EFA
MORATALAZ

*1º CFGS Desarrollo de Aplicaciones Web*

# *LENGUAJE DE MARCAS Y SISTEMAS DE GESTIÓN DE LA INFORMACIÓN*

*ROBERTO SÁNCHEZ CHACÓN*

# UT5 – UN NUEVO ESTÁNDAR-HTML 5

EFA
MORATALAZ

*1º CFGS Desarrollo de Aplicaciones Web*

*LENGUAJE DE MARCAS Y SISTEMAS DE GESTIÓN DE LA INFORMACIÓN*

# UT5 – UN NUEVO ESTÁNDAR-HTML 5

1. FORMULARIOS
2. MULTIMEDIA
3. GRÁFICOS
4. ANIMACIONES
5. ETIQUETAS SEMANTICAS HTML 5

# FORMULARIOS

Algunos formularios pueden ser visual y cognitivamente difíciles de usar para algunas personas. Por ello, debemos **hacer formularios accesibles, fáciles de usar** para **todas las personas**.

En los formularios HTML5 también se notan los esfuerzos por incluir elementos y atributos que mejoren la accesibilidad de la web.

Dos guías de estilo importantes que es necesario respetar:

- Para cada campo de formulario debe existir un **<label>** asociado.
- Para formularios largos o complejos, usar los elementos **<fieldset>** y **<legend>** para agrupar los elementos de forma lógica.

```
1. <label for="first_name">First Name</label>
2. <input id="first_name" type="text" name="fname"/>
```

First name:

Output format

- (•) Text file
- ( ) CSV file
- ( ) HTML file

```
1. <fieldset>
2. <legend>Output format</legend>
3. <div>
4. <input type="radio" name="format" id="txt" value="txt" checked>
5. <label for="txt">Text file</label>
6. </div>
7. <div>
8. <input type="radio" name="format" id="csv" value="csv">
9. <label for="csv">CSV file</label>
10. </div>
11. […]
12. </fieldset>
```

En la nueva versión de HTML5 se han creado nuevos tipos de input, atributos y elementos relacionados con los formularios.

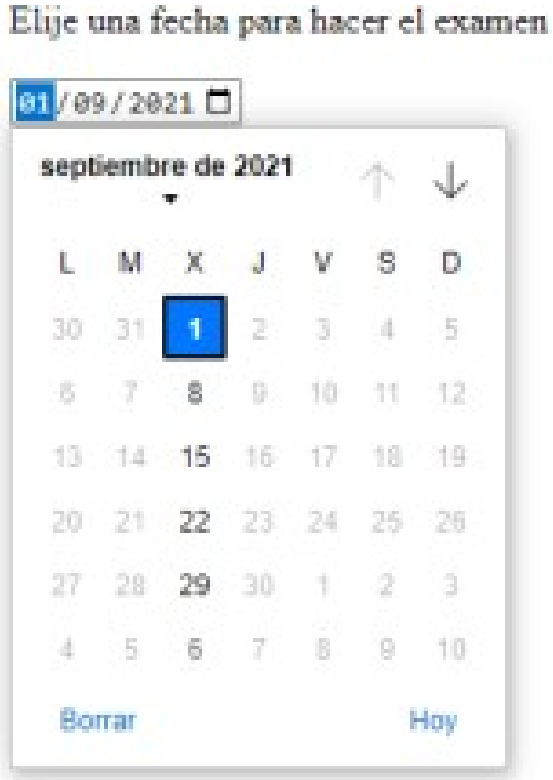
Los navegadores tienen una implementación nativa de ciertas acciones que antes se lograban mediante JavaScript.

VENTAJAS

- ✓ Se reduce el código JavaScript.
- ✓ Menor probabilidad de errores.
- ✓ Mejora del rendimiento general de la web.

```
<!DOCTYPE HTML>
<html>
<body>
<input type="color" list="colors">
<datalist id="colors">
    <option>#0000FF</option>
    <option>#00FF00</option>
    <option>#FF0000</option>
</datalist>
</body>
</html>
```

```
<!DOCTYPE HTML>
<html>
<body>
<label for="exam"> Elije una fecha para hacer el examen </label>
<input type="date"
    id="exam"
    value="2022-04-20"
    min="2021-09-01"
    max="2022-06-30"
    step="7"
/>
</body>
</html>
```



- Existen varias opciones para fechas, meses, horas, semanas, etc.

```
1   Selecciona una hora:
2   <input type="time">
```

Selecciona una hora: 01 : 58

```
<br />
Fecha y hora de la cita:
<input type="datetime-local">
```

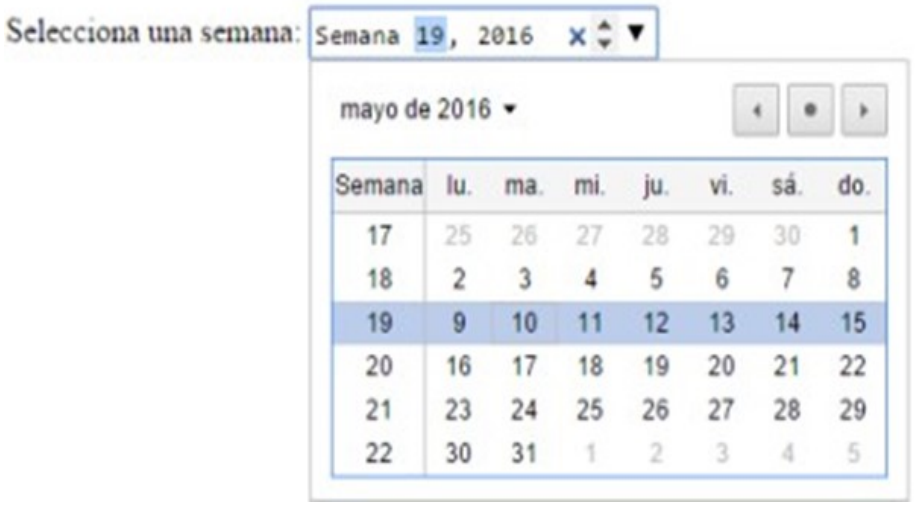
Fecha y hora de la cita 2012-07-2 12:00 UTC

