Programación HTML5 y CSS3

HTML5 y CSS3

Formador: Vladimir Bataller

https://es.linkedin.com/in/vladimirbataller
vladimirbataller@yahoo.es

Horario:

L a V de 9:00 a 14:00 y 15:00 a 18:00

Requisitos:

- HTML y CSS
- Javascript.

CSS3

profesor: Vladimir Bataller

Tags semánticos

Permiten describir el significado de cada bloque de texto, imagen, hipervínculo o cualquier otro recurso.

header: delimita la cabecera de una página, sección o artículo, en la cual suele aparecer el nombre y logo de la sección.

footer: delimita el píe de una página, sección o artículo, en el cual suele aparecer el copy right, información legal, "quienes somos", ampliación de información, etc.

section: divide la página en secciones y agrupa elementos que constituyen una sección diferenciada de otras secciones de la página. *article* y *aside* son casos especializados de *section*.

article: permite delimitar una unidad que trata un tema en concreto y que es el tema principal de página, por lo que en una página suele haber un único bloque *article*.

aside: permite delimitar un bloque de contenido secundario diferenciado del contenido principal de la página.

hgroup: agrupa varias etiquetas de cabecera h1,h2... h6 indicando que forman parte de una misma unidad de titulo y subtítulos.

nav: delimita el menú del sitio web mediante el cual se navega a las secciones del sitio. Esto facilita a herramientas de análisis de sitios el conocimiento de la estructura del sitio.

CSS3

profesor: Vladimir Bataller

Tags semánticos

Permiten describir el significado de cada bloque de texto, imagen, hipervínculo o cualquier otro recurso.

figure: permite agrupar varios elementos gáficos para indicar su vinculación semántica.

figcaption: permite declarar un texto la explicación de una imágen (pie de foto).

progress: declara una barra de progreso que se puede modificar desde Javascript para indicar el avance de una tarea.

time: permite declarar semánticamente una fecha u hora.

meter: permite declarar una medida escalar.

mark: permite destacar una parte del texto de mayor relevancia.

Fuentes incrustadas

En CSS3 se permite hacer referencia a recursos tipográficos (archivos de tipos de fuente) alojados en un servidor para ser usados en la página.

La instrucción css ***@font-face*** delimita la declaración de un tipo de fuente importado

```
@font-face{
        font-family:nombreQueSeLeAsignaALaFuente;
        src: url(fonts/Don_Quixote.eot), url(fonts/Don_Quixote.ttf);
}
```

Una vez declarado el nuevo tipo de fuente, se podrá asignar a la propiedad ***font-family*** de cualquier elemento o selector de la página.

Texto con sombra

En CSS3 se puede agregar sombra a los textos presentados (principalmente a títulos), sin necesidad de emplear imágenes.

Para agregar sombra a un texto, a este se le debe aplicar la propiedad CSS3 ***text-shadow*** la cual tiene la siguiente sintáxis:

text-shadow: desplHoriz desplVert difuminado colorSombra

Donde:

desplHoriz: es el desplazamiento horizontal de la sombra (hay que indicar las unidades).

desplVert: es el desplazamiento horizontal de la sombra (hay que indicar las unidades).

difuminado: cuanto mayor, mas difuminada se verá la sombra (hay que indicar las unidades).

colorSombra: color que se asigna a la sombra (nobre de color, código de color, función rgb).

CSS3

profesor: Vladimir Bataller

Texto en columnas

En CSS3 se puede distribuir el texto interior de un elemento en varias columnas. Para ello se puede emplear la propiedad column-count a la cual se le asigna un valor numérico con el numero de columnas deseado

Las principales propiedades CSS3 para definir el formato del texto en columnas son:

column-count: número de columnas desado

column-gap: espacio entre columnas

column-rule: propiedades de la línea de separación de columnas (ancho tipo color)

Formato de cajas

En CSS3 se ha incluido la posibilidad de configuar la sombra de las casjas así como que las cajas tengan las esquinas redondeadas.

Esquinas redondeadas

border-radius: radio de la circunferencia que da forma de cada esquina.

Sombra en cajas

box-shadow: desplHoriz desplVert difuminado colorSombra

Donde:

desplHoriz: es el desplazamiento horizontal de la sombra (hay que indicar las unidades).

desplVert: es el desplazamiento horizontal de la sombra (hay que indicar las unidades).

difuminado: cuanto mayor, mas difuminada se verá la sombra (hay que indicar las unidades).

colorSombra: color que se asigna a la sombra (nombre de color, código de color, función rgb).

