junio de 2013
Este es el texto de la noticia 2
Bienvenido a mi web
Este es un ejemplo de página web creada con HTML y CSS
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi massa tellus, interdum ut tincidunt vitae, imperdiet auctor turpis. Cras ornare vel elit eget scelerisque. Cras vel sem consequat, viverra eros nec, semper quam.
Vestibulum nec placerat erat, ut aliquam augue. Donec egestas mollis erat. Sed sodales ullamcorper mi. Curabitur porta dolor vitae semper interdum. Morbi volutpat turpis quis tortor aliquet adipiscing.
Duis id magna facilisis elit venenatis condimentum. Donec vitae dolor quis lorem lobortis pharetra et vitae dui.
Pie de página

Tal y como está ahora, el documento HTML tiene prácticamente todos los estilos reseteados gracias al archivo de reset CSS.

Guardar El archivo con otro nombre, por ejemplo web_tema_3_mod.html.

A partir de ahora se irán haciendo modificaciones al archivo, por lo que se recomienda tener siempre una copia de seguridad del archivo original.

SECCIONES Y ESTILOS

CONTENEDOR PRINCIPAL

Añadir El siguiente código dentro de las etiquetas <style> y </style>



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1" />
<title>web Tema 3</title>
<link href="reset.css" rel="stylesheet" type="text/css" />
<style type="text/css">
#visible {
background-color:#FFB973;
width:960px;
border:thin solid black;
margin:0 auto;
}
</style>
</head>
```

De esta forma se agregan estilos al elemento <div id="visible"> mediante la etiqueta style. Como se ha comentado, este elemento contiene toda la parte visible de la web. Se ha asignado un color de fondo, un ancho fijo de 960 pixeles y un borde. También se configuran los bordes, de forma que el primer parámetro "0" corresponde a los márgenes superior e inferior. El segundo parámetro se establece en "auto", quiere decir que los márgenes derecho e izquierdo se alinearán automáticamente. De esta forma el contenido aparecerá centrado en la pantalla.

Guardar El archivo

Observar El resultado en el navegador. Aunque se cambie el tamaño de pantalla del navegador, el ancho permanece fijo. Si el tamaño navegador es menor a 960 pixeles, el contenido aparece cortado.



CABECERA

Añadir El siguiente código dentro del contenedor <div> de la cabecera

```

<!-- Inicio de la cabecera -->
<div id="cabecera">
    
</div>
<!--Fin de la cabecera -->

```

Insertará la imagen milogo.jpg que está en la carpeta images. Tendrá un ancho y alto de 200 y 123 píxeles respectivamente.

Escribir La siguiente regla de estilo (para cambiar el color de fondo de cabecera)

```

<style type="text/css">
    #visible {
        background-color:#FFB973;
        width:960px;
        border:thin solid black;
        margin:0 auto;
    }
    #cabecera {
        background-color:#FFF;
    }
</style>

```

Ahora el fondo de la cabecera es de color blanco.

Comprobar Los cambios en el navegador



COLUMNAS DE CONTENIDO

Añadir Las siguientes reglas de estilos a la página:

```

<style type="text/css">

    #visible {
        background-color:#FFB973;
        width:960px;
        border :thin solid black;
        margin:0 auto;
    }

    #cabecera {
        background-color :#FFF;
    }

    #lateral {
        float:right;
        width:300px;
        background-color :#FFE599;
    }

    #principal {
        float:left;
        width:600px;
        background-color :#BFCFFF;
    }

</style>

```

De esta forma se han definido los estilos para los contenedores <div id="lateral"> y <div id="principal">. A la columna lateral se le asigna un ancho de 300 píxeles, y a la principal de 600 píxeles. En ambos casos se establecen colores de fondo diferentes.

La propiedad float hace que el elemento se sitúe de manera "flotante" sobre la pantalla, alineándose a la izquierda (float:left) o a la derecha (float:right) del contenedor en el que se encuentra. No se puede centrar un objeto utilizando esta propiedad.

PIE DE PÁGINA

Añadir La siguiente regla de estilo para el pie de página:

```

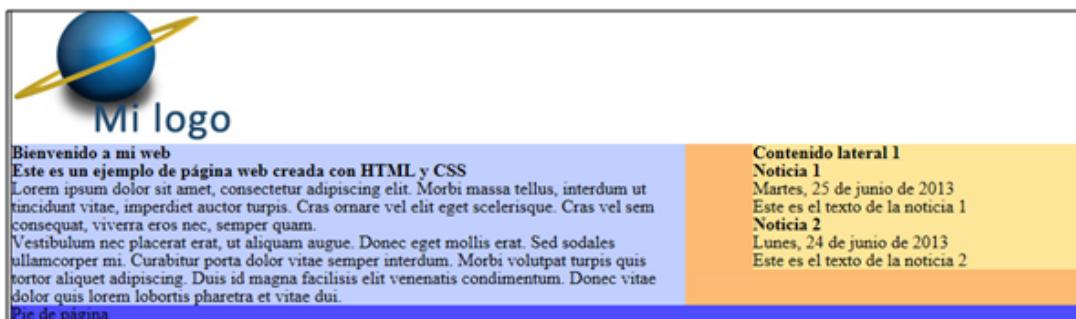
#piedepagina {
    clear:both;
    background-color :#4D4dff;
}

```

La propiedad clear de CSS indica que el elemento no admite elementos flotantes a su lado. Se puede especificar si se quiere que

no haya elementos flotantes a la izquierda, a la derecha o en ambos lados. En el caso del pie de página, se desea eliminarlos de ambos lados, por eso se utiliza both.

Observar resultado en el navegador.



MENÚ DE NAVEGACIÓN

Se ha dejado este elemento de la estructura para el final, debido a que es el más complejo.

Añadir La regla de estilos para el elemento #navegación entre las etiquetas <style> y </style>:

```
#navegacion {  
    background-color :#999;  
    height :40px  
}
```

Se asigna un color de fondo gris y una altura de 40 píxeles.

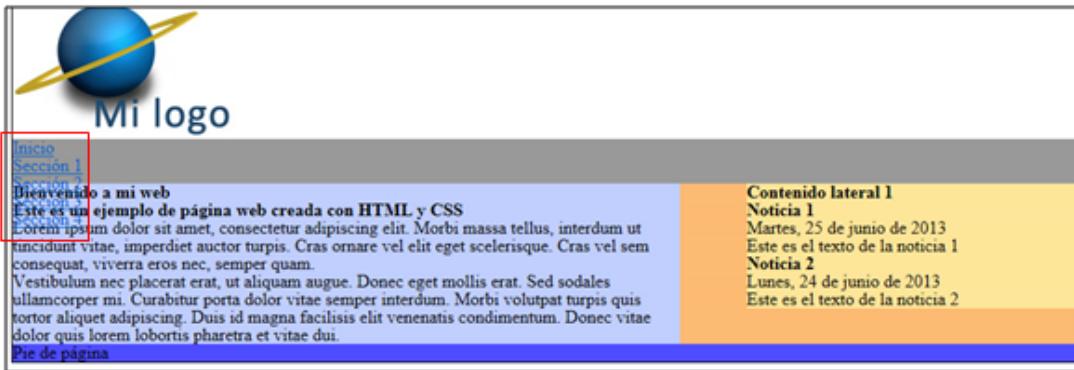
Añadir Los enlaces de las distintas secciones del menú mediante una lista no ordenada, dentro del contenedor <navegación>:

```
<!-- Inicio de la zona de navegación-->  


<ul>  
        <li><a href="inicio.html">Inicio</a></li>  
        <li><a href="seccion1.html">Sección 1</a></li>  
        <li><a href="seccion2.html">Sección 2</a></li>  
        <li><a href="seccion3.html">Sección 3</a></li>  
        <li><a href="seccion4.html">Sección 4</a></li>  
    </ul>  
</div>  
<!-- Fin de la zona de navegación-->


```

Observar El resultado en el navegador



El menú aparece superpuesto debido a que la lista está pisando a la columna principal flotante. Además la lista no tiene los puntos indicadores, ya que la página está asociada a una hoja de estilo CSS de reset y una de sus reglas declara la propiedad `list-style:none`, que elimina los puntos indicadores en las listas. En el caso de este ejemplo, la falta de puntos es deseable, ya que se va a utilizar la lista como menú de navegación.

Añadir La siguiente línea de código después de la regla de estilo `#navegación:`

```
#navegacion {  
    background-color:#999;  
    height:40px  
}  
  
#navegacion li {  
    float:left;  
    width:170px;  
    height:40px;  
    background-color:#CCC;  
    text-align:center;  
    border-left:1px black solid;  
    border-right:1px black solid;  
    line-height:40px;  
}
```

Se ha creado una regla CSS llamada selector de contexto, es decir se aplica exclusivamente a los elementos de lista (li) que hay dentro del `<div id="navegación">`.

Por tanto solamente a cada uno de los elementos de tipo lista dentro del menú de navegación se le aplican las propiedades: flotante a la izquierda, ancho y alto determinados, color de fondo, alineación de texto centrada, bordes a la derecha y a la izquierda, y altura del texto.

Comprobar El resultado en el navegador



Mi logo

[Inicio](#)

[Sección 1](#)

[Sección 2](#)

[Sección 3](#)

[Sección 4](#)

Bienvenido a mi web

Este es un ejemplo de página web creada con HTML y CSS

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi massa tellus, interdum ut tincidunt vitae, imperdiet auctor turpis. Cras ornare vel elit eget scelerisque. Cras vel sem consequat, viverra eros nec, semper quam.

Vestibulum nec placerat erat, ut aliquam augue. Donec eget mollis erat. Sed sodales ullamcorper mi. Curabitur porta dolor vitae semper interdum. Morbi volutpat turpis quis tortor aliquet adipiscing. Duis id magna facilisis elit venenatis condimentum. Donec vitae dolor quis lorem lobortis pharetra et vitae dui.

[Pie de página](#)

Contenido lateral 1

Noticia 1

Martes, 25 de junio de 2013

Este es el texto de la noticia 1

Noticia 2

Lunes, 24 de junio de 2013

Este es el texto de la noticia 2

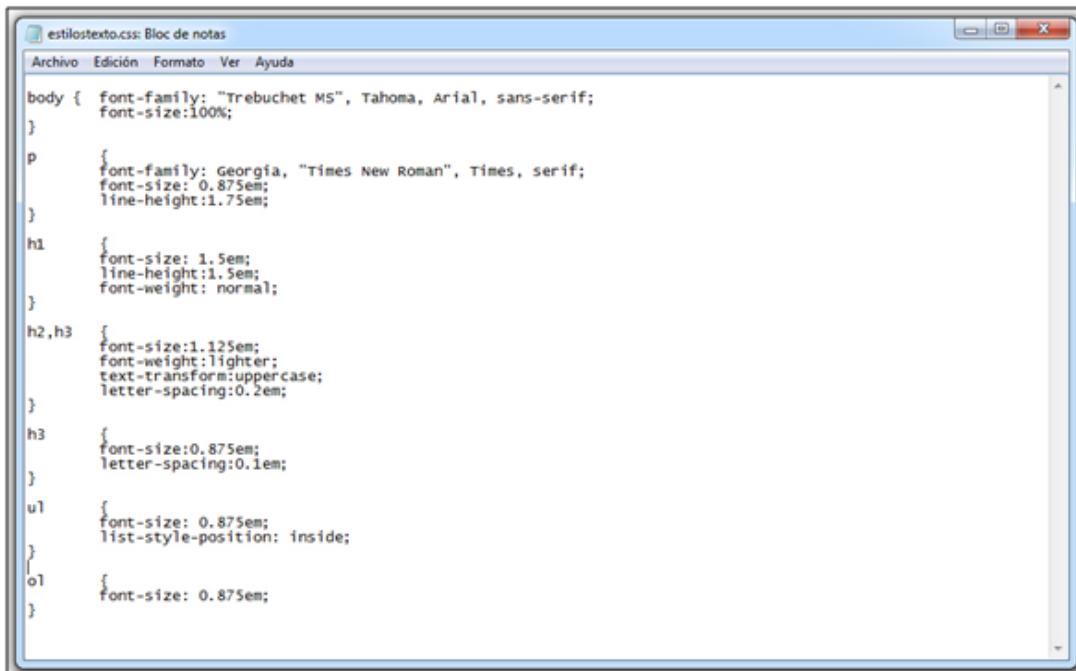
MÁS ESTILOS

ESTILOS SOBRE EL TEXTO

Hasta ahora, por comodidad, se han añadido los estilos de estructura a través de la etiqueta <style> dentro del mismo documento HTML.

Ahora se aplicará una nueva hoja de estilos externa para las propiedades de los diferentes tipos de texto.

Abrir El archivo estilostexto.css con el bloc de notas



```
estilostexto.css: Bloc de notas
Archivo Edición Formato Ver Ayuda
body { font-family: "Trebuchet MS", Tahoma, Arial, sans-serif;
font-size:100%; }
p {
font-family: Georgia, "Times New Roman", Times, serif;
font-size: 0.875em;
line-height:1.75em;
}
h1 {
font-size: 1.5em;
line-height:1.5em;
font-weight: normal;
}
h2,h3 {
font-size:1.125em;
font-weight:lighter;
text-transform:uppercase;
letter-spacing:0.2em;
}
h3 {
font-size:0.875em;
letter-spacing:0.1em;
}
ul {
font-size: 0.875em;
list-style-position: inside;
}
ol {
font-size: 0.875em;
}
```

Observar Las diferentes propiedades que se aplican al texto:

- **font-family:** establece una lista ordenada de fuentes tipográficas que se usarán para mostrar el texto del elemento. Si la primera fuente de la lista no está instalada en el ordenador desde el que se accede al sitio, se seguirá probando con la siguiente fuente hasta encontrar una fuente apropiada. Hay 3 tipos de familias de fuentes:
 - **Serif:** Se caracterizan por tener prolongaciones decorativas en los extremos. Ejemplos: Times New Roman, Garamond, Georgia.

- Sans-serif: Se caracterizan por no tener prolongaciones decorativas en los extremos. Ejemplos: Trebuchet, Arial, Verdana.
- Monospace: Se caracterizan porque todos los caracteres tienen un ancho fijo. Ejemplos: Courier, Courier New, Andeles Mono.
- font-size: Establece el tamaño de la fuente. Puede indicarse de varias formas diferentes, las más comunes son:
 - Unidades absolutas: Tamaño en píxeles (px) o en puntos (pt).
 - Constante: se puede asignar uno de los siguientes valores, de menor a mayor tamaño: xx-small, x-small, small, medium, large, x-large, xx-large.
 - Unidades relativas: El tamaño de letra se establece tomando como referencia el valor em, que es el tamaño base de fuente del navegador. Por ejemplo font-size:0.5em establecerá un tamaño de fuente que será la mitad del tipo base.
 - Unidades porcentuales: El tamaño de letra se establece tomando como referencia el valor em, que es el tamaño base de fuente del navegador, de forma que 1em=100%. Por ejemplo: font-size: 50% establecerá un tamaño de fuente que será la mitad del tipo base.
- line-height: Establece la altura de cada línea que forma el contenido de texto de un elemento, por lo que se emplea para controlar el interlineado del texto. Se expresa con los mismos tipos de unidades que la propiedad font-size.
- font-weight: Establece el grosor de cada letra. Los valores típicos de menor a mayor son lighter, normal, bold y bolder.
- text-transform: Transforma el texto de un elemento para mostrarlo en mayúsculas, minúsculas o una mezcla de ambas. Puede ser:

- capitalize: Pone en mayúscula sólo la primera letra de cada palabra.
 - uppercase: Todo en mayúsculas.
 - lowercase: Todo en minúsculas.
 - none: No aplica ninguna transformación.
- letter-spacing: Establece la separación entre las letras del texto. Se expresa con los mismos tipos de unidades que la propiedad font-size.
 - list-style-position: establece la posición del marcador de un elemento de lista respecto al contenido de ese mismo elemento. El valor por defecto que aplican los navegadores es outside, que hace que el marcador se muestre fuera de la caja invisible que encierra al contenido del elemento. El otro valor disponible es inside, que hace que el marcador se muestre dentro de la caja invisible que encierra al contenido del elemento.

Aplicar La hoja de estilos al documento html web_tema_3_mod.html, escribiendo la siguiente línea de código:



```

web_tema_3_mod.html: Bloc de notas
Archivo Edición Formato Ver Ayuda
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1" />
<title>Web Tema 3</title>
<link href="reset.css" rel="stylesheet" type="text/css" />
<link href="estilostexto.css" rel="stylesheet" type="text/css" />
```

Guardar El archivo web_tema_3_mod.html.

Comprobar El resultado en el navegador ejecutando el archivo web_tema_3_mod.html haciendo doble clic sobre él.



Mi logo

Inicio	Sección 1	Sección 2	Sección 3	Sección 4
Bienvenido a mi web ESTE ES UN EJEMPLO DE PÁGINA WEB CREADA CON HTML Y CSS Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi massa tellus, interdum ut tincidunt vitae, imperdiet auctor turpis. Cras ornare vel elit eget scelerisque. Cras vel sem consequat, viverra eros nec, semper quam. Vestibulum nec placerat erat, ut aliquam augue. Donec eget mollis erat. Sed sodales ullamcorper mi. Curabitur porta dolor vitae semper interdum. Morbi volutpat turpis quis tortor aliquet adipiscing. Duis id magna facilisis elit venenatis condimentum. Donec vitae dolor quis lorem lobortis pharetra et vitae dui.				CONTENIDO LATERAL 1 NOTICIA 1 Martes, 25 de junio de 2013 Este es el texto de la noticia 1 NOTICIA 2 Lunes, 24 de junio de 2013 Este es el texto de la noticia 2
Pie de página				

MÁRGENES

Debido a la utilización del archivo de reset, todos los márgenes entre el texto y los bordes están puestos a cero por defecto. Sin embargo, es evidente que es necesario establecer ciertos márgenes para mejorar la presentación de la web.

Existen dos formas de aplicar márgenes:

- Estableciendo un margen interior (padding) al elemento contenedor.
- Estableciendo un margen exterior (margin) a los elementos que hay dentro del contenedor, como párrafos (<p>), etc.

En cada caso concreto será mejor utilizar un método u otro. A continuación se describirá el efecto que producen ambos métodos en el ejemplo que se ha ido desarrollando en el tema.

Márgenes Usando Padding

Editar El archivo web_tema_3_mod.html con el bloc de notas

Agregar Las dos siguientes líneas de código CSS:

```

#lateral {
    float:right;
    width:300px;
    background-color:#FFE599;
    padding:0px 20px 0px 20px;
}

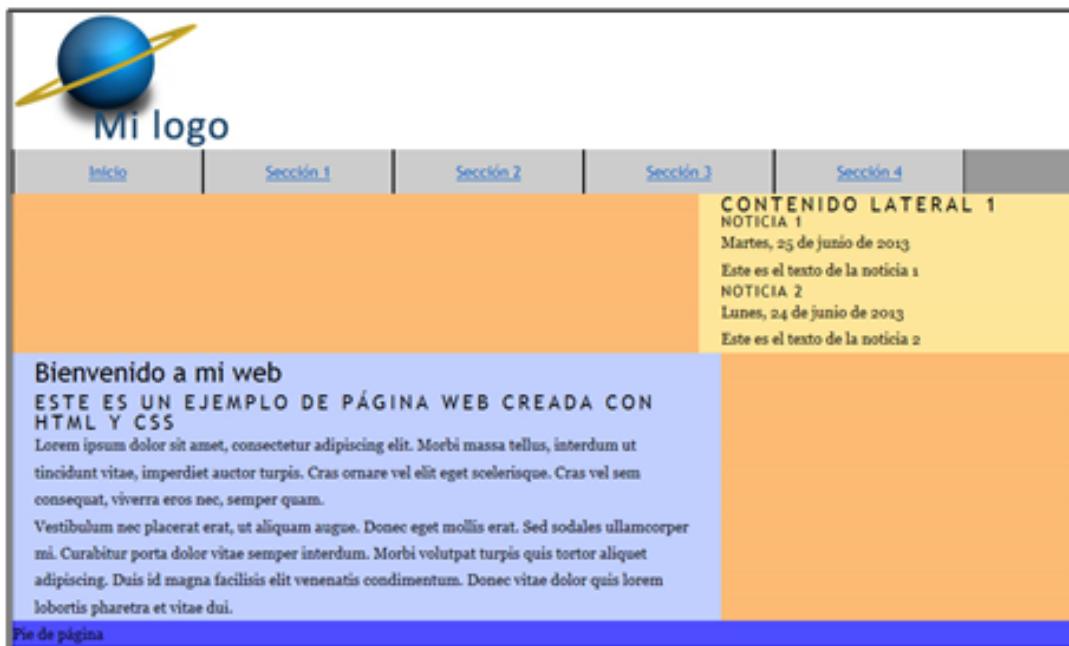
#principal {
    float:left;
    width:600px;
    background-color:#BFCFFF;
    padding:0px 20px 0px 20px;
}

```

En ambos casos, el primer valor de la propiedad padding (0px) es el margen interior con respecto al borde superior. El segundo valor (20px) corresponde a la separación con respecto al borde derecho. El tercero (0px) es la separación con respecto al borde inferior y el ultimo (20px) es el margen interior de separación con el borde izquierdo.

Guardar El archivo

Comprobar El resultado en el navegador



Se observa que la anchura total de las dos columnas, una vez añadido el padding, es mayor que la del contenedor donde están anidadas. Por tanto div "principal" se ha desplazado hasta el único

sitio donde puede desplegarse por completo, que es debajo del div "lateral".

Esto podría resolverse aumentando la anchura total del div "visible", reduciendo el ancho de "lateral" o de "principal", pero también utilizando el segundo método para incluir márgenes, que se explica a continuación.

Márgenes Usando Margin

Editar El archivo web_tema_3_mod.html con el bloc de notas.

Borrar Las dos líneas de código CSS que se escribieron en el apartado anterior:

```
#lateral {  
    float:right;  
    width:300px;  
    background-color:#FFE599;  
    padding:0px 20px 0px 20px;  
}  
  
#principal {  
    float:left;  
    width:600px;  
    background-color:#BFCFFF;  
    padding:0px 20px 0px 20px;  
}
```

Añadir Las siguientes reglas de estilo antes de la etiqueta </style>:

```
#lateral {  
    float:right;  
    width:300px;  
    background-color:#FFE599;  
}  
  
#principal {  
    float:left;  
    width:600px;  
    background-color:#BFCFFF;  
}  
  
.piepagina {  
    clear:both;  
    background-color:#4D4dff;  
}  
  
#lateral p, #lateral h2, #lateral h3, #principal p, #principal h1, #principal h2, #principal h3{  
    margin-left:20px;  
    margin-right:20px;  
}  
#principal p {  
    margin-left:30px;  
}  
#lateral h2 {  
    margin-top:15px;  
}  
p, h1, h2, h3 {  
    margin-bottom:20px;  
}  
  
/style>
```

Mediante una serie de reglas de selector de contexto, se aplican márgenes exteriores a los elementos que están dentro de los contenedores 'lateral' y 'principal'. Por otro lado también se añaden

reglas que sobrescriben parcialmente a las anteriores, y también márgenes para el caso general de párrafos y títulos.

Guardar El archivo

Comprobar El resultado en el navegador

The screenshot shows a web page with a header containing a logo and the text "Mi logo". Below the header is a navigation menu with five items: "Inicio", "Sección 1", "Sección 2", "Sección 3", and "Sección 4". The main content area has a light blue background and contains the following text:
Bienvenido a mi web
ESTE ES UN EJEMPLO DE PÁGINA WEB CREADA CON HTML Y CSS
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi massa tellus, interdum ut tincidunt vitae, imperdiet auctor turpis. Cras ornare vel elit eget scelerisque. Cras vel sem consequat, viverra eros nec, semper quam.
Vestibulum nec placerat erat, ut aliquam augue. Donec eget mollis erat. Sed sodales ullamcorper mi. Curabitur porta dolor vitae semper interdum. Morbi volutpat turpis quis tortor aliquet adipiscing. Duis id magna facilisis elit venenatis condimentum. Donec vitae dolor quis lorem lobortis pharetra et vitae dui.

The sidebar on the right has an orange background and is titled "CONTENIDO LATERAL 1". It contains two news items:
NOTICIA 1
Martes, 25 de junio de 2013
Este es el texto de la noticia 1
NOTICIA 2
Lunes, 24 de junio de 2013
Este es el texto de la noticia 2

At the bottom of the page is a blue footer bar with the text "Pie de página".

El resultado ahora es notablemente más atractivo.

Este segundo método tiene la ventaja de que el posicionamiento de la columna es más predecible, puesto que solo hay que tener en cuenta una anchura total. Su desventaja es que se necesita más código y que hay que prestar atención a los detalles, viendo cómo se posicionan los elementos individualmente.

Lo habitual es aplicar una combinación de ambos métodos.

IMÁGENES COMO FONDOS

Resulta interesante definir los fondos de las secciones mediante alguna imagen. En el ejercicio se colocará la imagen piepagina.jpg como fondo para el contenedor del pie de página.

Editar El archivo web_tema_3_mod.html con el bloc de notas

Añadir El siguiente texto como contenido del pie de página:

```
<!-- Inicio del pie de página -->
<div id="piepagina">
    <p>Copyright 2013 Mi web en HTML y CSS </p>
    <p>Este sitio es un ejercicio sobre cómo utilizar HTML y css para la creación de una página web.</p>
</div>
<!-- Fin del pie de página -->
```

Añadir Las siguientes reglas para #piedepagina en la hoja de estilos interna:

```
#piedepagina {  
    clear:both;  
    background-color:#4D4dff;  
    background-image:url(images/piedepagina.jpg);  
    background-repeat:no-repeat;  
    width:960px;  
    height:128px;  
    padding-top:10px;  
}  
#piedepagina p {  
    margin:10px 0px 0px 20px;  
    width:280px;  
    font-family:verdana, Geneva, sans-serif;  
    font-size:0.689em;  
}
```

Se añade la propiedad background-image, que indica la dirección donde se encuentra la imagen para el fondo.

La propiedad background-repeat: no repeat, indica que la imagen sólo debe mostrarse una vez y no repetirse a modo de baldosa por el espacio del contenedor.

Las propiedades width y height indican el ancho y el alto en píxels de la imagen.

También se ha añadido una regla de selector de contexto para dar formato a los párrafos que se incluyan dentro del pie de página.

Guardar El archivo web_tema_3_mod.html.

Comprobar El resultado en el navegador



EFFECTOS SOBRE LOS ENLACES

Los enlaces poseen varias pseudoclases. Se pueden definir estilos propios para cada una de ellas:

- a: hover: Se activa cuando el usuario pasa el ratón por encima del enlace.
- a: link: Es la apariencia por defecto del enlace.
- a: active: Se activa cuando el usuario pulsa con el ratón sobre el enlace.
- a: visited: El estilo se aplica a todos los enlaces que han sido visitados al menos una vez por el usuario.

A continuación se aplicará un efecto a los botones del menú de navegación de la web que se ha ido desarrollando en el tema.

Editar El archivo web_tema_3_mod.html con el bloc de notas.

Añadir El siguiente código:

```
#navegacion { background-color:#999; height:40px; }
#navegacion li {
    float:left;
    width:170px;
    height:40px;
    background-color:#ccc;
    text-align:center;
    border-left:1px black solid;
    border-right:1px black solid;
    line-height:40px;
}
#navegacion ul li a {
    color:#ffffff;
    text-decoration: none;
    display:block;
}
#navegacion ul li a:hover {
    background-color:#666666;
    color:#D2CD2F;
}
```

La primera regla asigna el color blanco (#ffffff) a los enlaces dentro de una lista en el contenedor #navegación. Además quita cualquier tipo de decoración (subrayado) a dichos enlaces. La propiedad display:block hace que estos elementos de navegación llenen por completo todo el espacio de la barra de navegación.

La segunda regla se aplica sólo a los enlaces de la pseudoclase a: hover, es decir, cuando se pasa el ratón por encima del enlace. Se cambia el color de fondo y el color del texto para crear el efecto deseado.

Guardar El archivo web_tema_3_mod.html.

Comprobar El resultado en el navegador



Mi logo

Inicio

Sección 1

Sección 2

Sección 3

Sección 4

Bienvenido a mi web

ESTE ES UN EJEMPLO DE PÁGINA WEB CREADA CON HTML Y CSS

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi massa tellus, interdum ut tincidunt vitae, imperdiet auctor turpis. Cras ornare vel elit eget scelerisque. Cras vel sem consequat, viverra eros nec, semper quam.

Vestibulum nec placerat erat, ut aliquam augue. Donec eget mollis erat. Sed sodales ullamcorper mi. Curabitur porta dolor vitae semper interdum. Morbi volutpat turpis quis tortor aliquet adipiscing. Duis id magna facilisis elit venenatis condimentum. Donec vitae dolor quis lorem lobortis pharetra et vitae dui.

CONTENIDO LATERAL 1

NOTICIA 1

Martes, 25 de junio de 2013

Este es el texto de la noticia 1

NOTICIA 2

Lunes, 24 de junio de 2013

Este es el texto de la noticia 2

Copyright 2013 Mi web en HTML y CSS

Este sitio es un ejercicio sobre cómo utilizar HTML y CSS para la creación de una página web.



PRIORIDADES EN CSS

Al crear estilos en diferentes hojas, internas o externas, puede surgir la duda de qué ocurrirá si se definen estilos diferentes para un mismo elemento en un sitio o en otro.

Las hojas de estilo se denominan "en cascada" precisamente por la forma que tienen de definir su prioridad. Cuando un mismo estilo está definido en varios lugares, se sigue un orden de preferencia muy claro:

- Las propiedades definidas en hojas de estilo internas prevalecen sobre las externas.
- Si hay dos hojas de estilo externas aplicadas a un mismo documento prevalecerán las propiedades de la hoja cuyo enlace aparezca después en el código HTML del documento.

El uso del modificador !important en una determinada regla hace que ésta se salte la cadena de prioridades y que se aplique de forma prioritaria. Se emplea por ejemplo así:

```
p {margin: 10px !important; }
```

Es importante recalcar que esto sólo se aplica en caso de colisión entre propiedades, ya que si por ejemplo en la hoja externa se define el tipo de letra de los párrafos y en el documento interior se define el tamaño, se aplicarían las dos propiedades, ya que no hay colisión entre ellas.

LO QUE HEMOS APRENDIDO

- Un archivo de reset CSS sirve para eliminar los estilos por defecto de los elementos HTML para evitar que los distintos navegadores apliquen propiedades indeseadas.
- Las estructuras de una página web se clasifican en dos categorías: de ancho fijo y de ancho variable.
- Las estructuras de ancho fijo son más fáciles de programar pero no se adaptan a todas las resoluciones de pantalla.
- Las estructuras de ancho variable o Responsive, se verán correctamente en las distintas resoluciones de los dispositivos pero exigen una complejidad extra al diseñador.
- Las webs en HTML se estructuran en base a etiquetas <div> que serán los contenedores. Cada etiqueta <div> llevará un identificador para poder aplicar después estilos diferentes a cada contenedor.
- Los distintos márgenes pueden definirse de dos formas: estableciendo un margen interior (padding) al contenedor o mediante un margen exterior (margin) sobre los elementos que hay dentro del contenedor.
- Los enlaces y algunos otros elementos poseen pseudoclases que permiten aplicar diferentes estilos dependiendo de la acción que se realice sobre ellos. Esto permite crear efectos de interactividad.
- En caso de colisión entre propiedades definidas en distintas hojas de estilos, CSS tiene unas reglas de prioridad muy claras.

UNIDAD 2.1

HTML 5 PARTE 2^a

Etiquetas HTML5

- Repaso a las Características de HTML5
- Transformación de una Web a HTML5
- Compatibilidad con todos los Navegadores

REPASO A LAS CARACTERÍSTICAS DE HTML5

Las nuevas etiquetas semánticas, incorpora HTML5 para sustituir el excesivo uso de las etiquetas <div> y dar un sentido más lógico a la estructura de la página web.

También en HTML5 se incluyen las nuevas etiquetas de contenido multimedia, de formato de dibujo, mejora de formularios, etc.

Otra diferencia con respecto a HTML 4 es la simplificación de la sentencia doctype, que se reduce a <!DOCTYPE HTML>.

Por otro lado, HTML5 flexibiliza la sintaxis de la siguiente forma:

- En HTML5 no se diferencia entre mayúsculas y minúsculas, es decir, se pueden utilizar etiquetas en mayúsculas, en minúsculas e incluso mezclarlas ambas y la página sigue siendo válida.
- No se exigen las etiquetas de cierre de los elementos.
- El uso de comillas en los atributos es opcional.

Sin embargo, se considera una buena práctica seguir manteniendo la sintaxis estricta de HTML 4, y hacer caso omiso a estas reglas.

TRANSFORMACIÓN DE UNA WEB A HTML5

A continuación se tomará como base la web en HTML creada en el tema anterior para transformarla en una web programada con HTML5. De esta forma se recalcarán las diferencias entre un lenguaje y otro.

Copiar Los archivos web_tema_3_final.html, reset.css, estilos.css y la carpeta images a una nueva carpeta.

Renombrar El archivo web_tema_3_final.html como web_tema_4.html dentro de esta nueva carpeta.



Editar El archivo web_tema_4.html con el bloc de notas.

Modificar El doctype en la primera línea para indicar que ahora es HTML5

```
web_tema_4.html: Bloc de notas
Archivo Edición Formato Ver Ayuda
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1" />
```

The screenshot shows a Windows Notepad window with the title 'web_tema_4.html: Bloc de notas'. The window contains the following HTML code:

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1" />
```

The first line, '`<!DOCTYPE html>`', is highlighted with a red rectangular selection bar.

Modificar El título de la página



```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <meta http-equiv="content-type" content="text/html; charset=iso-8859-1" />
    <title>web Tema 4</title>
    <link href="reset.css" rel="stylesheet" type="text/css" />
    <link href="estilos.css" rel="stylesheet" type="text/css" />
```

SUSTITUCIÓN DE ETIQUETAS <DIV> POR ETIQUETAS SEMÁNTICAS HTML5

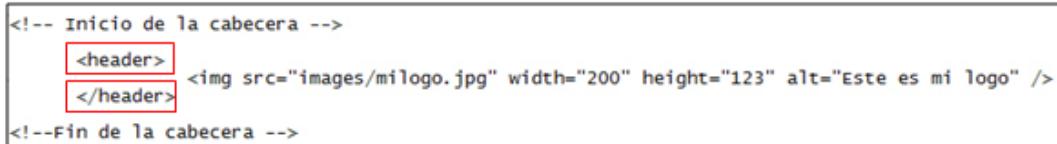
Como ya se ha comentado, HTML5 incorpora etiquetas específicas para ciertos elementos de la web, para que el contenido quede mejor estructurado semánticamente.

Cabecera

Editar El archivo web_tema_4.html con el bloc de notas.

Buscar La etiqueta <div id="cabecera"> y sustituirla por <header>.

Sustituir La etiqueta </div> de cierre de la cabecera por </header>.



```
<!-- Inicio de la cabecera -->
<header> 
</header>
<!--Fin de la cabecera -->
```

Guardar El archivo

Editar El archivo estilos.css con el bloc de notas

Sustituir El selector #cabecera por header.

```
#cabecera {
}
```

```
header {
}
```

Guardar El archivo.

Navegación, Lateral, Principal y Pie de Página

Siguiendo el mismo procedimiento descrito para la cabecera se deben sustituir los siguientes de elementos HTML y CSS como se indica en la siguiente tabla:

Etiqueta HTML	Sustituir por	Selector CSS	Sustituir por
<div id="navegación">...</div>	<nav>...</nav>	#navegación	nav
<div id="lateral">...</div>	<aside>...</aside>	#lateral	aside
<div id="principal">...</div>	<section>...</section>	#principal	section
<div id="piedepagina">...</div>	<footer>...</footer>	#piedepagina	footer

Una vez guardados los cambios en los archivos web_tema_4.html y estilos.css, se comprueba el resultado en el navegador, ejecutando el archivo web_tema_4.html. La apariencia debe ser idéntica a la web diseñada en el tema anterior.

UTILIZACIÓN DEL ELEMENTO ARTICLE

La etiqueta <article> se utiliza usada para representar un contenido específico de la web. Puede representar un artículo, una entrada en un blog, una noticia o simplemente un contenido estático de una web como por ejemplo el "quiénes somos".