Julio 2012

| Lun | Mar | Mie | Jue | Vie | Sab | Dom |
|---|---|---|---|---|---|---|
| 25 | 26 | 27 | 28 | 29 | 30 | 1 |
| 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 16 | 17 | 18 | 19 | 20 | 21 | 22 |
| 23 | 24 | 25 | 26 | 27 | 28 | 29 |
| 30 | 31 | 1 | 2 | 3 | 4 | 5 |

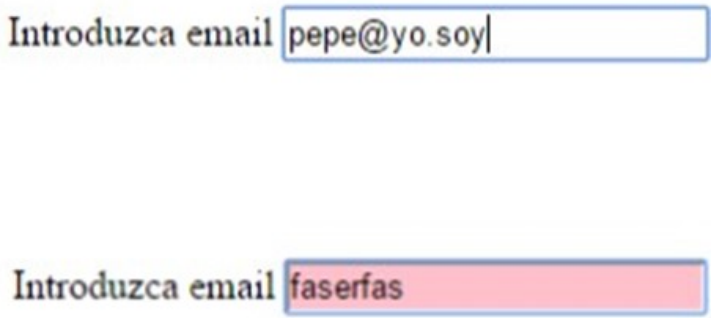
Hoy

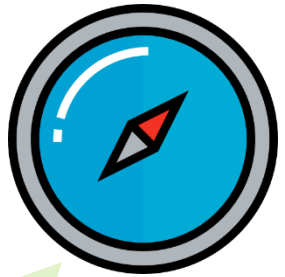
```
1  Selecciona una semana:
2  <input type="week">
```



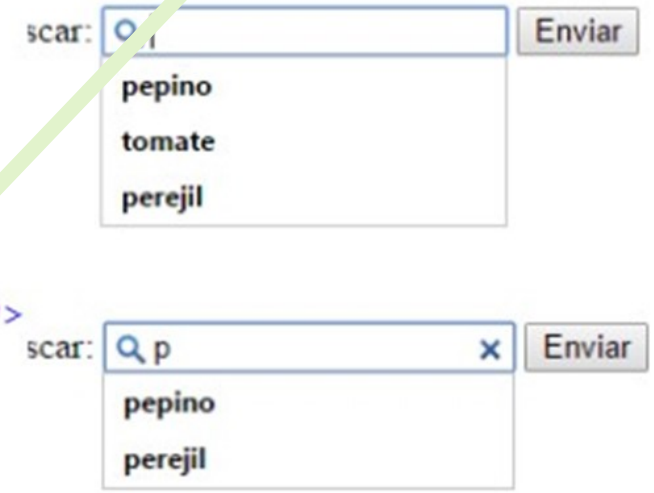

```
1  Selecciona un mes:
2  <input type="month">
```

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Ejemplo input type=email</title>
        <style>
            #email:invalid {
              background-color:pink;
            }
        </style>
    </head>
    <body>
        <label for="email">Introduzca email </label>
        <input type="email" id="email">
    </body>
</html>
```





```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Ejemplo input type=search</title>
    </head>
    <body>
        <form action="#">
            <label for="search">Buscar: </label>
            <input type=search id="search" results="2">