Imágenes de borde

En CSS3 se puede emplear una imagen para definir un borde de un elemento basado en dicha imagen. para ello se emplea la propiedad **border-image**.

La propiedad border-image tiene la siguiente sintaxis:

border-image: ***rutaDeLaImagen anchoEsquinas altoEsquinas modoEntreEsquinas***

Donde *anchoEsquinas* y *altoEsquinas* indica el rectángulo de la imagen que se emplea para las esquinas. El último argumento es el modo de repetición de la imagen entre esquinas:

round: la parte de la imagen entre las esquinas se repite escalándose ligeramente para encajar un número exacto de repeticiones en el espacio total.

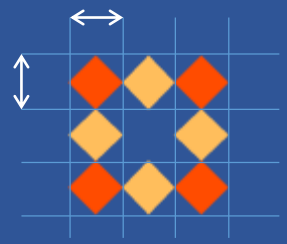
stretch: la parte de la imagen entre las esquinas se estira para abarcar todo el espacio a cubrir.

repeat: la parte de la imagen entre las esquinas se repite sin escalarse por lo que en la última repetición la imagen puede quedar cortada.

space: la parte de la imagen entre las esquinas se repite sin escalarse, pero se insertan espacios para encajar un número exacto de repeticiones en el espacio total.

Ejemplo:

border-image: url(imagen/border.png) 28 28 round;



Transparencia

En CSS3 se dispone de dos modos de transparencia: Transparencia solamente del fondo de un elemento o transparencia de elemento y todos sus elementos interiores.

Transparencia solamente del fondo del elemento

background-color: rgba(100,100,100,0.6);

Mediante el uso de la función rgba, además de especificar el color, también se indica la opacidad del fondo. El valor de la opacidad toma valores entre 0.0 (completamente transparente) y 1.0 (completamente opaco).

Transparencia del elemento y de todos sus elementos interiores

opacity: 0.6;

Mediante el uso de la propiedad ***opacity*** se indica la opacidad del elemento y sus elemento anidados. Toma valores entre 0.0 (completamente transparente) y 1.0 (completamente opaco).

Transformaciones

Los elementos del documento se pueden modificar (rotar, trasladar, escalar, inclinar, etc) mediante la propiedad CSS3 ***transform***.

Propiedad *transform*

La propiedad ***transform*** puede recibir una o varias funciones de transformación separadas por espacios. Algunas de estas funciones de transformación son:

rotate(15deg) -> rota el elemento 15 grados.

translate(40px,0px) -> traslada el elemento 40 px hacia la derecha.

scale(0.8,0.8) -> reduce el tamaño del elemento al 80% del original.

skew(-15deg,0) -> sesga el elemento desplazando sus filas de pixels horizontalmente.

rotateX(180deg) -> rota el elemento 180 grados al rededor del eje X.

Transiciones

Cuando una propiedad de estilo cambia su valor, se puede indicar mediante la propiedad ***transition*** que ese cambio se realice de forma gradual a lo largo de un periodo de tiempo.

Propiedad *transition*

transition: nombreDeLaPropiedad tiempoDeTransicion curvaDeTransicion

Donde:

nombreDeLaPropiedad: nombre de la propiedad cuya transición se quiere configurar.

tiempoDeTransicion: duración de la transición.

curvaDeTransicion: especifica el modo en el que se produce la transición (linear, ease, ease-in, ease-out, ease-in-out)

Ejemplo:

```
transition: background-color 1s linear
```

CSS3

profesor: Vladimir Bataller

Animaciones

Permiten especificar la evolución de una o varias propiedades CSS. Se declaran con la instrucción ***@keyframes*** y se aplican mediante la propiedad ***animation-name***, ***animation-duration*** y ***animation-timing-function***.

Declaración

En la declaración se especifican los valores que toman las propiedades animadas a lo largo de la evolución.

```
@keyframes nombreDeLaAnimacion{
        0%{ lista de propiedades}  ... 50%{ lista de propiedades} ... 100%{ lista de propiedades}
}
```

Aplicación de la animación

La animación se aplica a un elemento al cual se le aplican mediante selectores css las propiedades:

animation-name: nombre de la animación que se quiere aplicar.

animation-duration: duración de la animación (desde el 0% al 100%).

animation-timing-duration: especifica el modo en el que se produce la animación (linear, ease, ease-in, ease-out, ease-in-out).