Por lo tanto, este tipo de contenido tiene un alto valor semántico ya que aporta información relevante dentro del documento. Este elemento debería tener principalmente un título y un cuerpo con el detalle, pero también podría incorporar una imagen, la fecha de publicación, el autor o cualquier otra información adicional.

En el ejemplo se observa que claramente el contenido de la columna principal sería adecuado para identificarlo como article.



Para ello:

Editar El archivo web_tema_4.html con el bloc de notas.

Insertar Las etiquetas `<article>` y `</article>` rodeando el contenido del artículo.

```

<!-- Inicio de la columna principal-->
<section>
    <h1>Bienvenido a mi web </h1>
    <article>
        <h2>Este es un ejemplo de página web creada con HTML y CSS </h2>
        <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi massa tellus, interdum ut tincidunt vitae, imperdiet auctor turpis. Cras ornare vel elit eget scelerisque. Cras vel sem consequat, viverra eros nec, semper quam. </p>
        <p>Vestibulum nec placerat erat, ut aliquam augue. Donec eget mollis erat. Sed sodales ullamcorper mi. Curabitur porta dolor vitae semper interdum. Morbi volutpat turpis quis tortor aliquet adipiscing. Duis id magna facilisis elit venenatis condimentum. Donec vitae dolor quis lorem lobortis pharetra et vitae dui.</p>
    </article>
</section>
<!-- Fin de la columna principal-->

```

INCORPORACIÓN DE UNA IMAGEN EN EL ARTÍCULO

Los elementos `<figure>` y `<figcaption>`, junto a la ya utilizada `` permiten identificar imágenes y sus rótulos dentro del contenido.

Para añadir una imagen dentro del artículo anterior:

Editar El archivo web_tema_4.html con el bloc de notas

Incluir El siguiente código dentro del artículo

```
<article>
  <h2>Este es un ejemplo de página web creada con HTML y CSS </h2>
  <figure>
    
    <figcaption> Pie de foto
    </figcaption>
  </figure>

  <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi massa tellus, interdum ut tincidunt vitae, imperdiet auctor turpis. Cras ornare vel elit eget scelerisque. Cras vel sem consequat, viverra eros nec, semper quam. </p>
  <p>Vestibulum nec placerat erat, ut aliquam augue. Donec eget mollis erat, sed sodales ullamcorper mi. Curabitur porta dolor vitae semper interdum. Morbi volutpat turpis quis tortor aliquet adipiscing. Duis id magna facilisis elit venenatis condimentum. Donec vitae dolor quis lorem lobortis pharetra et vitae dui.</p>
</article>
```

De esta forma se ha añadido un elemento `<figure>` que incluye la imagen `` y el pie de foto con `<figcaption>`.

Lógicamente ha imagen gatito2.jpg debe encontrarse dentro de la carpeta images del proyecto web.

Editar El archivo estilos.css con el bloc de notas

Incluir Las siguientes reglas de estilo para figure y figcaption al final de la hoja

```
figure {
  float:right;
  padding:4px;
  margin:6px;
}

figcaption {
  text-align:center;
  font:italic 0.7em Georgia, "Times New Roman", Times, serif;
}
```

Con estas reglas se coloca la imagen de forma flotante alineada a la derecha del texto, y también se le asignan unos márgenes. Además se establece un tipo de letra al formato del pie de página, que se alinea al centro de la imagen.

Guardar Los archivos web_tema_4.html y estilos.css.

Comprobar El resultado en el navegador


Mi logo

Inicio | Sección 1 | Sección 2 | Sección 3 | Sección 4

Bienvenido a mi web

ESTE ES UN EJEMPLO DE PÁGINA WEB CREADA CON HTML Y CSS

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi massa tellus, interdum ut tincidunt vitae, imperdiet auctor turpis. Cras ornare vel elit eget scelerisque. Cras vel sem consequat, viverra eros nec, semper quam.

Vestibulum nec placerat erat, ut aliquam augue. Donec eget mollis erat. Sed sodales ullamcorper mi. Curabitur porta dolor vitae semper interdum. Morbi volutpat turpis quis tortor aliquet adipiscing. Duis id magna facilisis elit venenatis condimentum. Donec vitae dolor quis lorem lobortis pharetra et vitae dui.


Pie de foto

CONTENIDO LATERAL 1

NOTICIA 1
Martes, 25 de junio de 2013
Este es el texto de la noticia 1

NOTICIA 2
Lunes, 24 de junio de 2013
Este es el texto de la noticia 2

Copyright 2013 Mi web en HTML y CSS
Este sitio es un ejercicio sobre cómo utilizar HTML y CSS para la creación de una página web.



Cabe resaltar que conseguir este efecto sin las etiquetas figure y figcaption hubiera resultado mucho más complicado.

COMPATIBILIDAD CON TODOS LOS NAVEGADORES

En la actualidad, aunque cada vez son menos, muchos usuarios utilizan navegadores de internet con versiones antiguas que no son compatibles con HTML5.

Cuando un navegador encuentra un elemento desconocido y este elemento tiene estilos CSS asociados resulta un problema. Por ejemplo muchos navegadores antiguos tratan al elemento desconocido <header> como un elemento de texto normal o simplemente lo ignoran.

Existen varias formas para hacer que los navegadores más antiguos apliquen adecuadamente los estilos a las nuevas etiquetas de HTML5 y CSS3. Una manera de conseguirlo es utilizar la librería Javascript Modernizr. Esta librería incorpora los nuevos elementos HTML5 a los navegadores antiguos. El código Javascript de esta librería está pensado para detectar funcionalidades dentro de cualquier navegador y añadirles el soporte necesario para poder utilizar los nuevos objetos y estilos si no están soportados.

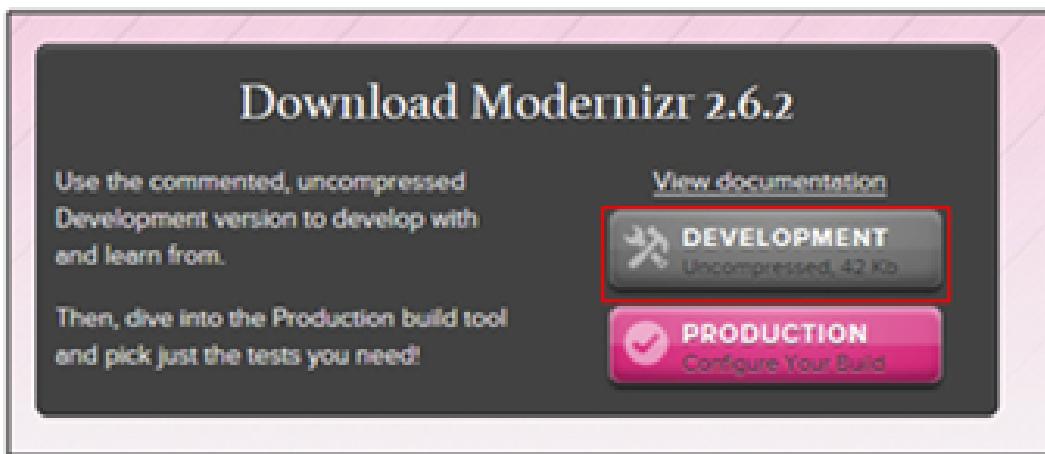
DESCARGA DE MODERNIZR

El primer paso consistirá en descargar el archivo con el código fuente de Modernizr de la web <http://modernizr.com>.

The screenshot shows the official Modernizr website. At the top, there's a navigation bar with links for DOWNLOAD, DOCUMENTATION, RESOURCES, and NEWS. Below the navigation, a quote from Bruce Bowman says: "An indispensable tool." — Bruce Bowman, sr. product manager, Edge Tools & Services. The main content area features a large button labeled "Download Modernizr 2.6.2". To the left of this button, there's a section about what Modernizr is and why it's useful. To the right, there are two buttons: one for "DEVELOPMENT" (Uncompressed, 42 Kb) and one for "PRODUCTION" (Configure Your Build). At the bottom, there's a section titled "Get started with Modernizr" with some introductory text.

Se ofrecen dos modalidades de descarga:

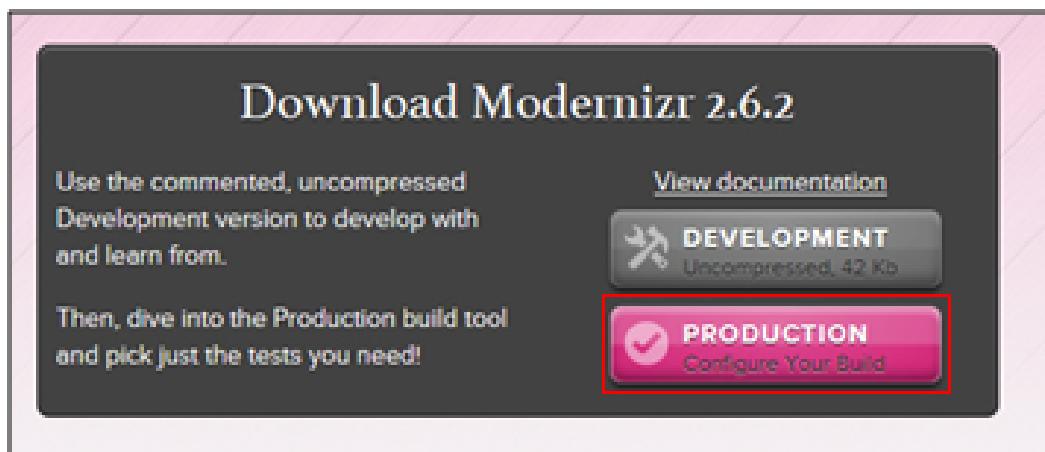
- Desarrollo (Development): Es un script completo, con todas las funcionalidades básicas de Modernizr, sin comprimir y con comentarios. Se puede usar este archivo durante la fase de desarrollo del proyecto, cuando aún no se sabe qué funcionalidades se van a utilizar. Para descargar este paquete basta con pulsar el botón Development en la página principal de Modernizr y se descargará el archivo modernizr-latest.js en el ordenador.



- Producción (Production): Permite seleccionar de una lista sólamente las funcionalidades que se quieran incluir porque sean las que se necesiten para el proyecto web que se ha desarrollado en concreto. El peso de las librerías no es muy grande, pero es recomendable optimizar la descarga para incluir solo aquellos módulos que se vayan a utilizar.

Para elegir esta segunda opción:

Pulsar Production en la página principal de la web de Modernizr.

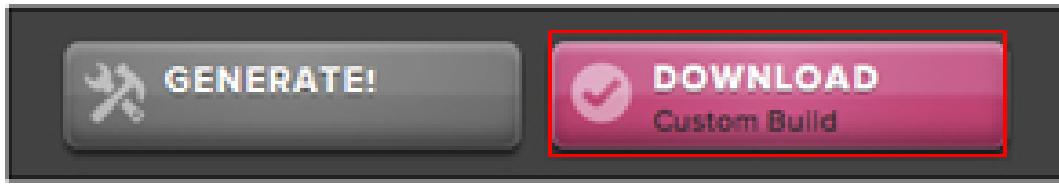


Seleccionar Las opciones que se deseen

Pulsar Generate

The screenshot shows the Modernizr build tool interface. It has three main sections: "CSS3", "HTML5", and "Misc.". Each section has a "TOGGLE" button. Under "CSS3", all items are checked. Under "HTML5", most items are checked, except for "applicationCache" which is unchecked. Under "Misc.", "Geolocation API" is checked, while others like "Inline SVG" and "WebGL" are unchecked. In the bottom left, there's an "Extensibility" section with a "GENERATE!" button. In the bottom right, there's a "Non-core detects" section. A red box highlights the "GENERATE!" button.

Hacer clic en Download



Guardar El archivo con extensión .js en el ordenador



INSTALACIÓN DE MODERNIZR EN LA PÁGINA

Una vez que se ha descargado el archivo .js con la librería, hay que incluirla en el código HTML de la página. En el ejemplo se ha descargado la opción de Desarrollo (Development), se trata por tanto del archivo modernizr-latest.js.

Copiar El archivo modernizr-latest.js a la carpeta donde se encuentra el archivo web_tema_4.html

Editar El archivo web_tema_4.html con el bloc de notas

Añadir La siguiente línea de código dentro de la parte <head> de la página

```
<head>
  <meta http-equiv="content-type" content="text/html; charset=iso-8859-1" />
  <title>web Tema 4</title>
  <link href="reset.css" rel="stylesheet" type="text/css" />
  <link href="estilos.css" rel="stylesheet" type="text/css" />
  <script src="modernizr-latest.js"></script>
<style type="text/css">
</style>
</head>
```

Guardar Los cambios en el archivo

POLYFILLS

Un Polyfill o Polyfiller es una librería o plugin para ampliar las funcionalidades de navegadores antiguos que no soportan funcionalidades modernas, como son las APIs de HTML5. Los Polyfill no forman parte de Modernizr específicamente, pero éste sí proporciona elementos para cargarlos en el navegador cuando no se tenga soporte nativo.

En el paquete básico de Modernizr no está disponible, pero sí entre los "Extra" que se pueden seleccionar en la página de Production, se trata del método Modernizr.load() que sirve para cargar Pollyfills.

The screenshot shows the 'Download Modernizr 2.6.2' page. It features three main sections: 'CSS3', 'HTML5', and 'Misc.'. Below these are two buttons: 'Extensibility' and 'Non-core detects'. At the bottom is a 'GENERATE!' button with a gear icon. A red box highlights the 'Modernizr.load()' checkbox in the 'Extra' section.

Use the [Development version](#) to develop with and learn from. Then, when you're ready for production, use the build tool below to pick only the tests you need.

CSS3 [TOGGLE]

- @font-face
- background-size
- border-image
- border-radius
- box-shadow
- Flexible Box Model (flexbox)
- Flexbox Legacy
- font
- multiple backgrounds
- opacity
- rgba()
- text-shadow
- CSS Animations
- CSS Columns
- CSS Generated Content (content attr)
- CSS Gradients
- CSS Reflections
- CSS 2D Transforms
- CSS 3D Transforms
- CSS Transitions

HTML5 [TOGGLE]

- applicationCache
- Canvas
- Canvas Text
- Drag & Drop
- Hashchange
- History (pushState)
- HTML5 Audio
- HTML5 Video
- IndexedDB
- Input Attributes

Note: does not add classes

- Input Types
- Note: does not add classes
- localStorage
- postMessage
- sessionStorage
- Web Sockets
- Web SQL Database
- Web Workers

Misc. [TOGGLE]

- Geolocation API
- Inert SVG
- SMIL
- SVG
- SVG clip paths
- Touch Events
- WebGL

Extra

- html5shiv v3.6
- html5shiv v3.6 w/ printshiv
- Modernizr.load ()

! Importante !

- Media Queries
- Add CSS Classes

(classNames prefix:)

Extensibility

Non-core detects

GENERATE!

Desde la propia documentación de Modernizr aconsejan utilizar los Pollyfills con cuidado, porque puede bajar el rendimiento de los navegadores, no obstante, es una buena opción para poder innovar en el desarrollo y mantener soporte hacia atrás con navegadores antiguos.

LO QUE HEMOS APRENDIDO

- HTML5 flexibiliza la sintaxis HTML en varios aspectos, como el uso de mayúsculas y minúsculas, aunque es una buena práctica seguir utilizando las reglas de HTML 4.
- En HTML5 se sustituyen muchas de las etiquetas `<div>` por etiquetas de contenido semántico como `<header>`, `<section>`, `<nav>`, etc., lo cual proporciona un contenido más estructurado al diseñador.
- La etiqueta `<article>` se utiliza para delimitar contenidos específicos como artículos, entradas en un blog, noticias o contenido estático.
- Las etiquetas `<figure>` y `<figcaption>` facilitan la incorporación de imágenes y pies de foto a los contenidos.
- A la hora de diseñar una web con HTML5 es importante tener en cuenta la compatibilidad con los navegadores que no soportan este lenguaje.
- Para que una web sea compatible con todos los navegadores, incluso con los que no soportan HTML5 se utilizan librerías como Modernizr y los Polyfills.

UNIDAD 2.2

HTML 5 PARTE 2^a

El Canvas de HTML5

- Introducción a Canvas
- Fundamentos de Javascript
- Utilización de Canvas

INTRODUCCIÓN A CANVAS

Como ya se comentó brevemente en temas anteriores, HTML5 incorpora un entorno para crear imágenes y gráficos de forma dinámica llamado Canvas (del inglés, lienzo).

Para añadir un lienzo de tipo canvas dentro de un documento HTML5, se utilizan las etiquetas `<canvas>` y `</canvas>`.

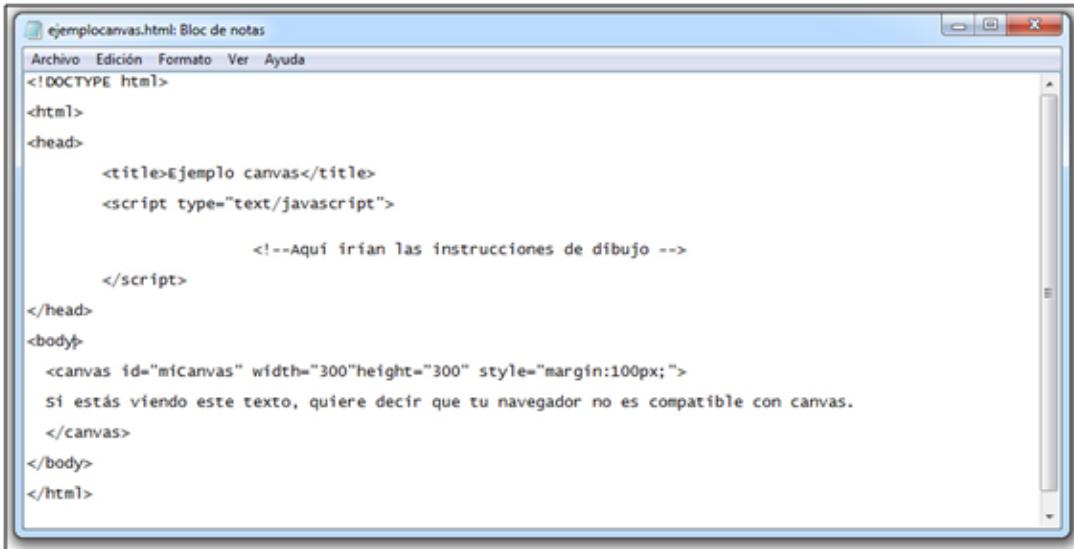
Entre los atributos que se deben definir dentro de `<canvas>` se encuentran:

- Identificador: Mediante `id="nombre_del_canvas"` se asigna un nombre único al canvas para poder referirse a él.
- Alto: La propiedad `height` indica la altura que tendrá el lienzo.
- Ancho: La propiedad `width` indica el ancho que tendrá el lienzo.