            <input type="submit">
        </form>
    </body>
</html>
```



```html
<!DOCTYPE html>
<html>
    <head>
        <title>Ejemplo de input type=number</title>
        <style>
        #number:invalid {
            background-color:pink;
        }
        #number:valid {
            background-color:lightGreen;
        }
        </style>
    </head>
    <body>
        <label for="number">Cantidad entre 0 y 500,
        debe ser multiplo de 5</label> <br/>
        <input type="number" id="number"
            value="25"
            min="0" max="500"
            step="5"
        />
    </body>
</html>
```

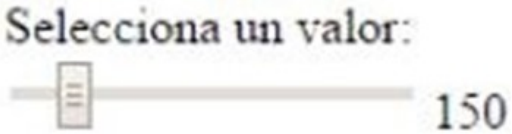
Cantidad entre 0 y 500, debe ser multiplo de 5

22

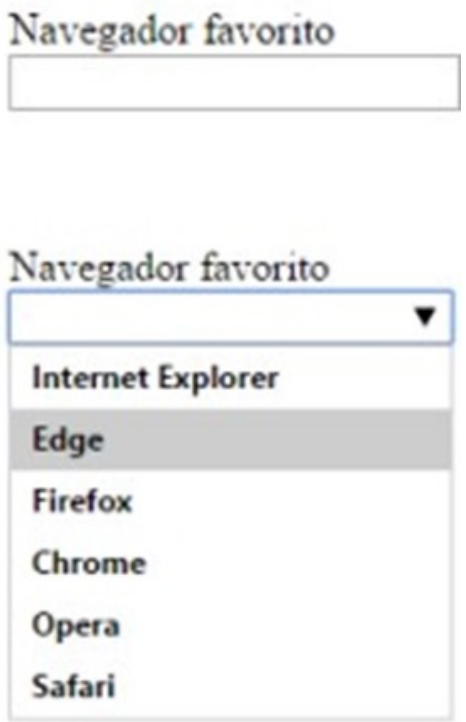
Cantidad entre 0 y 500, debe ser multiplo de 5

27

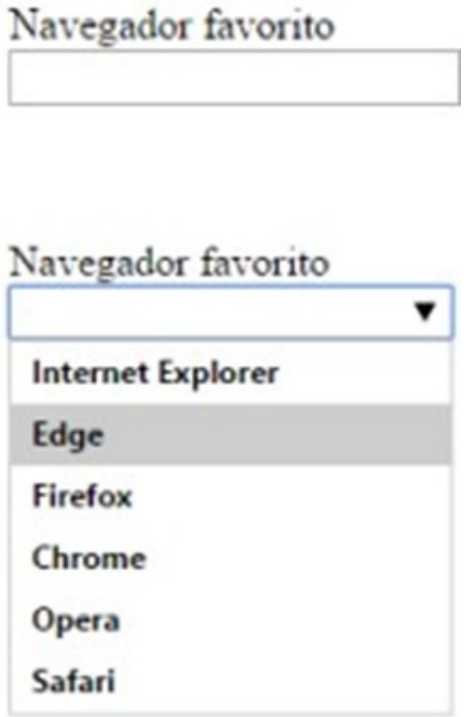
```html
<!DOCTYPE html>
<html>
    <head>
        <script>
            function printValue(sliderID, textbox) {
                var x = document.getElementById(textbox);
                var y = document.getElementById(sliderID);
                x.value = y.value;
            }
        </script>
    </head>
    <body onload="printValue('slider1','textbox')">
        <label for="slider1">Selecciona un valor:</label>
        <input id="slider1" type="range"
            min="100" max="500" step="10" value="150"
            oninput="printValue('slider1','textbox')"/>
        <output id="textbox"></output>
    </body>
</html>
```



```html
<!DOCTYPE html>
<html>
    <head>
        <title>Ejemplo de input list="..."</title>
    </head>
    <body>
        <label for="mybrowser">Navegador favorito</label>
        <input list="browsers" id="mybrowser" />

        <datalist id="browsers">
            <option value="Internet Explorer">
            <option value="Edge">
            <option value="Firefox">
            <option value="Chrome">
            <option value="Opera">
            <option value="Safari">
        </datalist>
    </body>
</html>
```



```html
<!DOCTYPE html>
<html>
    <head>
        <title>Ejemplo de input list="..."</title>
    </head>
    <body>
        <label for="mybrowser">Navegador favorito</label>
        <input list="browsers" id="mybrowser" />

        <datalist id="browsers">
            <option value="Internet Explorer">
            <option value="Edge">
            <option value="Firefox">
            <option value="Chrome">
            <option value="Opera">
            <option value="Safari">
        </datalist>
    </body>
</html>
```



# MULTIMEDIA

No es posible usar **<video>** con videos de algunos sitios webs como Youtube, Dailymotion, Vimeo, …

→ ¿POR QUÉ? Por la promoción y anuncios que se muestran en estos sitios.

```
<video width="320" height="240">
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
  Your browser does not support the video tag.
</video>
```

- En algunos sitios se proporciona el código fuente a añadir en la página web para incrustar el video.

Para poner videos también podemos usar **<iframe>.**

```
<iframe width="560" height="315"
src="https://www.youtube.com/embed/ZH1XOsv8Oyo" frameborder="0"
allowfullscreen></iframe>
```

```
<video controls preload="none">
   <source src="movie.mp4" type="video/mp4">
   <source src="movie.ogg" type="video/ogg">
   Your browser does not support the video tag.
</video>
```

Algunos atributos interesantes de este elemento son:

- **preload = “none”**
  Si el video no debe comenzar a cargarse al entrar en la web.
- **autoplay**
  Si el video comienza a reproducirse en cuanto se ha cargado el buffer suficiente.
- **poster = “url”**
  Elimina el cuadro negro inicial del video y lo sustituye por una imagen.
- **Loop**
  Si se llega al final del video se repetirá en modo bucle.

El elemento **<audio>** se utiliza para reproducir audio incrustado o audio en streaming.

- Es muy similar al elemento <video>.

```
<audio controls>
  <source src="horse.ogg" type="audio/ogg">
  <source src="horse.mp3" type="audio/mpeg">
  Your browser does not support the audio tag.
</audio>
```

El elemento **<track>** se utiliza para incluir subtítulos y sonidos de ruidos como fuego, lluvia…

- Es usado como hijo de los elementos <audio> y <video>.
- Las pistas están en formato WebVTT (.vtt) - pistas de texto de video web.
- Los famosos subtítulos en formato .srt no son soportados.
- Se pueden incluir varios elementos <track> con subtítulos en varios idiomas.
- El atributo default indica que los subtítulos se activan por defecto al reproducirse el video.