WebWorkers (I)

Por defecto Javascript se ejecuta en el navegador en un único hilo. Mediante los Web Workers se puede ejecutar un hílo en paralelo con el hilo principal de Javascript. Además ambos hilos podrán enviarse mensajes.

El código JavaScript del hilo del Web Worker se debe guardar en un archivo *.js.

La creación e inicio de ejecución del WebWorker se realiza de la siguiente forma (puede lanzar excepciones):

w = new Worker(*rutaDelArchivoJsDelWebWorker*);

El hilo principal se puede suscribir a los eventos de envío de mensajes del Worder mediante **onmessage**:

w.onmessage = *funcionManejadoraDelEvento*;

Para detener el WebWorker se ejecuta su método ***terminate***:

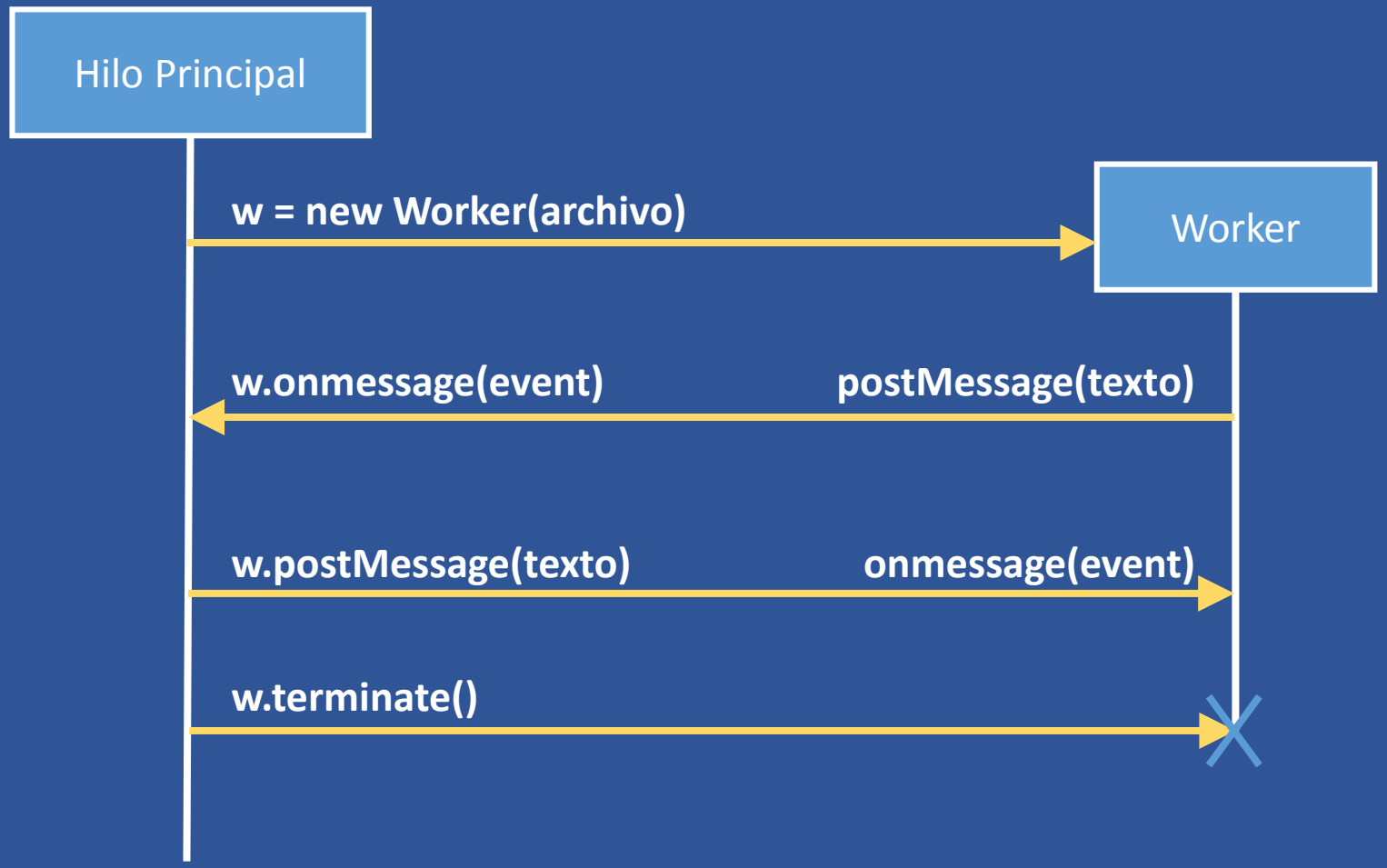
w.terminate();