También se pueden definir estilos para el lienzo como márgenes, color de fondo, etc.

Todo lo que se escriba entre las etiquetas `<canvas>` y `</canvas>` sólo será mostrado si el navegador no es compatible con este formato de dibujo, por tanto lo normal es incluir algún tipo de mensaje de advertencia para el usuario.

El código que implementa lo que será dibujado en el canvas se realiza a través de lenguaje Javascript, que normalmente se incrusta en el head del documento HTML mediante las etiquetas `<script>` y `</script>`.



```
ejemplocanvas.html: Bloc de notas
Archivo Edición Formato Ver Ayuda
<!DOCTYPE html>
<html>
<head>
    <title>Ejemplo canvas</title>
    <script type="text/javascript">

        <!--Aqui irian las instrucciones de dibujo -->
    </script>
</head>
<body>
    <canvas id="micanvas" width="300" height="300" style="margin:100px;">
        Si estás viendo este texto, quiere decir que tu navegador no es compatible con canvas.
    </canvas>
</body>
</html>
```

Por tanto, para poder trabajar con esta herramienta de HTML5 hay que tener ciertos conocimientos de lenguaje Javascript.

FUNDAMENTOS DE JAVASCRIPT

Javascript es un lenguaje de programación que permite crear pequeños programas que, incrustados dentro de una página web, realizan funciones dinámicas y de interacción con el usuario, cosa que no permitía HTML.

Para introducir código JavaScript en una página HTML, debe incluirse entre las siguientes etiquetas:

```
<script language="JavaScript" type="text/javascript">...</script>
```

También puede indicarse que el código Javascript está ubicado en algún fichero externo (con extensión .js), en este caso se usaría:

```
<script type="text/javascript" src="mifichero.js"></script>
```

Estas etiquetas normalmente se insertan dentro de `<head>` y `</head>` del documento HTML, aunque hay excepciones.

ELEMENTOS DE JAVASCRIPT

- **Variables:** Una variable es un elemento que puede almacenar distintos valores. Se definen mediante la palabra reservada `var`. Por ejemplo: `var n=145`, crea una variable llamada "n" y le asigna un valor de 145. Las variables pueden ser de tipo numérico, cadenas de caracteres, valores booleanos (sólo pueden tomar dos valores: verdadero o falso), o bien de tipo `null`, que no contienen nada.
- **Condiciones:** Mediante las palabras reservadas `if` y `else`, se pueden definir condiciones, es decir, el código entre las llaves que lo componen sólo se ejecutará si se cumple una determinada condición. En el siguiente ejemplo, si el contenido de la variable `n` es menor que 10 se ejecutará el código 1 y en otro caso (si `n` es mayor que 10) se ejecutará el código 2:

```
if (n<10){ codigo 1 }
else{ codigo 2}
```

- Bucles: El bucle FOR se utiliza para repetir una o más instrucciones un determinado número de veces. Por ejemplo, suponiendo que la función escribir, pinta en pantalla el contenido de la variable entre paréntesis, el siguiente bucle escribirá 012345678910:

```
for (i=0;i<=10;i++) {
    escribir(i);
}
```

- Arrays: Es un tipo de variable que incluye una cadena de elementos del mismo tipo, se definen con sentencias de este tipo:

```
var miArray = new Array("casa", "coche", "perro");
```

Para acceder a un elemento concreto del array se utilizan corchetes con el número del índice del array, así por ejemplo miArray[2] contiene "coche".

- Comentarios: Como en otros lenguajes, los comentarios son líneas que no se ejecutan, sólo sirven para incluir anotaciones dentro del código. Para comentar una sola línea en Javascript se utiliza // al principio de esa línea. Para comentar varias líneas se delimitan entre /* y */.
- Expresiones: Son operaciones que se realizan con las variables.
Ejemplo:

```
x = y +10;
```

Tras esta operación, la variable x contendrá el valor de la variable y más 10.

- Funciones: Son conjuntos de operaciones e instrucciones que se agrupan para formar una operación más compleja. Pueden ser definidas por el programador o bien se pueden utilizar funciones que ya están predefinidas en la librería Javascript. Para definir una función se utiliza la palabra reservada function. A las funciones se les pueden pasar argumentos entre () y devolver un resultado con la palabra return. Ejemplo:

Definición de una función que calcula el volumen de un cubo:

```
function volumencubo(numero) {  
    var volumen= numero * numero * numero;  
    return volumen;  
}
```

Para llamar a dicha función se utilizaría por ejemplo:

```
var lado=3;  
volumen_de_mi_cubo=cubo(lado);
```

- Eventos: Ciertos elementos de HTML poseen la capacidad de generar eventos. Por ejemplo, un enlace HTML puede generar el evento onClick, que es cuando se pulsa dicho enlace. Se pueden hacer llamadas a funciones de Javascript cuando se produzca uno de estos eventos, lo cual permite crear interactividad con el usuario. Ejemplo:

```
<a href="enlace.html" onClick="miFuncion( )"> Pulse aquí </a>
```

OBJETOS

Los objetos son variables complejas que contienen distintos tipos de datos de forma estructurada. Son la base de los lenguajes que, como Javascript, se llaman orientados a objetos.

Para crear un objeto, lo primero es definir su estructura o clase.
Ejemplo:

```
function persona(nombre, edad, telefono) {  
    this.nombre = nombre;  
    this.edad = edad;  
    this.telefono = telefono;  
}
```

La partícula this se refiere al objeto en el que se utiliza.

Una vez definida la clase, se pueden crear variables (instanciar objetos) de esa clase de la siguiente manera:

```
persona_1 = new persona("Francisco", 26, 630555555);
```

Dentro de la definición del objeto, pueden incluirse funciones que accedan a sus propiedades, que se llaman métodos.

El Objeto Document

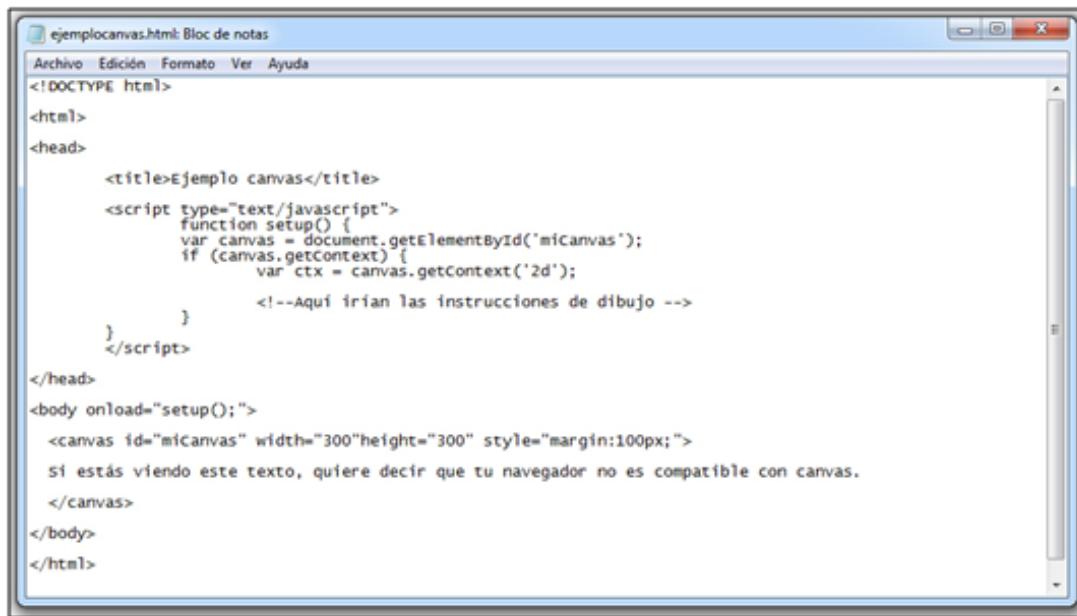
Es un tipo de objeto que ya viene predefinido en Javascript y que hace referencia al documento HTML que se está ejecutando. También tiene ya implementados varios métodos que resultan muy útiles para interaccionar con la página web. Es importante conocer algunos de estos métodos como:

- `document.write(...)`: Escribe lo que se encuentre dentro del paréntesis en la pantalla, puede ser una cadena de caracteres, una variable, etc.
- `document.getElementById('identificador')`: Permite obtener la referencia a un elemento HTML de la página mediante la propiedad id de dicho elemento. Una vez obtenida la referencia al elemento, se puede acceder a todas sus propiedades para mostrarlas, modificarlas, etc.

UTILIZACIÓN DE CANVAS

Para comenzar a trabajar con el elemento Canvas y Javascript lo primero es definir una función de establecimiento o setup, que encontrará el elemento canvas por su identificador y declarará una variable con la referencia al contexto de dibujo del canvas. Una vez que está disponible esa referencia, se podrá dibujar libremente sobre esa superficie.

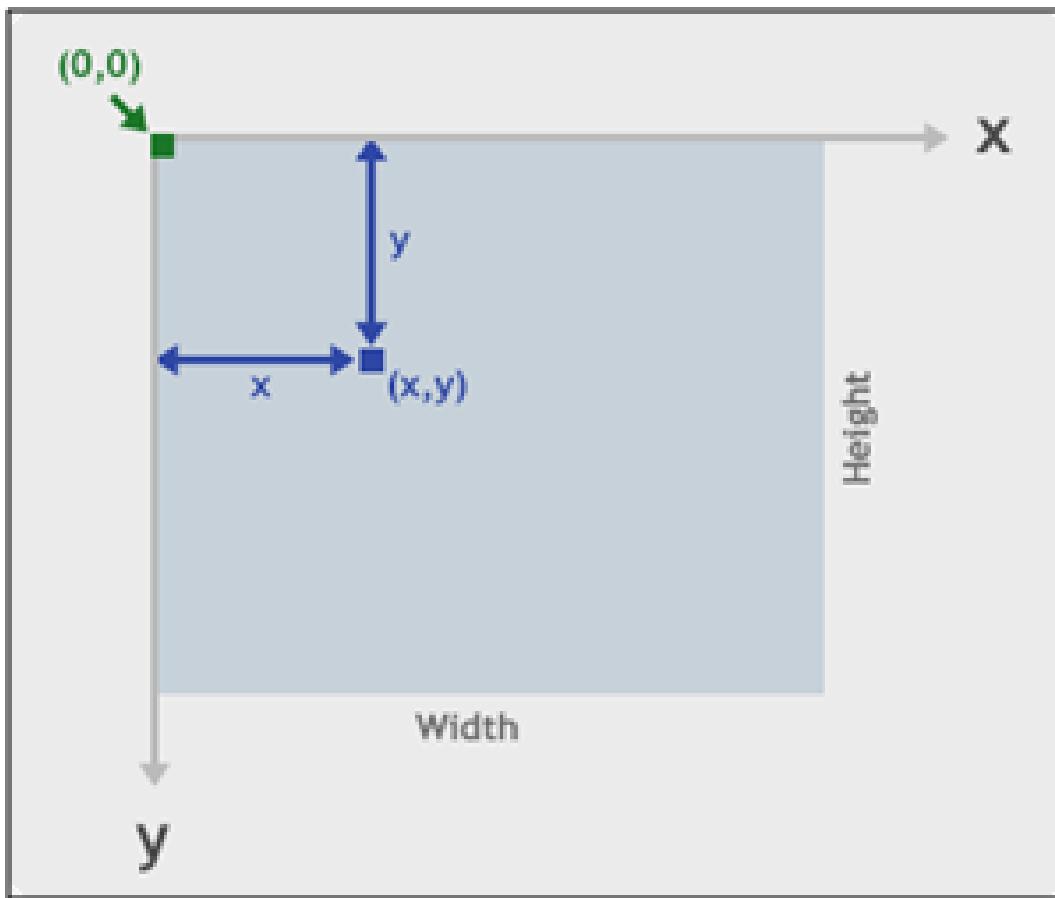
Este código siempre será el mismo por lo que se puede utilizar a modo de plantilla:



```
ejemplocanvas.html: Bloc de notas
Archivo Edición Formato Ver Ayuda
<!DOCTYPE html>
<html>
<head>
    <title>Ejemplo canvas</title>
    <script type="text/javascript">
        function setup() {
            var canvas = document.getElementById('micanvas');
            if (canvas.getContext) {
                var ctx = canvas.getContext('2d');
            }
        }
    </script>
</head>
<body onload="setup();">
    <canvas id="micanvas" width="300" height="300" style="margin:100px;">
        Si estás viendo este texto, quiere decir que tu navegador no es compatible con canvas.
    </canvas>
</body>
</html>
```

REFERENCIA A LAS POSICIONES

Para dibujar elementos en el canvas, existen una serie de funciones que se irán describiendo a continuación. Estas funciones tendrán en muchos casos unos parámetros o propiedades que establecerán donde irá colocado el elemento dentro del lienzo. El siguiente dibujo muestra cómo se definen estos parámetros:



Es decir, los parámetros x e y indican la distancia al punto de referencia $(0,0)$ que es la esquina superior izquierda del canvas. Los parámetros $width$ y $height$ indicarán el ancho y el alto del elemento respectivamente.

DIBUJO DE RECTÁNGULOS

Métodos relacionados con el dibujo de rectángulos:

- `fillRect(x, y, width, height):`

Dibuja un rectángulo en la posición determinada por x e y , con el ancho y alto indicados por $width$ y $height$. Este tipo de rectángulo irá relleno por el color que tenga la variable `fillStyle` en ese momento.

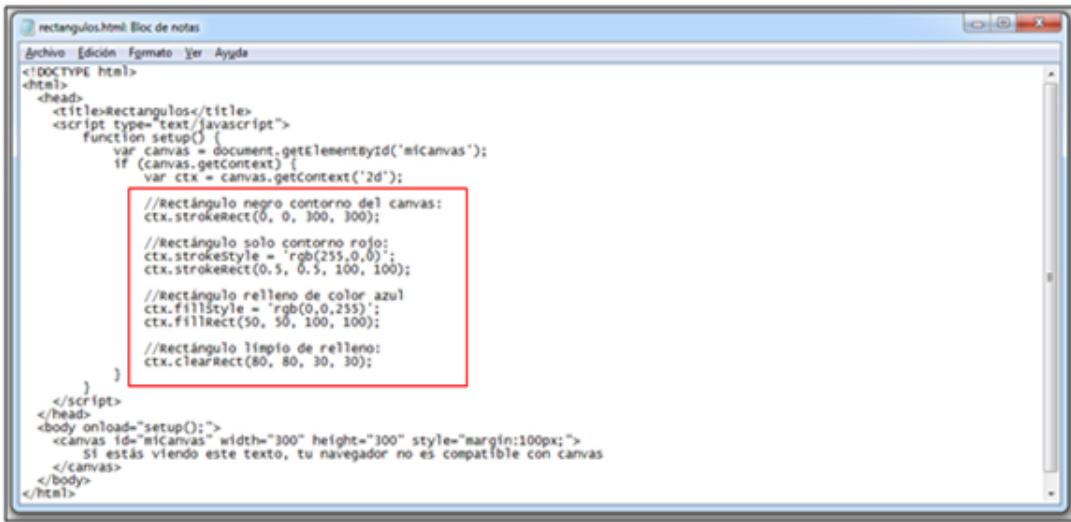
- `strokeRect(x, y, width, height):`

Genera un rectángulo pero en lugar de rellenarlo, dibuja el contorno basándose en el estilo definido en ese momento para la variable `strokeStyle`.

- `clearRect(x, y, width, height)`:

Crea un rectángulo pero que borra todos los pixels del canvas que quedan dentro del área del rectángulo definido.