```
1  <video width="320" height="240" controls>
2    <source src="forrest_gump.mp4" type="video/mp4">
3    <source src="forrest_gump.ogg" type="video/ogg">
4    <track src="subtitles_en.vtt" kind="subtitles"
5      srclang="en" label="English">
6    <track src="subtitles_no.vtt" kind="subtitles"
7      srclang="no" label="Norwegian">
8  </video>
```

# GRÁFICOS

## Existen dos formas de representar gráficos en HTML5:

- Elemento **<canvas>.**
- Elemento **<svg>.**

Ambos tienen ventajas y desventajas. Se deben elegir en función del uso que se le vaya a dar en la web.

| Elemento CANVAS | Elemento SVG |
|---|---|
| • Orientado a **pixel**.<br>• Se utilizan scripts en JavaScript para poder realizar dibujos en el lienzo.<br>• Es más óptimo en animaciones muy dinámicas y complejas. | • Orientado a gráficos **vectoriales**.<br>• El dibujo se adapta al tamaño, de forma que al ampliarlo o reducirlo no se produce pérdida de calidad ni se ve pixelado.<br>• Es adecuado para impresión de gráficos de gran tamaño. |

| Elemento CANVAS | Elemento SVG |
|---|---|
| • Orientado a pixel.<br>• Se utilizan scripts en JavaScript para poder realizar dibujos en el lienzo.<br>• Es más óptimo en animaciones muy dinámicas y complejas. | • Orientado a gráficos vectoriales.<br>• El dibujo se adapta al tamaño, de forma que al ampliarlo o reducirlo no se produce pérdida de calidad ni se ve pixelado.<br>• Es adecuado para impresión de gráficos de gran tamaño. |


**Scalable Vector Graphics (SVG)** es un formato de imagen que puede dibujar gráficos 2D en un navegador web.

- Está basado en XML.
- Soporta características como la interacción, transiciones y animaciones.
- SVG es un estándar creado por el W3C.

| Elemento CANVAS | Elemento SVG |
|---|---|
| • Orientado a pixel.<br>• Se utilizan scripts en JavaScript para poder realizar dibujos en el lienzo.<br>• Es más óptimo en animaciones muy dinámicas y complejas. | • Orientado a gráficos vectoriales.<br>• El dibujo se adapta al tamaño, de forma que al ampliarlo o reducirlo no se produce pérdida de calidad ni se ve pixelado.<br>• Es adecuado para impresión de gráficos de gran tamaño. |

**SVG** introduce una variedad de formas que pueden ser utilizadas en un amplio abanico de escenarios.

Además, Se pueden combinar formas para crear formas más complejas.

```
1  <svg height="200" width="400">
2      <rect x="100" y="50" rx="20" ry="20" width="250" height="100" fill="#1B043C" />
3      <rect x="100" y="50" width="200" height="100" fill="#1B043C" />
4      <circle cx="100" cy="100" r="50" fill="#472772" />
5      <text fill="#D7BFF3" font-size="28" font-family="Segoe UI Light" x="160"
6      y="108">CompanyName</text>
7  </svg>
```

| Elemento CANVAS | Elemento SVG |
|---|---|
| • Orientado a pixel.<br>• Se utilizan scripts en JavaScript para poder realizar dibujos en el lienzo.<br>• Es más óptimo en animaciones muy dinámicas y complejas. | • Orientado a gráficos vectoriales.<br>• El dibujo se adapta al tamaño, de forma que al ampliarlo o reducirlo no se produce pérdida de calidad ni se ve pixelado.<br>• Es adecuado para impresión de gráficos de gran tamaño. |

**EJEMPLO:**

SVG define un **circulo** con el centro en una coordenada, un radio determinado y lo colorea.

| Attribute | Description |
|---|---|
| cx & cy | These two attributes together define the coordinates for the center of the circle. By default, the center of the circle is *(0,0)* |
| r | This attribute specifies the radius of the circle |
| fill | This attribute defines the color used for the interior of the circle |
| stroke | This attribute defines the color used for the border of the circle |
| stroke-width | This attribute defines the width of the border of the circle |

```
<svg height="200" width="200">
    <circle
        cx="100" cy="100"
        r="50"
        fill="#6CBEE2"
    />
</svg>
```

## Elemento CANVAS

- Orientado a pixel.
- Se utilizan scripts en JavaScript para poder realizar dibujos en el lienzo.
- Es más óptimo en animaciones muy dinámicas y complejas.

## Elemento SVG

- Orientado a gráficos vectoriales.
- El dibujo se adapta al tamaño, de forma que al ampliarlo o reducirlo no se produce pérdida de calidad ni se ve pixelado.
- Es adecuado para impresión de gráficos de gran tamaño.

**EJEMPLO:**

SVG define un **cuadrado** con cuatro lados a partir de un punto definido (esquina superior izquierda), un ancho y un alto, y lo colorea.

| Attribute | Description |
|---|---|
| x & y | These two attributes together define the coordinates for the top-left of the rectangle. By default, the top-left of the rectangle is *(0,0)* |
| rx & ry | These two attributes specify the radius for rounded corners on the x or y axis |
| width | This attribute defines the width of the rectangle |
| height | This attribute defines the height of the rectangle |
| fill | This attribute defines the color used for the interior of the rectangle |
| stroke | This attribute defines the color used for the border of the rectangle |
| stroke-width | This attribute defines the width of the border of the rectangle |