Para enviar mensajes al hilo se emplea el método ***postMessage***:

w.postMessage(*contenidoDelMensaje*);

WebWorkers (I)

Por defecto Javascript se ejecuta en el navegador en un único hilo. Mediante los Web Workers se puede ejecutar un hílo en paralelo con el hilo principal de Javascript. Además ambos hilos podrán enviarse mensajes.



HTML5

profesor: Vladimir Bataller

Formularios

Los formularios en HTML5 incluyen nuevos tipos de input, nuevos atributos y un api para su validación.

Los input pueden tener los siguientes nuevos tipos, de modo que el navegador puede ayudar a introducir el tipo específico de datos y a su validación.

search, email, url, tel, date, time, number, range, color, datetime-local, month, week, datetime

Los controles de formulario en HTML5 disponen de los siguientes nuevos atributos relativos a la validación:

required, pattern, min, max, minlength, maxlength.

Los controles de formulario en HTML5 disponen de los siguientes nuevos atributos relativos a su comportamiento y visualización:

placeholder, disabled, readonly, multiple, autofocus, form, autocomplete, list.

multiple: para input de type email o file permite seleccionar varios valores.

autocomplete: si se le asigna el valor "off" impide el autocompletado del navegador.

form: permite indicar el id de uno o varios formularios a los que pertenece el input, aunque no esté dentro de él.

Validación de formularios

Además de los atributos de validación de los controles de formularios, estos disponen de un api JavaScript para verificar si el control cumple o no las reglas de validación y en su caso qué reglas no cumple.

Los controles de formulario disponen del método ***checkValidity()*** que retorna true si el control cumple todas las reglas de validación y false en caso contrario.

Los controles de formulario disponen de la propiedad ***validity*** que es un objeto con una serie de propiedades booleanas para indicar qué tipo de regla de validación es la que no se está cumpliendo.

valueMissing: el campo es obligatorio y no se ha insertado ningún valor.

rangeUnderflow: el valor está por debajo del rango indicado.

rangeOverflow: el valor está por encima del rango indicado.

stepMismatch: no se cumple el step (sato entre valores) especificado.

badInput o **typeMismatch**: el formato no coincide con el esperado.

SVG

Lenguaje de gráficos vectoriales basado en XML. Actualmente es una especificación W3C. (http://www.w3.org/TR/SVG/index.html)

En html5, se puede incluir código svg directamente dentro del documento.

```
<!DOCTYPE html>
<html lang="es">
        <head></head>
        <body>
                <svg>.....</svg>
        </body>
</html>
```

HTML5

profesor: Vladimir Bataller

SVG

Lenguaje de gráficos vectoriales basado en XML. Actualmente es una especificación W3C. (http://www.w3.org/TR/SVG/index.html)

```
<svg>
        <circle id="cir" cx="100" cy="100" r="80" fill="blue" stroke="black" stroke-with="5"/>
        <rect id="rec" x="150" y="150" width="100" height="50"
                        fill="red" stroke="black" stroke-with="5" />
</svg>
```

Video

La etiqueta ***video*** permite la reproducción de vídeos sin necesidad de ningún plugin adicional instalado en el navegador.

En la etiqueta ***video*** se pueden anidar una o varias etiquetas ***source*** para indicar los archivos en diferentes formatos que proporcionan el vídeo a reproducir.

```
<video>
        <source src="img/movie.mp4" type="video/mp4"/>
        ....
</video>
```

La etiqueta vídeo dispone de varios atributos de los cuales los más destacables son:

controls="controls" -> Muestra los controles para manejar la reproducción.

autoplay="autoplay" -> Reproduce el vídeo desde el principio.

loop="loop" -> Una vez termina la reproducción empieza de nuevo.