El siguiente ejemplo toma como base la plantilla inicial y dibuja unos rectángulos en el canvas:



```
rectangulos.html - Bloc de notas
Archivo Edición Formato Ver Ayuda
<!DOCTYPE html>
<html>
<head>
<title>Rectángulos</title>
<script type="text/javascript">
function setup() {
    var canvas = document.getElementById('micanvas');
    if (canvas.getContext) {
        var ctx = canvas.getContext('2d');

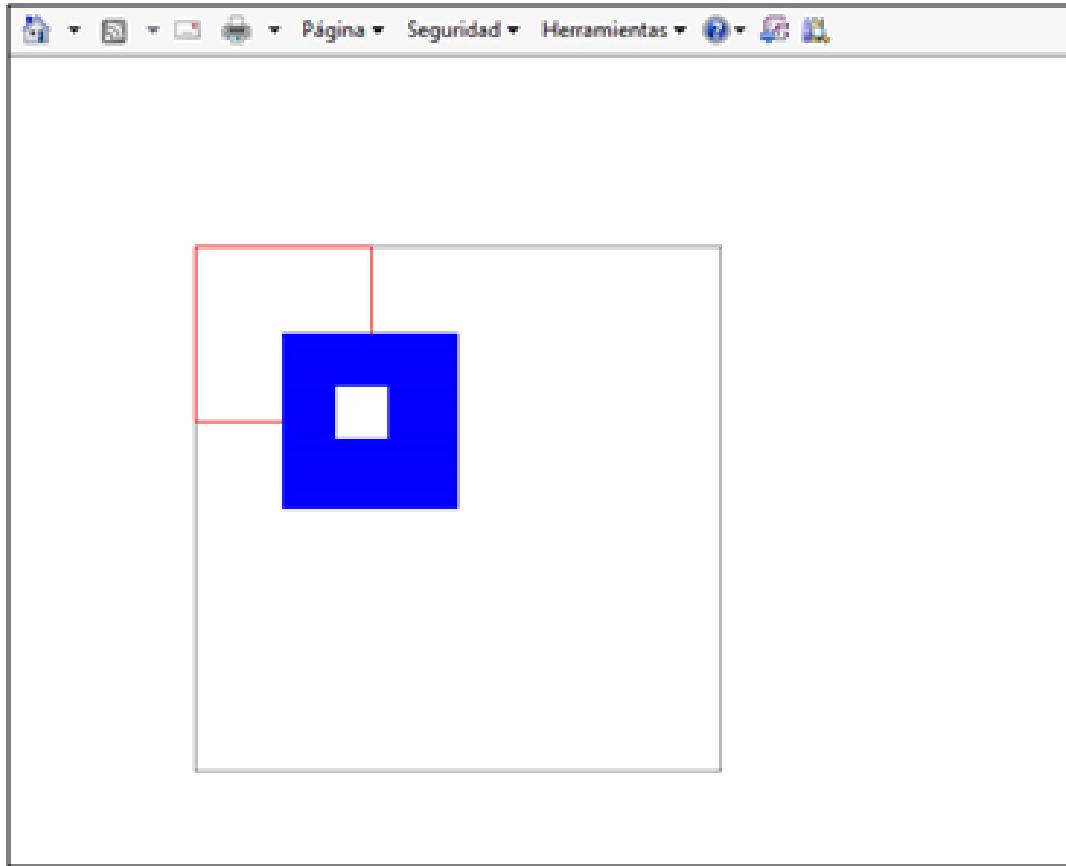
        //Rectángulo negro contorno del canvas:
        ctx.strokeRect(0, 0, 300, 300);

        //Rectángulo solo contorno rojo:
        ctx.strokeStyle = 'rgb(255,0,0)';
        ctx.strokeRect(0.5, 0.5, 100, 100);

        //Rectángulo lleno de color azul
        ctx.fillStyle = 'rgb(0,0,255)';
        ctx.fillRect(50, 50, 100, 100);

        //Rectángulo limpio de relleno:
        ctx.clearRect(80, 80, 30, 30);
    }
}
</script>
</head>
<body onload="setup()">
<canvas id="micanvas" width="300" height="300" style="margin:100px;">
    Si estás viendo este texto, tu navegador no es compatible con canvas
</canvas>
</body>
</html>
```

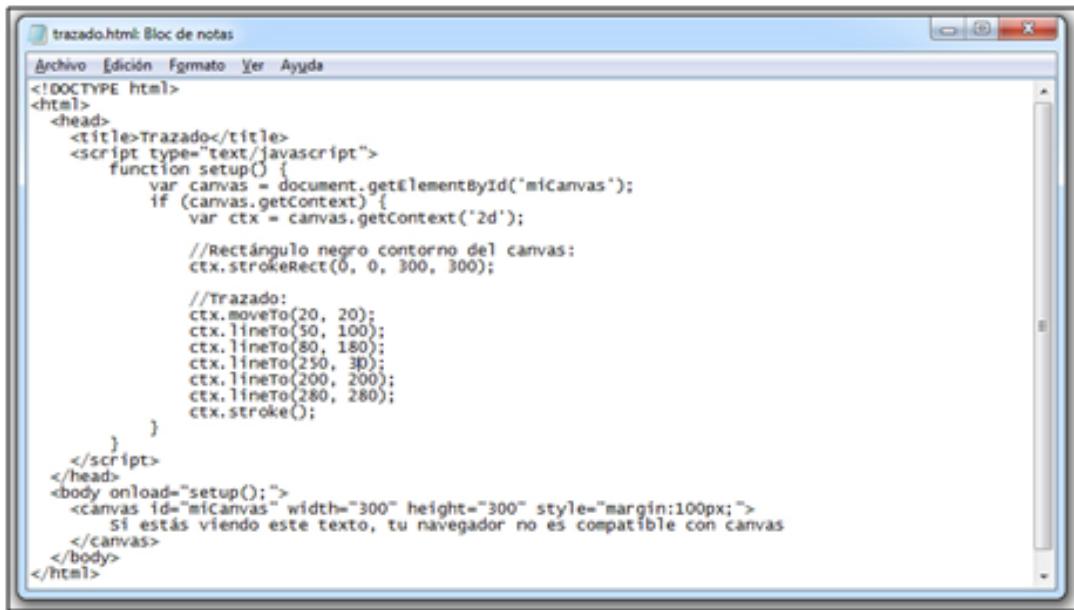
El resultado en el navegador es:



DIBUJO DE TRAZADOS RECTILÍNEOS

Para dibujar un trazado compuesto por varias líneas, se define el primer punto mediante la función `moveTo(x,y)`, a continuación se van definiendo los puntos del trazado con la función `lineTo(x,y)` y finalmente se cierra con una llamada a la función `stroke()` que dibuja el trazado.

Ejemplo:

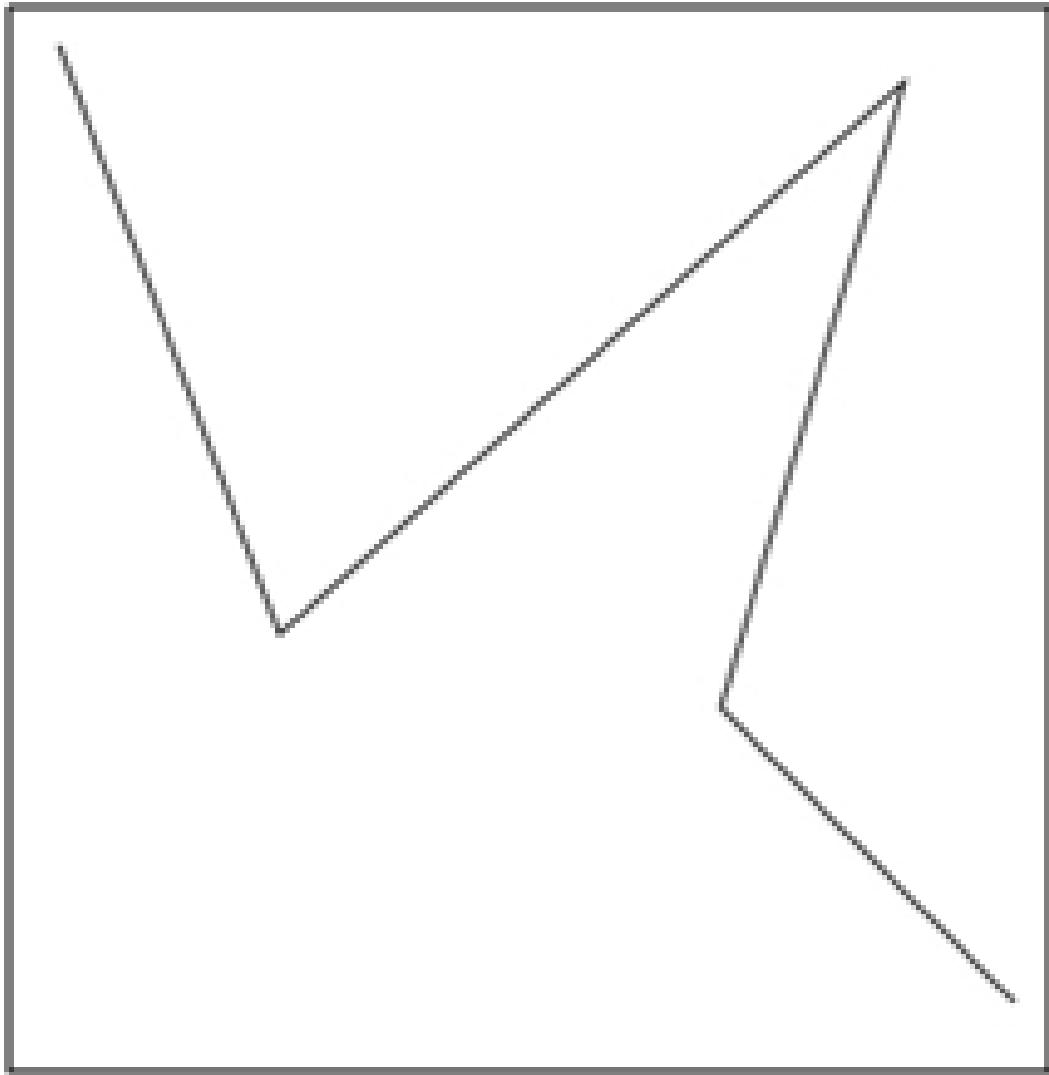


```
<!DOCTYPE html>
<html>
<head>
<title>Trazado</title>
<script type="text/javascript">
function setup() {
    var canvas = document.getElementById('micanvas');
    if (canvas.getContext) {
        var ctx = canvas.getContext('2d');

        //Rectángulo negro contorno del canvas:
        ctx.strokeRect(0, 0, 300, 300);

        //Trazado:
        ctx.moveTo(20, 20);
        ctx.lineTo(50, 100);
        ctx.lineTo(80, 180);
        ctx.lineTo(250, 30);
        ctx.lineTo(200, 200);
        ctx.lineTo(280, 280);
        ctx.stroke();
    }
}
</script>
</head>
<body onload="setup();">
<canvas id="micanvas" width="300" height="300" style="margin:100px;">
    Si estás viendo este texto, tu navegador no es compatible con canvas
</canvas>
</body>
</html>
```

Dibujará:



Para dibujar varios trazados diferentes en un mismo canvas se utilizan las funciones `beginPath()` y `closePath()` para indicar el principio y el fin del trazado.

Si se utiliza la función `fill()` en lugar de `stroke()`, el trazado se rellenará con el color que contenga la variable `fillStyle` en ese momento.

DIBUJO DE CURVAS

Circunferencias y Arcos

El método `arc` permite dibujar caminos con arcos, es decir, segmentos de circunferencias, de la siguiente forma:

`arc(x, y, radio, angulo_inicial, angulo_final, sentido_antihorario)`

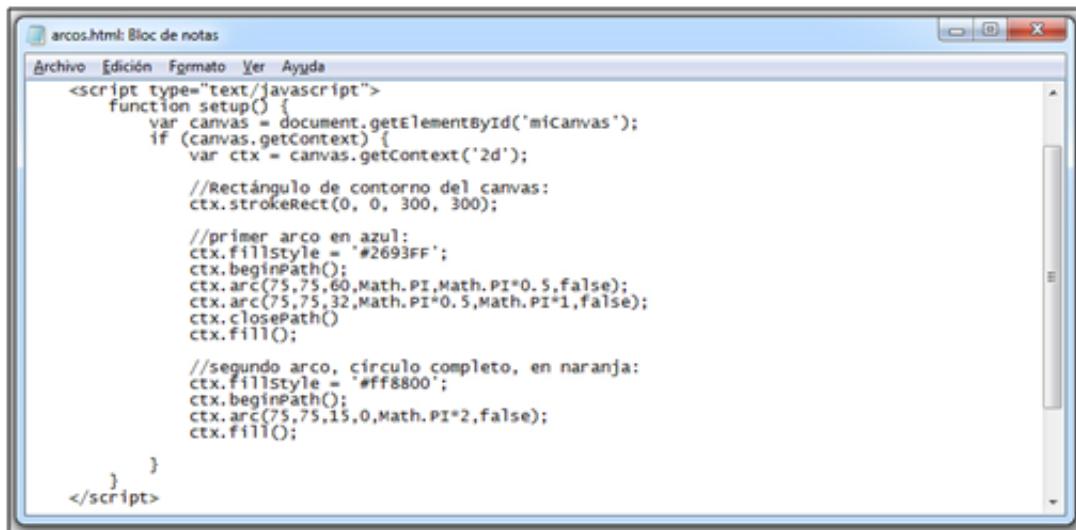
Los parámetros x, y corresponden con las coordenadas del centro del arco.

El parámetro radio es el número de píxeles que tiene el arco como radio.

Los parámetros angulo_inicial y angulo_final son los ángulos donde comienza y acaba el arco en radianes. Conviene recordar que 180° son π radianes. En Javascript se indica con la constante Math.PI.

El parámetro sentido_antihorario es un parámetro booleano, que puede valer TRUE (significa que el trazo va desde el ángulo de inicio al de fin en el sentido contrario al de las agujas del reloj) o FALSE (indica que ese camino es en dirección al de las agujas del reloj).

Ejemplo:



```
arcos.html: Bloc de notas
Archivo Edición Fgrmato Ver Ayuda
<script type="text/javascript">
    function setup() {
        var canvas = document.getElementById('micanvas');
        if (canvas.getContext) {
            var ctx = canvas.getContext('2d');

            //Rectángulo de contorno del canvas:
            ctx.strokeRect(0, 0, 300, 300);

            //primer arco en azul:
            ctx.fillStyle = '#2693FF';
            ctx.beginPath();
            ctx.arc(75,75,60,Math.PI,Math.PI*0.5,false);
            ctx.arc(75,75,32,Math.PI*0.5,Math.PI*1,false);
            ctx.closePath();
            ctx.fill();

            //segundo arco, círculo completo, en naranja:
            ctx.fillStyle = '#ff8800';
            ctx.beginPath();
            ctx.arc(75,75,15,0,Math.PI*2,false);
            ctx.fill();
        }
    }
</script>
```

Resultado:

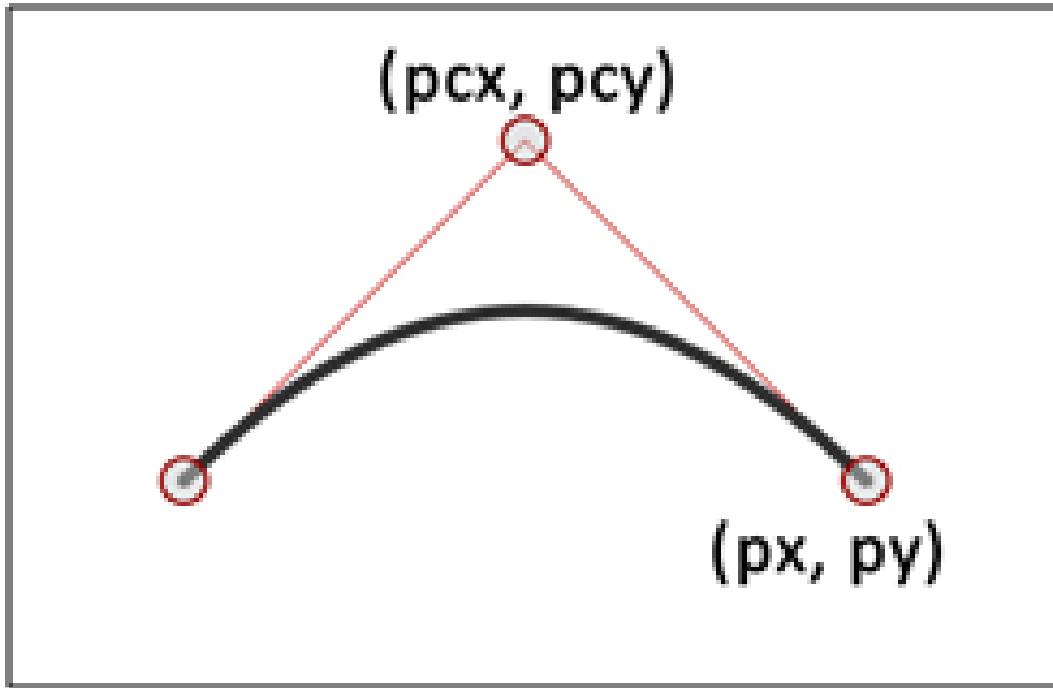


Curvas Cuadráticas

Este tipo de curvas se definen mediante la función:

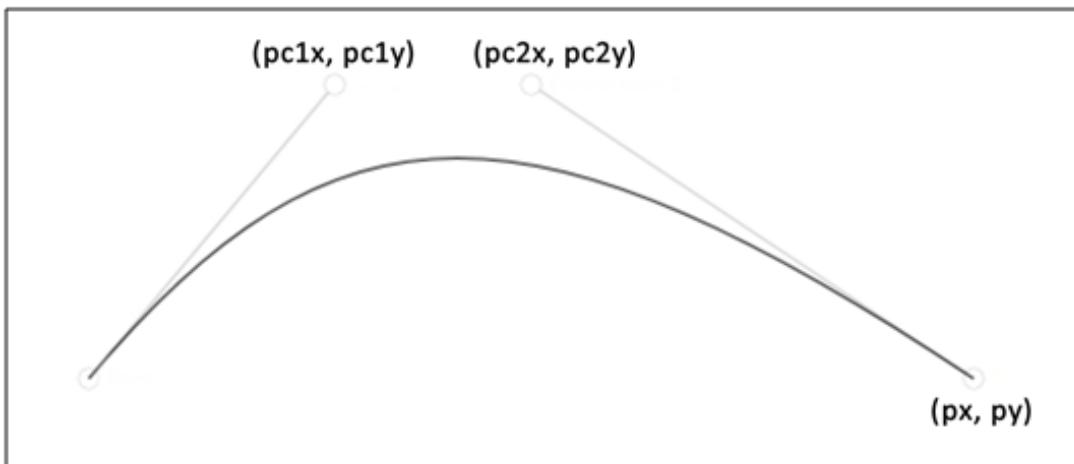
`quadraticCurveTo(px, py, x, y)`

Donde (px, py) son las coordenadas del punto final de la curva y (pcx, pcy) las coordenadas de la posición de la tendencia de la curva, como se ve en la siguiente ilustración:



Curvas de Bezier

La función `bezierCurveTo`, permite dibujar curvas más complejas, ya que se definen dos puntos de tendencia de la curva en lugar de uno:



`bezierCurveTo(pc1x, pc1y, pc2x, pc2y, x, y).`

Ejemplo:

```
<script type="text/javascript">
    function setup() {
        var canvas = document.getElementById('miCanvas');
        if (canvas.getContext) {
            var ctx = canvas.getContext('2d');

            //Rectángulo de contorno del canvas:
            ctx.strokeRect(0, 0, 300, 300);

            //Color de relleno rojo
            ctx.fillStyle = 'rgb(255,0,0)';

            //Inicio del trazado:
            ctx.beginPath();
            //Punto inicial:
            ctx.moveTo(75,40);

            //varias curvas de bezier:
            ctx.bezierCurveTo(75,37,70,25,50,25);
            ctx.bezierCurveTo(20,25,20,62.5,20,62.5);
            ctx.bezierCurveTo(20,80,40,102,75,120);
            ctx.bezierCurveTo(110,102,130,80,130,62.5);
            ctx.bezierCurveTo(130,62.5,130,25,100,25);
            ctx.bezierCurveTo(85,25,75,37,75,40);

            //dibuja la forma con relleno:
            ctx.fill();
        }
    }
</script>
```

Resultado:



INSERCIÓN DE TEXTO

Además de líneas y formas, en un canvas se pueden dibujar textos mediante los métodos:

- `fillText(texto, x, y)`: Dibuja el texto contenido en la cadena texto en las coordenadas (x,y). El texto irá relleno del color que tenga la variable `fillStyle` en ese momento.
- `strokeText(texto, x, y)`: Dibuja el texto contenido en la cadena texto en las coordenadas (x,y), pero sólo el contorno, que irá del color que tenga la variable `strokeStyle` en ese momento.