```
1  <svg height="200" width="200">
2      <rect
3          x="0" y="0"
4          width="200" height="100"
5          fill="#15744C"
6      />
7  </svg>
```

| Elemento CANVAS | Elemento SVG |
|---|---|
| • Orientado a pixel.<br>• Se utilizan scripts en JavaScript para poder realizar dibujos en el lienzo.<br>• Es más óptimo en animaciones muy dinámicas y complejas. | • Orientado a gráficos vectoriales.<br>• El dibujo se adapta al tamaño, de forma que al ampliarlo o reducirlo no se produce pérdida de calidad ni se ve pixelado.<br>• Es adecuado para impresión de gráficos de gran tamaño. |

**EJEMPLO:**

SVG define un **texto** a partir de un punto definido (esquina inferior-izquierda), con una fuente y le pone color.

| Attribute | Description |
|---|---|
| x & y | These two attributes together define the coordinates for the bottom-left of the text. |
| fill | This attribute defines the color used for the text |
| font-size | This attribute defines the size of the text font |
| font-family | This attribute defines the text font |

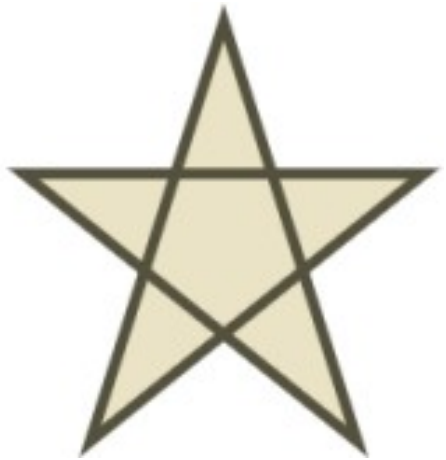
```
1  <svg height="200" width="200">
2      <text
3          font-family="Segoe UI Light"
4          font-size="28"
5          x="0" y="30"
6          fill="#B6652A">
7              Hello world!
8      </text>
9  </svg>
```

| Elemento CANVAS | Elemento SVG |
|---|---|
| • Orientado a pixel.<br>• Se utilizan scripts en JavaScript para poder realizar dibujos en el lienzo.<br>• Es más óptimo en animaciones muy dinámicas y complejas. | • Orientado a gráficos vectoriales.<br>• El dibujo se adapta al tamaño, de forma que al ampliarlo o reducirlo no se produce pérdida de calidad ni se ve pixelado.<br>• Es adecuado para impresión de gráficos de gran tamaño. |

**EJEMPLO:**

SVG define un **polígono** a partir de coordenadas en un plano 2D.

```
1  <svg width="300" height="300">
2      <polygon
3          points="100,10 40,198 190,78 10,78 160,198"
4          stroke="#54523F"
5          stroke-width="5"
6          fill="#EBE3C5"
7      />
8  </svg>
```

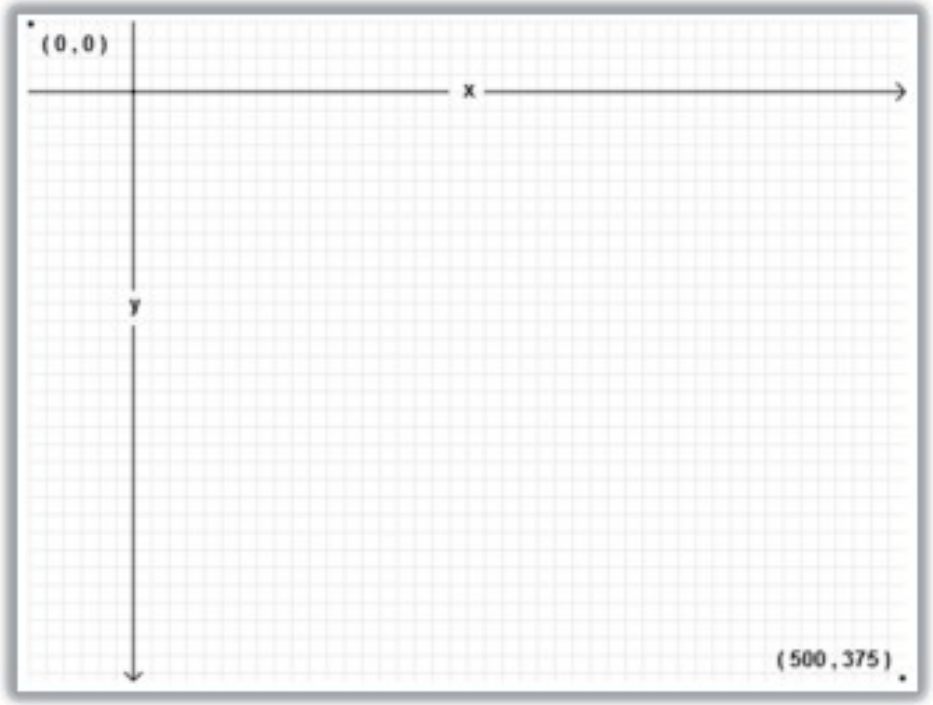


https://www.w3schools.com/graphics/svg_polygon.asp

| Elemento CANVAS | Elemento SVG |
| --- | --- |
| • Orientado a pixel.<br>• Se utilizan scripts en JavaScript para poder realizar dibujos en el lienzo.<br>• Es más óptimo en animaciones muy dinámicas y complejas. | • Orientado a gráficos vectoriales.<br>• El dibujo se adapta al tamaño, de forma que al ampliarlo o reducirlo no se produce pérdida de calidad ni se ve pixelado.<br>• Es adecuado para impresión de gráficos de gran tamaño. |

- El API de canvas soporta diferentes formas: líneas, rectángulos, eclipses, arcos, curvas, textos, imágenes…
- Canvas tiene soporte 2D/3D usando el API WebGL.
- Dispone de un canal alpha para añadir transparencia.
- Se usa principalmente para hacer animaciones, videos con efectos especiales, juegos, etc.

| Elemento CANVAS | Elemento SVG |
|---|---|
| • Orientado a pixel.<br>• Se utilizan scripts en JavaScript para poder realizar dibujos en el lienzo.<br>• Es más óptimo en animaciones muy dinámicas y complejas. | • Orientado a gráficos vectoriales.<br>• El dibujo se adapta al tamaño, de forma que al ampliarlo o reducirlo no se produce pérdida de calidad ni se ve pixelado.<br>• Es adecuado para impresión de gráficos de gran tamaño. |




El sistema de coordenadas usado para dibujar canvas es similar al de otras APIs de dibujo 2D. El punto (0,0) está situado en la esquina superior-izquierda del lienzo.

| Elemento CANVAS | Elemento SVG |
|---|---|
| • Orientado a pixel.<br>• Se utilizan scripts en JavaScript para poder realizar dibujos en el lienzo.<br>• Es más óptimo en animaciones muy dinámicas y complejas. | • Orientado a gráficos vectoriales.<br>• El dibujo se adapta al tamaño, de forma que al ampliarlo o reducirlo no se produce pérdida de calidad ni se ve pixelado.<br>• Es adecuado para impresión de gráficos de gran tamaño. |

## 1. Añadir un elemento <canvas> a la página web.