Video API

La etiqueta ***video*** proporciona un API Javascript que permite su control programático. A continuación se describen sus atributos, métodos y eventos más destacables.

atributos

paused -> toma el valor true cuando el vídeo está parado.

ended -> toma el valor true cuando el vídeo ha terminado de reproducirse.

currentTime -> retorna o recibe el instante de reproducción actual (en segundos).

duration -> retorna la duración total del vídeo.

métodos

play() -> inicia la reproducción del vídeo.

pause() -> detiene la reproducción del vídeo.

eventos

onended -> se lanza una vez el vídeo ha terminado de reproducirse.

ontimeupdate -> se lanza repetidamente conforme se va reproduciendo el vídeo.

Audio

La etiqueta ***audio*** permite la reproducción de archivos de sonido sin necesidad de ningún plugin adicional instalado en el navegador.

En la etiqueta ***audio*** se pueden anidar una o varias etiquetas ***source*** para indicar los archivos en diferentes formatos que proporcionan el audio a reproducir.

```
<audio>
        <source src="img/song.mp3" type="audio/mp3"/>
        ....
</audio>
```

La etiqueta ***audio*** dispone de varios atributos de los cuales los más destacables son:

controls="controls" -> Muestra los controles para manejar la reproducción.

autoplay="autoplay" -> Reproduce el audio desde el principio.

loop="loop" -> Una vez termina la reproducción empieza de nuevo.

Canvas

La etiqueta ***canvas*** permite representa un mapa de bits del cual se tiene control absoluto mediante su API JavaScript.

En la etiqueta ***canvas*** tiene los atributos ***width*** y ***height*** para indicar el alto y ancho de los pixels de la imagen generada. De este modo, con el api no se podrá dibujar mas allá de los valores indicados con ***width*** y ***height***. De todas formas, al igual que ocurre con la etiqueta ***img***, independientemente de que esta tenga un tamaño en pixels, posteriormente, mediante propiedades CSS se puede mostrar en el documento con otras dimensiones.

```
<canvas width="500" height="400"></canvas>
```

Para empezar a dibujar desde JavaScript, hay que obtener el objeto Context contenido en el canvas, a continuación se muestra como obtener su context 2D.

```
var ctx = document.querySelector("canvas").getContext("2d");
```

HTML5

profesor: Vladimir Bataller

Canvas

A la hora de dibujar una figura geométrica, se deben seguir tres fases implícita o explícitamente: definición de la forma de la figura, selección de colores y estilos y dibujo propiamente dicho en el canvas.

1.- DEFINICIÓN DE LA FIGURA

```
context.beginPath(); // Empezamos una nueva figura
context.moveTo(10, 10); // Nos posicionamos en un punto para empezar a dibujar desde ahí.
context.lineTo(110, 10); // Definimos una línea desde el punto anterior hasta el punto indicado.
context.lineTo(110, 110);// Definimos una línea desde el punto anterior hasta el punto indicado.
context.closePath(); // Terminamos de definir la figura con una línea desde el último punto al primero.
```

2.- COLORES A EMPLEAR

```
context.fillStyle = 'lime'; // Color de relleno.
context.strokeStyle = 'black'; // Color del contorno
context.lineWidth = 4; // Ancho de la línea del contorno.
```

3.- DIBUJAR EN EL CANVAS

```
context.fill(); //Dibujar el relleno
context.stroke(); //Dibujar el contorno.
```

Canvas

El API del Canvas permite dibujar formas primitivas, líneas, circulos, rectángulos, etc., permite controla pixel a pixel, insertar imágenes y realizar transformaciónes.

MÉTODOS QUE DEFINEN Y DIBUJAN A LA VEZ UN RECTÁNGULO

```
context.fillRect(x1, y1, ancho, alto); //Dibujar el relleno
context.strokeRect(x1, y1, ancho, alto); //Dibujar el contorno.
```

MÉTODO QUE DEFINE UN ARCO DE CIRCUNFERENCIA

```
context.arc(x, y, r, ang_ini, ang_fin); //Arco de circunferencia
```

ASIGNAR A UN HIPERVÍNCULO LA DESCARGA DE LA IMÁGEN ASOCIADA AL CANVAS

```
var data = document.querySelector("canvas").toDataURL("image/png");
event.currentTarget.href = data; // Asignamos los pixels de la imagen png al hipervínculo.
event.currentTarget.download = "dibujo.png"; //Nombre sugerido para el archivo.
```