La variable font indicará en cada momento las propiedades como el tipo de letra, el tamaño, etc, de la misma forma que en CSS.

Ejemplo:

```
<script type="text/javascript">
    function setup() {
        var canvas = document.getElementById('micanvas');
        if (canvas.getContext) {
            var ctx = canvas.getContext('2d');

            //Rectángulo de contorno del canvas:
            ctx.strokeRect(0, 0, 300, 300);

            //Textos:
            ctx.fillStyle="rgb(0,0,255)";
            ctx.font="bold 25px Arial";
            ctx.fillText("Hola",15,50);

            ctx.fillStyle="rgb(0,255,255)";
            ctx.font="30px Verdana";
            ctx.fillText("Mundo",50,100);

            ctx.strokeStyle="rgb(255,0,255)";
            ctx.strokeText("Fin",100,250);
        }
    }
</script>
```

Resultado:



GRADIENTES

Un gradiente es un degradado de color. Pueden definirse varios tipos de gradientes para que sirvan como relleno de las figuras dibujadas.

Gradiente Lineal

1. Definir una línea imaginaria que será la dirección del gradiente, mediante dos coordenadas de inicio (x_1, y_1) y fin (x_2, y_2).
2. Llamar al método `createLinearGradient`, que devolverá una variable de tipo gradiente:

```
var grad=ctx.createLinearGradient(x1, y1, x2, y2)
```

3. Definir los puntos de parada de color que se deseen mediante la función addColorStop:

```
grad.addColorStop (numero, color)
```

El parámetro número es un número entre 0 y 1 que indica la posición del punto de parada en la línea imaginaria. El parámetro color indica el color del gradiente en ese punto.

4. Asignar el contenido de la variable de tipo gradiente a la variable fillStyle.

5. Aplicar el estilo sobre el elemento deseado.

Ejemplo:

```
<script type="text/javascript">
    function setup() {
        var canvas = document.getElementById('micanvas');
        if (canvas.getContext) {
            var ctx = canvas.getContext('2d');

            //Rectángulo de contorno del canvas:
            ctx.strokeRect(0, 0, 300, 300);

            //Creacion de la variable de tipo gradiente:
            var grad=ctx.createLinearGradient(0,0,150,150);

            //Primera parada, al inicio rojo:
            grad.addColorStop(0,"rgb(255,0,0)");

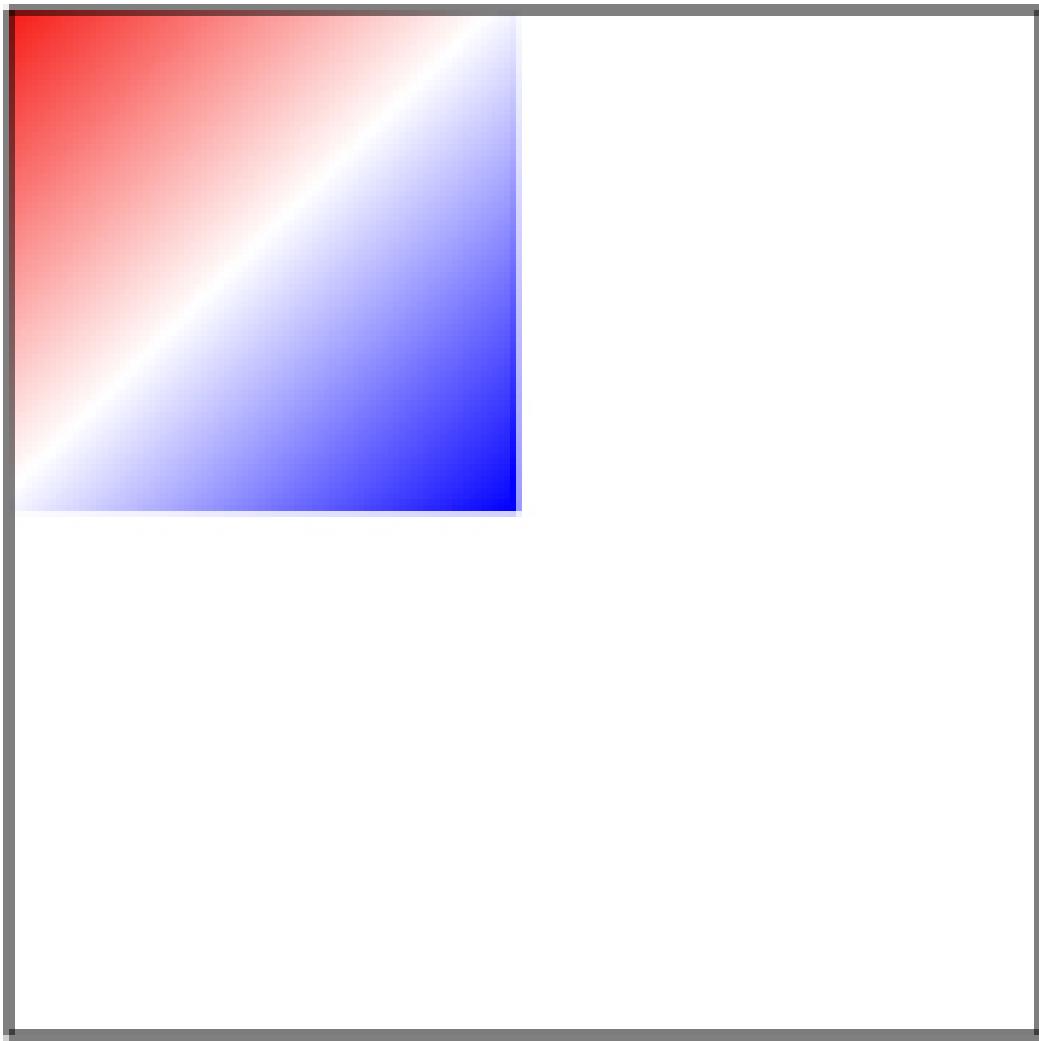
            //Segunda parada, a la mitad blanco:
            grad.addColorStop(0.5,"rgb(255,255,255)");

            //Tercera parada, al final azul:
            grad.addColorStop(1,"rgb(0,0,255)");

            //se asigna el gradiente a la variable fillstyle
            ctx.fillStyle=grad;

            //se dibuja una forma con relleno:
            ctx.fillRect(0,0,150,150);
        }
    }
</script>
```

Resultado:



Gradiente Radial

1. Definir dos círculos imaginarios que contendrán el gradiente, mediante las coordenadas del centro y el radio (x_1, y_1, r_1) (x_2, y_2, r_2).
2. Llamar al método `createRadialGradient`, que devolverá una variable de tipo gradiente:

```
var radgrad=ctx.createLinearGradient(x1, y1, r1, x2, y2, r2)
```

3. Definir los puntos de parada de color que se deseen mediante la función `addColorStop` de la misma forma que se hacía con el

gradiente lineal.

4. Asignar el contenido de la variable de tipo gradiente a la variable `fillStyle`.
5. Aplicar el estilo sobre el elemento deseado.

Ejemplo:

```
<script type="text/javascript">
    function setup() {
        var canvas = document.getElementById('miCanvas');
        if (canvas.getContext) {
            var ctx = canvas.getContext('2d');

            //Rectángulo de contorno del canvas:
            ctx.strokeRect(0, 0, 300, 300);

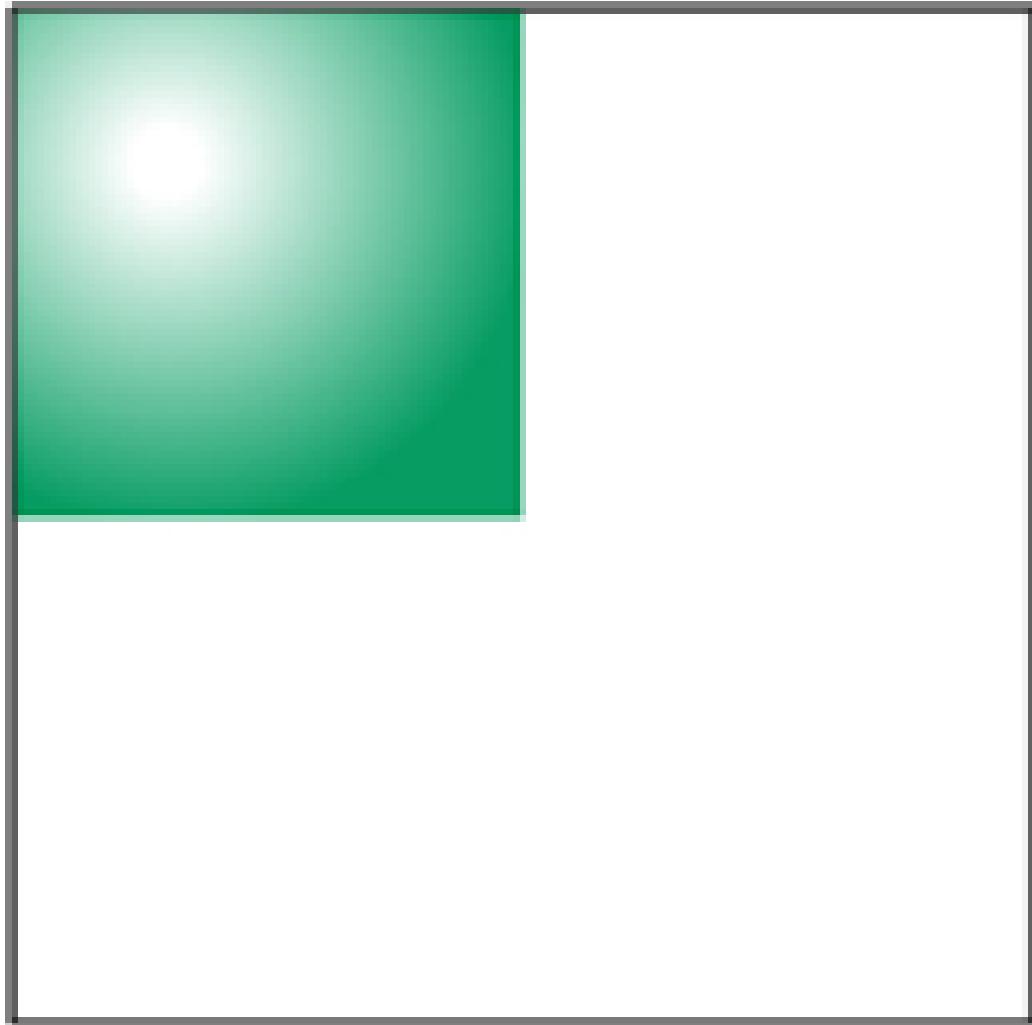
            //Creacion de la variable de tipo gradiente:
            var radgrad = ctx.createRadialGradient(45,45,10,52,50,200);

            //Se definen los puntos de parada de color:
            radgrad.addColorStop(0, '#FFFFFF');
            radgrad.addColorStop(0.5, '#019F62');
            radgrad.addColorStop(1, 'rgb(1,159,98)');

            //se asigna el gradiente a la variable fillstyle
            ctx.fillStyle = radgrad;

            //Se dibuja un elemento para rellenarlo
            ctx.fillRect(0,0,150,150);
        }
    }
</script>
```

Resultado:



TRANSFORMACIONES

Las transformaciones permiten modificar las propiedades de los elementos dibujados en el canvas de forma dinámica. Hay tres tipos básicos de transformaciones:

- Traslación: El método `translate(x,y)` permite cambiar el origen de coordenadas del canvas a otro punto del canvas. Una vez cambiado las figuras que se dibujen tendrán como origen de coordenadas ese nuevo punto.
- Rotación: El método `rotate(rad)` sirve para rotar el canvas sobre su origen, y sólo tiene un parámetro que es el ángulo de rotación medido en radianes en el sentido de las agujas del reloj. El origen

de coordenadas por defecto está siempre en la esquina superior izquierda. Si se quiere cambiar para que gire alrededor de otro punto se deberá utilizar la el método `translate()` antes de aplicar la rotación.

- Escalado: El método `scale(x,y)` permite crea una imagen del canvas a escala, es decir aumenta o disminuye los objetos del canvas en la proporción indicada por x e y. Por tanto los números entre 0 y 1 disminuyen el tamaño del canvas, y los números mayores que 1 lo aumentan. El número 1 deja el canvas en el mismo tamaño.

Salvar y Restaurar

Cuando se aplican transformaciones es importante guardar el estado anterior del canvas para poder restaurarlo más adelante.

El estado del canvas en un momento dado puede ser guardado mediante el método `save()`. Este método no tiene parámetros, y guarda los estilos, transformaciones, etc. que tiene el canvas en el momento de escribirlo.

Después de guardar el estado, se puede seguir dibujando en el canvas normalmente, y cambiar propiedades y métodos.

Cuando se desea recuperar el estado anterior, se utiliza el método `restore()`.

Este método tampoco tiene parámetros y restaura el estado del canvas al que tenía cuando se escribió el método `save()`.

Se pueden anidar varios métodos `save()`, creando así una serie de "capas". Cada vez que se llama al método `restore()` se sale de una capa para ir a la anterior.

El siguiente ejemplo ilustra la utilización de las transformaciones así como el uso de `save()` y `restore()`:

```
<script type="text/javascript">
    function setup() {
        var canvas = document.getElementById('miCanvas');
        if (canvas.getContext) {
            var ctx = canvas.getContext('2d');

            //Rectángulo de contorno del canvas:
            ctx.strokeRect(0, 0, 300, 300);

            //rectángulo azul
            ctx.fillStyle = 'rgb(0,0,255)';
            ctx.fillRect(30, 120, 40, 40);

            //salva
            ctx.save();

            //traslacion 50 pixels hacia abajo:
            ctx.translate(0, 50);

            //rectángulo rojo
            ctx.fillStyle = 'rgb(255,0,0)';
            ctx.fillRect(80, 120, 40, 40);

            //salva
            ctx.save();

            //Rota
            ctx.rotate(0.19);

            //rectángulo verde
            ctx.fillStyle = '#008200';
            ctx.fillRect(130, 120, 40, 40);

            //Restaura
            ctx.restore();

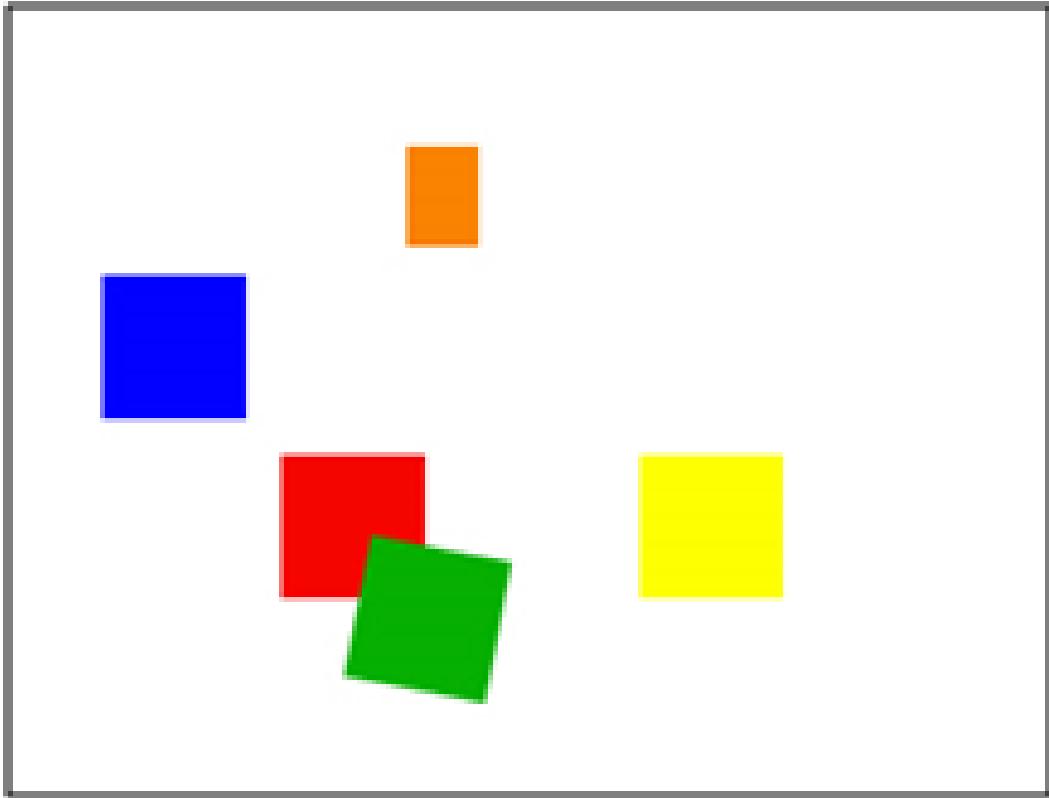
            //rectángulo amarillo
            ctx.fillStyle = '#FFFF00';
            ctx.fillRect(180, 120, 40, 40);

            //Restaura
            ctx.restore();

            //Escala
            ctx.scale(0.5, 0.7);

            //rectángulo naranja
            ctx.fillStyle = '#FF8000';
            ctx.fillRect(230, 120, 40, 40);
        }
    }
</script>
```

El resultado es:



IMÁGENES

Para insertar una imagen dentro del canvas, se utiliza la función:

```
drawImage(img, x, y, width, height)
```

Donde img es una variable que carga el contenido de la imagen mediante la función onload. Las variables x,y indican las coordenadas del canvas donde se colocará la imagen. Los parámetros width y height indican el ancho y el alto de la imagen en píxeles respectivamente.