```
1. <canvas id="myCanvas" width="300" height="225">
2.     Fallback content that will be displayed in case the web browser
3.     does not support the canvas tag or in case scripting
4.     is disabled.
5. </canvas>
```

- No tendrá ningún efecto visual en la página web, ya que por defecto el lienzo es transparente.
- Se utiliza un mensaje de texto para mostrar al usuario en caso de que canvas no sea soportado por el navegador.

| Elemento CANVAS | Elemento SVG |
|---|---|
| • Orientado a pixel.<br>• Se utilizan scripts en JavaScript para poder realizar dibujos en el lienzo.<br>• Es más óptimo en animaciones muy dinámicas y complejas. | • Orientado a gráficos vectoriales.<br>• El dibujo se adapta al tamaño, de forma que al ampliarlo o reducirlo no se produce pérdida de calidad ni se ve pixelado.<br>• Es adecuado para impresión de gráficos de gran tamaño. |

## 2. Seleccionar el elemento **<canvas>** desde Javascript.

```
1. var canvas = document.getElementById("myCanvas");
```

```
1. var canvas = document.querySelector("#myCanvas");
```

- Se utilizará el atributo **id** del elemento canvas para obtener el elemento desde Javascript y poder trabajar con él.

| Elemento CANVAS | Elemento SVG |
|---|---|
| • Orientado a pixel.<br>• Se utilizan scripts en JavaScript para poder realizar dibujos en el lienzo.<br>• Es más óptimo en animaciones muy dinámicas y complejas. | • Orientado a gráficos vectoriales.<br>• El dibujo se adapta al tamaño, de forma que al ampliarlo o reducirlo no se produce pérdida de calidad ni se ve pixelado.<br>• Es adecuado para impresión de gráficos de gran tamaño. |

## 3. Obtener el contexto asociado.

- Cuando se tiene acceso al canvas, se obtiene el contexto.
- Existe contexto 2D y 3D.
- Gracias al contexto se pueden definir propiedades de dibujo

```
1. var ctx=canvas.getContext('2d');
```

```
1. ctx.fillStyle='red';
```

```
1. ctx.fillRect(0,0,80,100);
```

| Elemento CANVAS | Elemento SVG |
|---|---|
| • Orientado a pixel.<br>• Se utilizan scripts en JavaScript para poder realizar dibujos en el lienzo.<br>• Es más óptimo en animaciones muy dinámicas y complejas. | • Orientado a gráficos vectoriales.<br>• El dibujo se adapta al tamaño, de forma que al ampliarlo o reducirlo no se produce pérdida de calidad ni se ve pixelado.<br>• Es adecuado para impresión de gráficos de gran tamaño. |

```html
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Mi primer canvas</title>
        <script src="reglas.js"></script>
        <style>
            #myCanvas{
                border: 1px solid black;
            }
        </style>
    </head>
    <body onload="init();">
        <canvas id="myCanvas" width="200" height="200">
            Tu navegador no soporta canvas
        </canvas>
    </body>
</html>
```

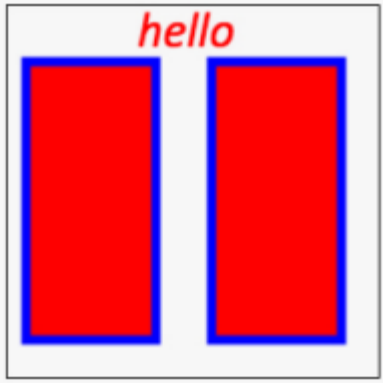
```javascript
var canvas, ctx;

function init() {
    // 1 - Se obtiene el canvas
    //canvas = document.getElementById('myCanvas');
    canvas = document.querySelector('#myCanvas');

    // 2 - Se obtiene el contexto
    ctx = canvas.getContext('2d');