El siguiente ejemplo insertará la imagen del archivo "imagen1.jpg" en la posición (0,0) del canvas (esquina superior izquierda), con un ancho de 300 píxeles y un alto de 200 píxeles:



```
<!DOCTYPE html>
<html>
  <head>
    <title>Imagen</title>
    <script type="text/javascript">
      function setup() {
        var canvas = document.getElementById('micanvas');
        if (canvas.getContext) {
          var ctx = canvas.getContext('2d');

          var img = new Image();
          img.onload = function(){
            ctx.drawImage(img, 0, 0, 300, 200);
          }
          img.src = 'Imagen1.jpg';
        }
      }
    </script>
  </head>
  <body onload="setup();">
    <canvas id="micanvas" width="300" height="300" style="margin:100px;">
      Si estás viendo este texto, tu navegador no es compatible con canvas
    </canvas>
  </body>
</html>
```

ANIMACIONES

El método `setInterval(f, t)` ejecuta la función `f` cada intervalo de tiempo `t` dado en milisegundos. Es sencillo observar que con este método se pueden crear animaciones en el canvas.

Ejemplo: hacer que un círculo se mueva por el canvas

1. Definir e inicializar las variables con la posición inicial del círculo.
2. Dentro de la función `setup` incluir la llamada a la función `setInterval`:

```
setInterval(dibuja_circulo, 100);
```

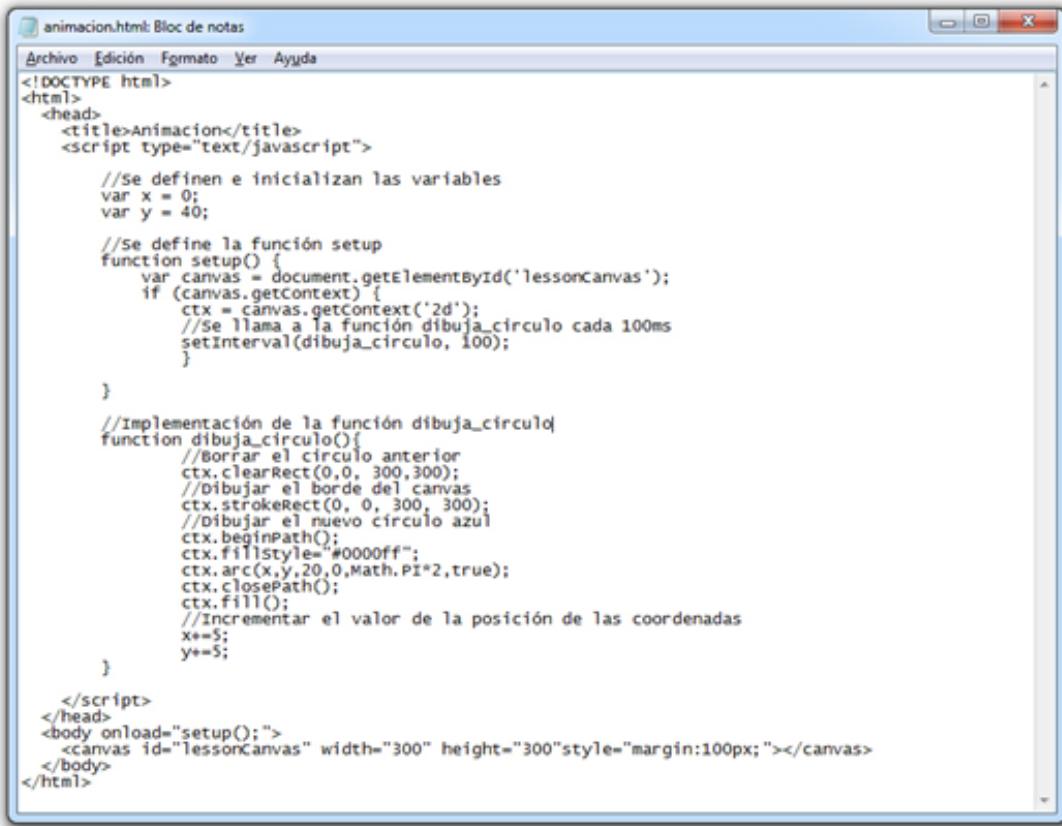
La función `dibuja_circulo` se ejecutará cada 100 milisegundos.

A continuación de la función `setup` escribir el código de la función `dibuja_circulo`:

- Borrar el círculo anterior con la función `clearRect`. Si no se hace éste paso, en cada ejecución de `dibuja_circulo` se dibujará el nuevo círculo encima del anterior.
- Dibujar el borde del canvas con `strokeRect`.
- Dibujar un nuevo círculo en la posición de las variables `(x, y)` mediante la función `arc`.

- Incrementar el valor de las variables x e y en un valor de por ejemplo 5 píxeles, esto se realiza mediante el operador "+=":

El código resultante es:



```

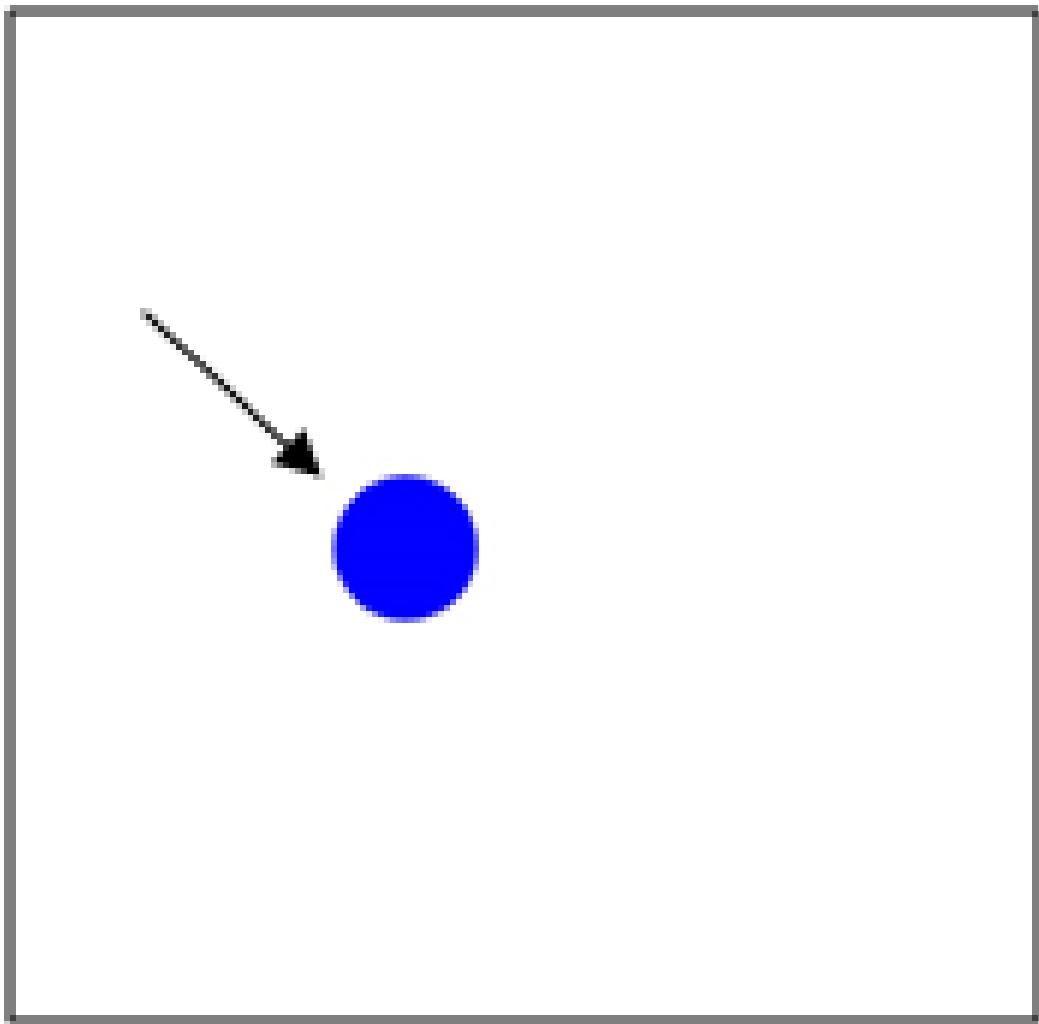
animacion.html: Bloc de notas
Archivo Edición Formato Ver Ayuda
<!DOCTYPE html>
<html>
<head>
<title>Animacion</title>
<script type="text/javascript">
    //Se definen e inicializan las variables
    var x = 0;
    var y = 40;

    //Se define la función setup
    function setup() {
        var canvas = document.getElementById('lessonCanvas');
        if (canvas.getContext) {
            ctx = canvas.getContext('2d');
            //se llama a la función dibuja_circulo cada 100ms
            setInterval(dibuja_circulo, 100);
        }
    }

    //implementación de la función dibuja_circulo
    function dibuja_circulo(){
        //Borrar el círculo anterior
        ctx.clearRect(0,0, 300,300);
        //Dibujar el borde del canvas
        ctx.strokeRect(0, 0, 300, 300);
        //Dibujar el nuevo círculo azul
        ctx.beginPath();
        ctx.fillStyle="#0000ff";
        ctx.arc(x,y,20,0,Math.PI*2,true);
        ctx.closePath();
        ctx.fill();
        //Incrementar el valor de la posición de las coordenadas
        x+=5;
        y+=5;
    }
</script>
</head>
<body onload="setup()">
<canvas id="lessonCanvas" width="300" height="300" style="margin:100px;"></canvas>
</body>
</html>

```

El resultado en el navegador es un círculo que se mueve en diagonal por el área del canvas:



LO QUE HEMOS APRENDIDO

- HTML5 incorpora un entorno para crear imágenes y gráficos de forma dinámica llamado Canvas (lienzo).
- Las etiquetas <canvas> y </canvas> definen el tamaño del lienzo y su identificador, pero lo que se escriba entre ellas sólo se verá en la pantalla si el navegador que se esté utilizando no es compatible con el entorno Canvas.
- Para definir lo que se dibujará en el canvas se utilizan funciones en lenguaje Javascript, que normalmente se insertan dentro del <head> del documento HTML.
- Javascript es un lenguaje de programación orientado a objetos parecido a Java y a C/C++. Puede definir variables, bucles, condiciones, arrays, funciones, métodos, etc.
- Existen funciones Javascript para dibujar rectángulos, trazados rectilíneos, arcos, y curvas en el canvas. También se pueden insertar textos e imágenes.
- Se pueden definir transformaciones para los objetos dibujados de forma dinámica y efectos de relleno como gradientes.
- La función setInterval ejecuta una función Javascript cada cierto tiempo, lo cual permite programar animaciones en la pantalla del canvas sobre los objetos dibujados de forma sencilla.

UNIDAD 2.3

HTML 5 PARTE 2^a

Drag and Drop

- Características de Drag and Drop
- Ejemplo: Cesta de la Compra con Drag and Drop

CARACTERÍSTICAS DE DRAG AND DROP

Como ya se comentó brevemente en el tema 2, Drag and Drop es una API de HTML5 que permite arrastrar y soltar elementos dentro de una página web, lo cual puede resultar muy útil para crear sitios interactivos y dinámicos.

Hacer que un objeto se pueda arrastrar es muy sencillo. Sólo hay que establecer el atributo `draggable="true"` en el elemento que se quiere mover. La función de arrastre se puede habilitar prácticamente en cualquier elemento, como imágenes, enlaces, etc.

Las funciones que indican cómo se realizarán los movimientos de arrastre se programan mediante código Javascript, aunque en esta ocasión, a diferencia de lo que se ha visto en el tema anterior, el código Javascript se inserta al final antes de la etiqueta de cierre `</body>`, debido a que se necesita que los elementos para arrastrar y soltar estén cargados antes de llamar a las funciones Javascript.

FLUJO DE DRAG AND DROP

Para organizar el flujo de Drag and Drop se necesita:

- Un elemento de origen: Es el elemento en el que se origina el movimiento de arrastre. Puede ser una imagen, una lista, un enlace, un objeto de archivo, un bloque `<div>` de HTML, etc.
- La carga de datos: Es lo que se quiere soltar.
- Un elemento de destino: Es el área donde se soltarán los datos. Es la zona de arrastre (o un conjunto de zonas de arrastre) donde se aceptan los datos que el usuario intenta soltar. No todos los elementos HTML pueden ser elementos de destino, por ejemplo, las imágenes.

EVENTOS DE ARRASTRE

Se deben tener en cuenta distintos eventos de los elementos HTML que se arrastrarán para controlar todo el proceso de arrastrar y soltar:

- ondragstart: Se activa cuando comienza el proceso de arrastrado, es decir, el usuario pulsa sobre el objeto.
- ondrag: Se activa cuando el usuario arrastra un objeto por la ventana del navegador. A través de este evento se puede controlar el tiempo en el que el elemento esté capturado a través del ratón.
- ondragenter: Se activa cuando el usuario arrastra un objeto y entra en una zona válida para soltarlo. Permite conocer el primer momento en el que un elemento se posiciona, aunque sea parcialmente, sobre otro.
- ondragleave: Se activa en el momento en el que un elemento que está siendo arrastrado deja de estar posicionado sobre una zona válida para soltarlo.
- ondragover: Se activa cuando el usuario está arrastrando un objeto sobre un objetivo válido para alojarlo.
- ondrop: Se activa cuando se deja de pulsar el botón del ratón tras una operación de arrastre.
- ondragend: Se activa cuando el usuario suelta el objeto al finalizar el arrastre.

EL OBJETO DATATRANSFER

Se trata de un objeto Javascript utilizado para almacenar y recuperar la información que está siendo arrastrada durante la operación de Drag and Drop. Contiene los datos que se envían en la acción de arrastre.

Una instancia de este objeto debe crearse en el código Javascript, suele llamarse con la letra "e". Tiene diferentes funciones y propiedades que conviene conocer:

- e.dataTransfer.effectAllowed: Determina qué tipos de efectos están permitidos cuando se suelta un objeto. Puede tomar los siguientes valores principalmente:
 - copy: Permite que al soltar el objeto arrastrado se cree una copia del original.

- move: El objeto arrastrado puede ser movido
 - link: Al soltar el objeto arrastrado se puede crear un enlace al original.
 - all: Permite todas las acciones anteriores.
- e.dataTransfer.dropEffect: Realizará el efecto indicado, siempre y cuando esté permitido por la propiedad anterior.
 - e.dataTransfer.setData(tipo, datos): Establece el contenido del objeto en el formato "tipo" y se transmite la carga de datos en la variable "datos".
 - e.dataTransfer.getData(tipo, datos): Función complementaria a la anterior, se utiliza para la extracción de los datos de arrastre.

EJEMPLO: CESTA DE LA COMPRA CON DRAG AND DROP

Para comprender las funciones de Drag and Drop descritas anteriormente, a continuación se desarrollará un ejemplo completo paso a paso. Se trata de una web sencilla: en una columna estarán los productos para elegir, otra columna será la zona donde se puedan arrastrar los productos a modo de cesta de la compra.

CREACIÓN DE LA ESTRUCTURA HTML DE LA WEB

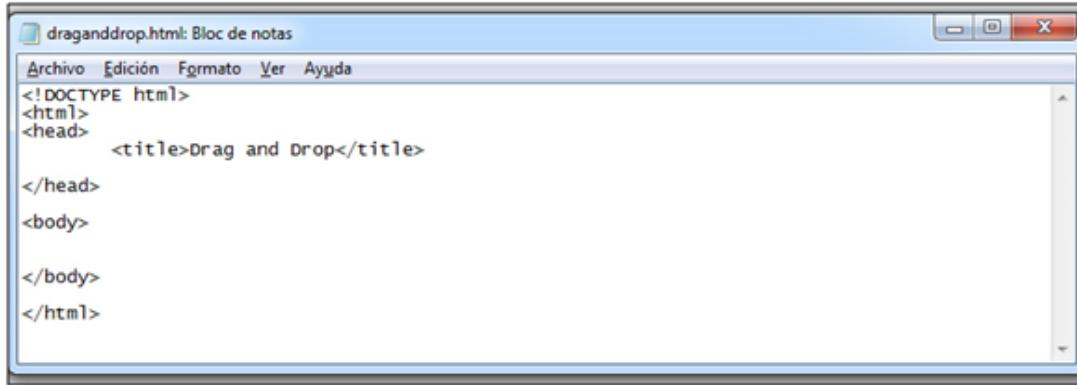
Crear Un nuevo documento HTML con el nombre por ejemplo draganddrop.html

Editar El documento creado con el bloc de notas

Escribir El <!DOCTYPE html> en la primera línea

Escribir Las etiquetas <html>, </html>, <head>, </head>, <body> y </body>

Escribir El título de la página dentro del head



```
<!DOCTYPE html>
<html>
<head>
    <title>Drag and Drop</title>
</head>
<body>
</body>
</html>
```

Crear Dos divisiones <div> para las dos columnas principales que tendrá la web (Dentro del body)

Aplicar A ambos <div> la clase CSS 'cuadro' y alineación centrada

Crear Un párrafo <p> con el título "PRODUCTOS" (Dentro de la primera columna)

Crear Un nuevo <div> con el id="platano" y la propiedad draggable="true" para que pueda ser arrastrado y soltado su contenido

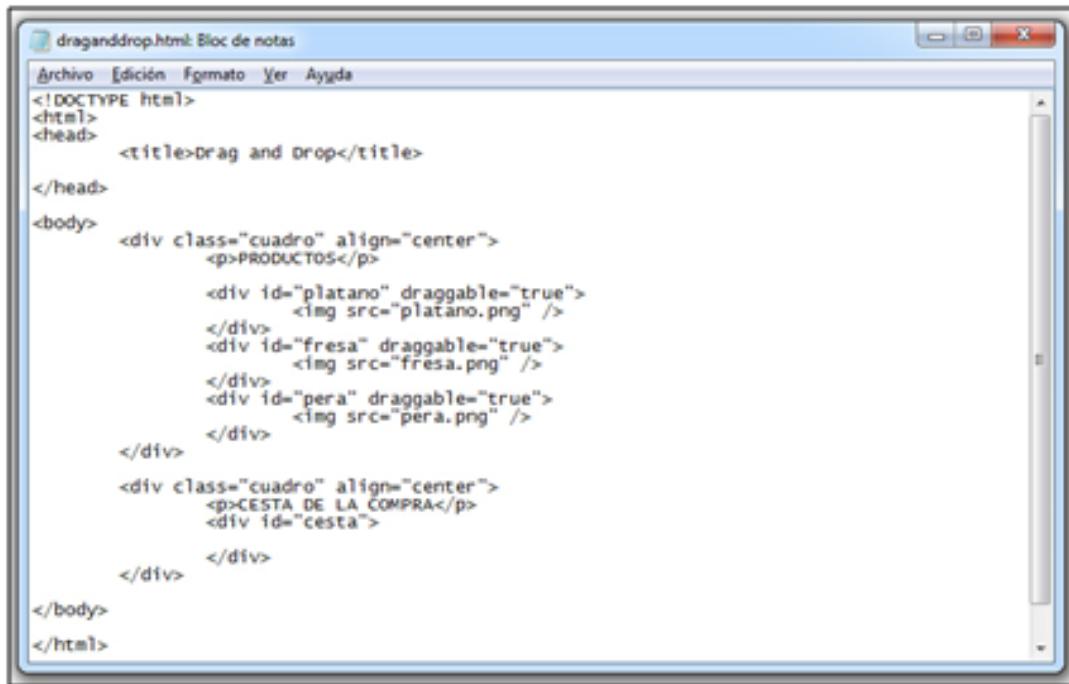
Insertar La imagen platano.png (Dentro de este <div>)

Repetir Los pasos 9 y 10 para "fresa" y "pera"

Crear Un párrafo <p> con el título 'CESTA DE LA COMPRA' (En la segunda columna, es decir, el segundo <div> de clase cuadro)

Crear Un nuevo <div> con el id="cesta". Esta será la zona para soltar los productos arrastrados, por lo que se deja vacío.