    // 3 - dibujar
    dibujar();
}

function dibujar() {
    // dibujar un rectangulo rojo
    ctx.fillStyle='#FF0000';
    ctx.fillRect(10,10,180,180);
}
```

Algunas propiedades del contexto canvas son los siguientes:

- **fillStyle:**
  - Define un color, una textura o un gradiente para el **relleno** de las formas.
  - El color por defecto es negro.
- **fillRect(x, y, width, height):**
  - Dibuja un rectángulo relleno.
  - El relleno lo determina la última propiedad fillStyle.
- **strokeStyle**:
  - Define un color, una textura o un gradiente para el **borde** de las formas.
  - El color por defecto es negro.

Algunas propiedades del contexto canvas son los siguientes:

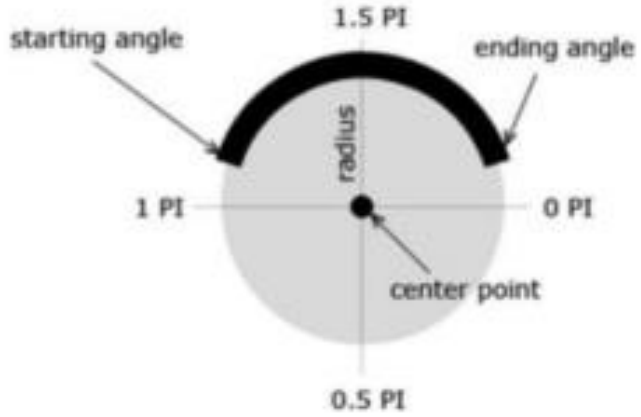
- **strokeRect (x,y,width, height:**
  - Dibuja el borde de un rectángulo.
  - El relleno del borde lo determina la última propiedad *strokeStyle* definida.
- **clearRect(x, y, width, height):**
  - Borra un rectángulo aplicando un color transparente.
- **font**:
  - Define la fuente y las propiedades a utilizar.
  - ctx.font = “italic 20pt Calibri”;
- **fillText(texto,x,y):**
  - Define el texto y su posición en el lienzo.



```
var canvas, ctx;

function init() {
    // 1 - Se obtiene el canvas
    //canvas = document.getElementById('myCanvas');
    canvas = document.querySelector('#myCanvas');

    // 2 - Se obtiene el contexto
    ctx = canvas.getContext('2d');

    // 3 - dibujar varias lineas
    dibujarLinea(0, 0, 100, 100);
    dibujarLinea(0, 50, 150, 200, 'red');
    dibujarLinea(10, 100, 100, 10, 'green', 10);

}
```

```
function dibujarLinea(x1, y1, x2, y2, color, width) {
    ctx.save();

    // modifica el color y el ancho de la linea
    // solo si estos parametros han sido definidos
    if(color)
      ctx.strokeStyle = color;
    if(width)
      ctx.lineWidth = width;

    // se crea la linea
    ctx.beginPath();
    ctx.moveTo(x1, y1);
    ctx.lineTo(x2, y2);
    ctx.stroke();

    ctx.restore();
}
```

```
var canvas, ctx;

function init() {
    // 1 - Se obtiene el canvas
    //canvas = document.getElementById('myCanvas');
    canvas = document.querySelector('#myCanvas');

    // 2 - Se obtiene el contexto
    ctx = canvas.getContext('2d');

    // 3 - dibujar circulo
    dibujarCirculo(75, 75, 50, 0, 2*Math.PI);
}
```

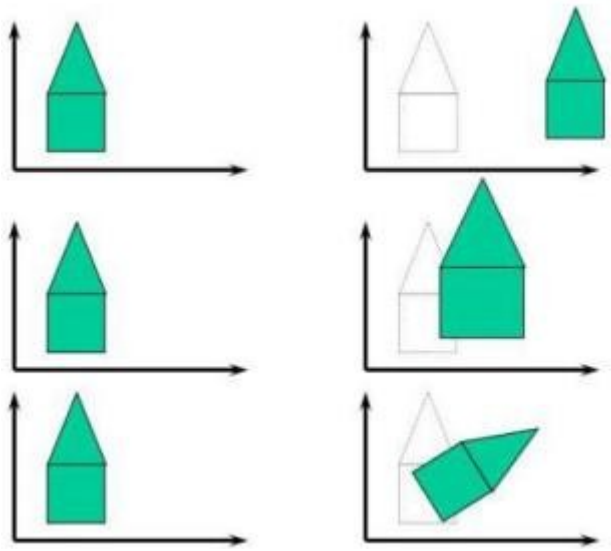



```
function dibujarCirculo(cx, cy, r, sa, ea) {
    ctx.save();

    // se define el circulo
    ctx.beginPath();
    ctx.arc(cx, cy, r, sa, ea);
    // se define y pinta el relleno
    ctx.fillStyle = "lightBlue";
    ctx.fill();
    // se define y pinta el borde
    ctx.lineWidth = 10;
    ctx.stroke();

    ctx.restore();
}
```

```
var canvas, ctx;

function init() {
    // 1 - Se obtiene el canvas
    //canvas = document.getElementById('myCanvas');
    canvas = document.querySelector('#myCanvas');

    // 2 - Se obtiene el contexto
    ctx = canvas.getContext('2d');

    // 3 - dibujar
    dibujar(0,0,"goku.png");

}
```

```
function dibujar(x, y, pathToImage) {

    var imageObj = new Image();

    // llama a la funcion imageObj.onload de forma asincrona
    imageObj.src = pathToImage;