El código HTML de esta parte quedaría así:



```
<!DOCTYPE html>
<html>
<head>
    <title>Drag and Drop</title>
</head>
<body>
    <div class="cuadro" align="center">
        <p>PRODUCTOS</p>
        <div id="platano" draggable="true">
            
        </div>
        <div id="fresa" draggable="true">
            
        </div>
        <div id="pera" draggable="true">
            
        </div>
    </div>
    <div class="cuadro" align="center">
        <p>CESTA DE LA COMPRA</p>
        <div id="cesta"></div>
    </div>
</body>
</html>
```

CREACIÓN DE ESTILOS CSS

Por simplificar, los estilos se aplicarán de forma interna en lugar de con un archivo CSS externo.

Editar El documento draganddrop.html con el bloc de notas.

Insertar Debajo del título, las etiquetas <style type="text/css"> y </style>

Asignar Los siguientes estilos a la clase 'cuadro':

- width: 300px. Ancho del cuadro.

- height: 600px. Altura del cuadro.
- margin: 10px. Establecerá un margen de 10 pixels alrededor del cuadro.
- float: left. Los cuadros se alinearán de forma flotante a la izquierda de la pantalla.
- background-color: #BBFFFF. El cuadro tendrá un color de fondo azul claro.
- cursor: pointer. Al pasar el ratón por encima del cuadro, el puntero cambiará de forma.

Asignar Los siguientes estilos al <div> cuyo identificador es "cesta":

- width: 300px. Ancho de la cesta.
- height: 550px. Altura de la cesta.

Asignar Los siguientes estilos de formato de letra a los párrafos <p>:

- font-weight:bold;
- font-family:Verdana, Geneva, sans-serif;

El código correspondiente a esta parte sería:

```
<!DOCTYPE html>
<html>
<head>
    <title>Drag and Drop</title>
    <style type="text/css">
        .cuadro{
            width:300px;
            height:600px;
            margin:10px;
            float:left;
            background-color: #BBBBFF;
            cursor:pointer;
        }
        #cesta{
            width:300px;
            height:550px;
        }
        p {
            font-weight:bold;
            font-family:verdana, Geneva, sans-serif;
        }
    </style>
</head>
```

CREACIÓN DEL CÓDIGO JAVASCRIPT

Se implementarán tres funciones: coger, arrastrar y soltar. Finalmente se conectarán los eventos de los objetos HTML con dichas funciones.

Toda esta parte del código se encierra entre las etiquetas `<script>` y `</script>`, y como ya se ha comentado debe ir justo antes de la etiqueta de cierre `</body>`, debido a que el código Javascript necesita que las imágenes se hayan cargado previamente a la ejecución de dicho código.

Función Coger

Esta función se ejecutará al pulsar sobre uno de los productos.

Definir La función pasándole la instancia al objeto dataTransfer 'e'.

Establecer El identificador del objeto que se haya cogido con la función setData.

Permitir Que se pueda mover el objeto con `e.dataTransfer.effectAllowed = 'move'`

Código de la función:

```
<script>
    function coger(e){
        e.dataTransfer.setData('Text', this.id);
        e.dataTransfer.effectAllowed = 'move';
    }

</script>

</body>
</html>
```

Función Arrastrar

Esta función permitirá que se pueda soltar el elemento al llegar a la zona de destino "cesta".

Código de la función:

```
<script>
    function coger(e){
        e.dataTransfer.setData('Text', this.id);
        e.dataTransfer.effectAllowed = 'move';
    }

    function arrastrar(e){
        e.dataTransfer.dropEffect = 'move';
        return false;
    }

</script>
```

Función Soltar

Esta función se ejecuta cuando el usuario suelta un producto sobre la cesta de la compra.

Crear Una nueva variable de tipo imagen

Transferir El nombre del objeto soltado al de la imagen creada, para que la nueva imagen sea igual a la que se ha soltado

Añadir La nueva imagen en la zona "cesta" de la pantalla del documento, mediante la función appendChild(imagen)

Código de la función:

```
<script>
    function coger(e) {
        e.dataTransfer.setData('Text', this.id);
        e.dataTransfer.effectAllowed = 'move';
    }

    function arrastrar(e) {
        e.dataTransfer.dropEffect = 'move';
        return false;
    }

    function soltar (e) {
        imagen = new Image();
        imagen.src = e.dataTransfer.getData('Text') + '.png';
        document.getElementById('cesta').appendChild(imagen);
    }

</script>
```

Conexión entre Funciones y Eventos

Conectar La función ‘coger’ con el evento ondragstart de cada uno de los <div> ‘plátano’, ‘fresa’ y ‘pera’.

Conectar La función ‘arrastrar’ con el evento ondragover del elemento ‘cesta’.

Conectar La función ‘soltar’ con el evento ondrop del elemento "cesta".

Código de esta parte:

```
<script>
    function coger(e) {
        e.dataTransfer.setData('Text', this.id);
        e.dataTransfer.effectAllowed = 'move';
    }

    function arrastrar(e) {
        e.dataTransfer.dropEffect = 'move';
        return false;
    }

    function soltar (e) {
        imagen = new Image();
        imagen.src = e.dataTransfer.getData('Text') + '.png';
        document.getElementById('cesta').appendChild(imagen);
    }

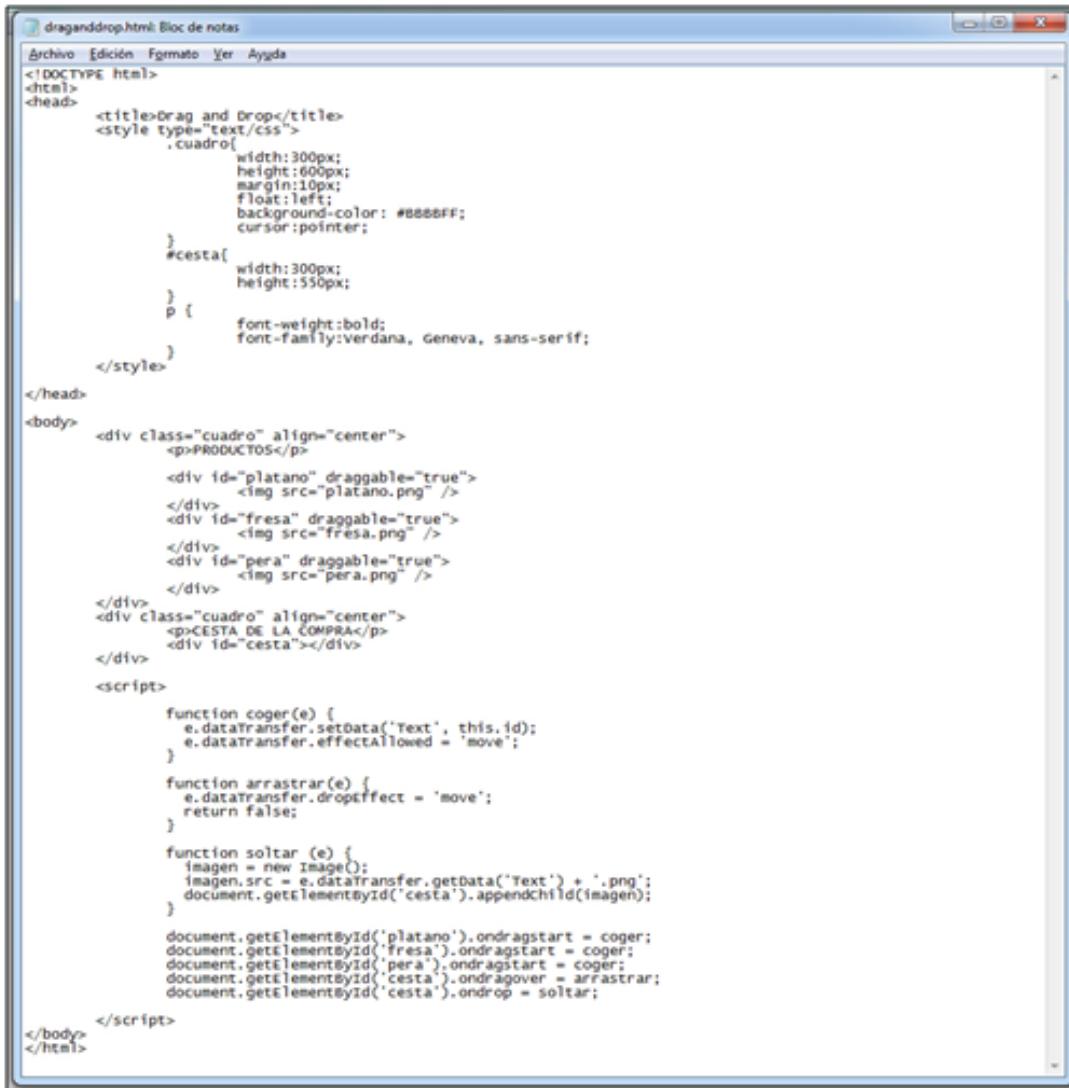
    document.getElementById('platano').ondragstart = coger;
    document.getElementById('fresa').ondragstart = coger;
    document.getElementById('pera').ondragstart = coger;

    document.getElementById('cesta').ondragover = arrastrar;
    document.getElementById('cesta').ondrop = soltar;

</script>
```

RESULTADO

El código del ejemplo completo es:



```
draganddrop.html Bloc de notas
Archivo Edición Formato Ver Ayuda
<!DOCTYPE html>
<html>
<head>
    <title>Drag and Drop</title>
    <style type="text/css">
        .cuadro{
            width:300px;
            height:600px;
            margin:10px;
            float:left;
            background-color: #BBBBFF;
            cursor:pointer;
        }
        #cesta{
            width:300px;
            height:550px;
        }
        p {
            font-weight:bold;
            font-family:verdana, Geneva, sans-serif;
        }
    </style>
</head>
<body>
    <div class="cuadro" align="center">
        <p>PRODUCTOS</p>
        <div id="platano" draggable="true">
            
        </div>
        <div id="fresa" draggable="true">
            
        </div>
        <div id="pera" draggable="true">
            
        </div>
    </div>
    <div class="cuadro" align="center">
        <p>CESTA DE LA COMPRA</p>
        <div id="cesta"></div>
    </div>
    <script>
        function coger(e) {
            e.dataTransfer.setData('Text', this.id);
            e.dataTransfer.effectAllowed = 'move';
        }

        function arrastrar(e) {
            e.dataTransfer.dropEffect = 'move';
            return false;
        }

        function soltar (e) {
            imagen = new Image();
            imagen.src = e.dataTransfer.getData('Text') + '.png';
            document.getElementById('cesta').appendChild(imagen);
        }

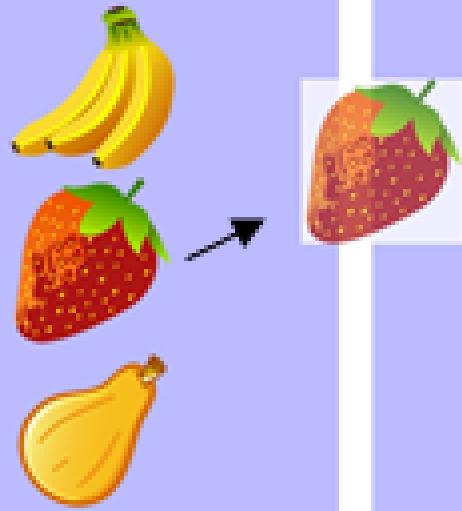
        document.getElementById('platano').ondragstart = coger;
        document.getElementById('fresa').ondragstart = coger;
        document.getElementById('pera').ondragstart = coger;
        document.getElementById('cesta').ondragover = arrastrar;
        document.getElementById('cesta').ondrop = soltar;
    </script>
</body>
</html>
```

Al ejecutar el archivo draganddrop.html aparecerá la siguiente pantalla:

PRODUCTOS**CESTA DE LA COMPRA**

Se podrán arrastrar los objetos de la columna de la izquierda y soltarlos a la derecha tantas veces como se desee hasta completar el espacio de la cesta.

PRODUCTOS



CESTA DE LA COMPRA



PRODUCTOS



CESTA DE LA COMPRAS



LO QUE HEMOS APRENDIDO

- Drag and Drop es una API de HTML5 que permite arrastrar y soltar elementos dentro de una página web.
- Para hacer que un elemento HTML se pueda arrastrar hay que establecer su atributo draggable="true".
- Por otro lado hay que configurar las acciones de Drag and Drop mediante funciones de código Javascript.
- Para organizar el flujo de Drag and Drop se necesitan: un elemento de origen, una carga de datos y un elemento de destino.
- Existen una serie de eventos de los elementos HTML relacionados con el proceso de arrastrado. Estos eventos deberán relacionarse con las funciones Javascript para que el conjunto funcione correctamente.
- El objeto dataTransfer es una instancia de una clase de Javascript que se utiliza para almacenar y recuperar la información que está siendo arrastrada durante la operación de Drag and Drop.

GLOSARIO

Booleano

Un tipo de dato booleano es aquel que puede representar valores de lógica binaria, es decir 2 valores, que normalmente representan verdadero o falso.

Búfer

Es un espacio de memoria en el que se almacenan datos para evitar que el programa o recurso que los requiere se quede sin datos durante una transferencia.

Clase

Es una construcción que se utiliza en programación como un modelo o plantilla para crear objetos de ese tipo. El modelo describe el estado y el comportamiento que todos los objetos de la clase comparten.

Cookies

Una cookie (del inglés, galleta) es una pequeña información enviada por un sitio web que se almacena en el navegador del usuario, de manera que el sitio web puede consultar la actividad previa de dicho usuario.

Enlace o Hipervínculo

Es un elemento que permite saltar de una página a otra del mismo sitio o bien de una página a otra ubicada en cualquier parte de internet.

Flash

Es una tecnología para crear animaciones gráficas vectoriales independientes del navegador y que necesitan poco ancho de banda

para mostrarse en los sitios web. Los navegadores necesitan instalar un plug-in para mostrar animaciones en Flash.

GPS

Del inglés, Global Positioning System, significa Sistema de Posicionamiento Global. Es un sistema global de navegación por satélite que permite determinar en todo el mundo la posición de un objeto, una persona o un vehículo con una precisión hasta de centímetros (si se utiliza GPS diferencial), aunque lo habitual son unos pocos metros de precisión.

IP

Del inglés Internet Protocol, una dirección IP es una etiqueta numérica que identifica a un dispositivo dentro de una red como por ejemplo internet.

Metainformación

La metainformación o metadatos son "datos sobre datos". En términos informáticos, la metainformación son una serie de palabras que se incluyen dentro del código de una página web para hacer más sencilla la indexación a los buscadores de internet de la información que contiene dicha web.

Navegador

Un navegador, navegador web, o navegador de internet es una aplicación que interpreta la información de documentos HTML o de otros lenguajes y permiten su visualización.

Píxel

Del inglés picture element significa elemento de imagen. Es la menor unidad homogénea en color que forma parte de una imagen digital.

Smartphone

En inglés, ‘teléfono inteligente’, este término se utiliza para diferenciar los teléfonos que poseen funcionalidades añadidas a las de un teléfono común, como por ejemplo acceso a internet, pantalla táctil, instalación de aplicaciones, etc.

URL

Del inglés Uniform Resource Locator, o localizador de recursos uniforme. Es una secuencia de caracteres, de acuerdo a un formato estándar, que se usa para nombrar recursos en Internet para su localización o identificación.

Wifi

Mecanismo de conexión de dispositivos de forma inalámbrica.

BIBLIOGRAFÍA

Título: Crea tu Página Web

Editorial: Anaya

Autor: Coley, L., Fulton, J., Scott M. Fulton

Año: 2007

Título: El Gran Libro de HTML5, CSS3 y Javascript

Editorial: Marcombo

Autor: Gauchat, J.D.

Año: 2012

Título: HTML5 Canvas

Editorial: O'Reilly Media

Autor: Fulton, S., Fulton, J.

Año: 2011

Título: Responsive Web Design

Editorial: A Book Apart

Autor: Marcotte, E.

Año: 2011

Páginas Web:

Título: Especificación Oficial HTML 4.0

Web: <http://www.w3.org/TR/html401>

Título: Especificación Oficial HTML5

Web: <http://www.w3.org/TR/2011/WD-html5-20110525/>

Título: Especificaciones Oficiales CSS

Web: <http://www.w3.org/Style/CSS/specs>

Título: Especificación Oficial Javascript (ECMA-262)

Web:<http://www.ecma-international.org/publications/standards/Ecma-262.htm>

Título: Modernizr

Web: <http://modernizr.com/>