    // esta funcion es llamada por imageObj.src
    imageObj.onload = function() {
        // se dibuja la imagen cuando se ha terminado de cargar
        ctx.drawImage(imageObj, 0, 0);
    };
}
```

Con *canvas* también podemos aplicar transformaciones a las formas:



Translación: ctx.translate(x,y)

Especificamos la coordenada a la que se tiene que transladar la forma.

- Rotación:

  ctx.rotate(angle)

Sirve para rotar una forma especificando un ángulo de rotación (expresado en radianes).

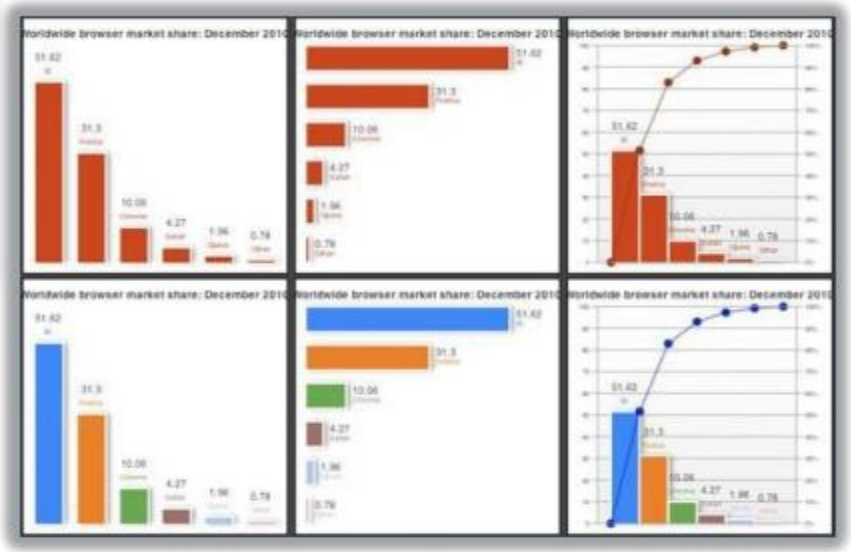
- Escalado:

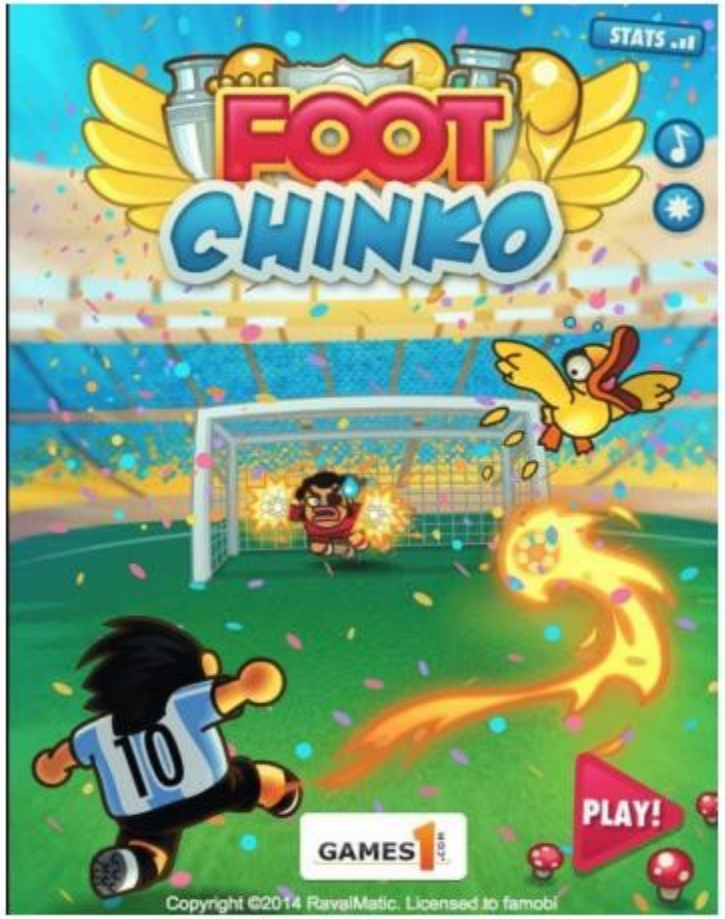
  ctx.scale(x,y)

Sirve para aumentar o disminuir el tamaño de la forma. Una escala (1,1) dejaría la forma con el tamaño actual.

# ANIMACIONES

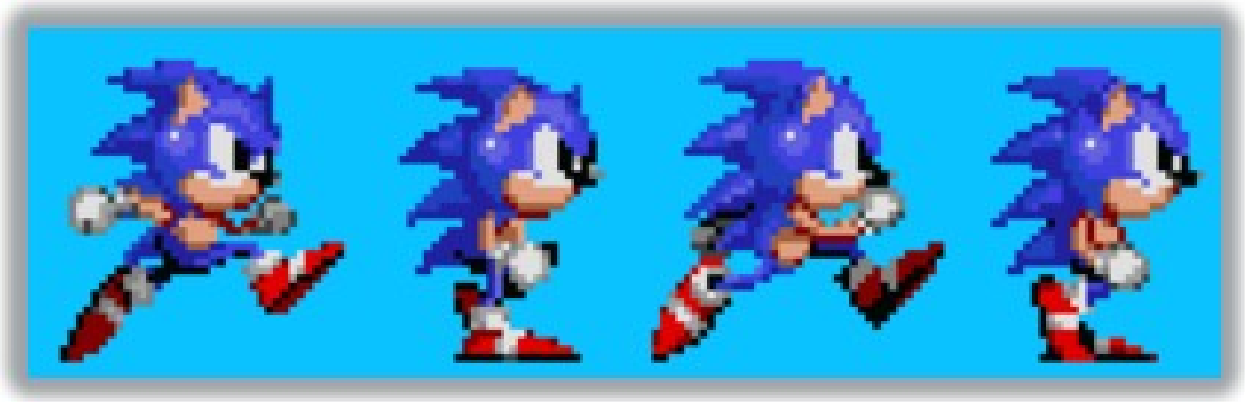

Love HTML5? Love Canvas!
canvasdemos




STATS
FOOT
CHINKO
10
GAMES
PLAY!
Copyright ©2014 RavalMatic. Licensed to famobi

# Cualquier animación *canvas* realiza estos pasos:

1. **Limpiar el contexto** de *canvas*.
   - Se puede usar el método **ctx.clearRect(0,0,canvasWith,canvasHeight);**
2. Dibujar las formas de la animación.
3. Mover las formas (traslación, rotación, escalado), cambiar de color, etc…
4. Volver al paso 1.



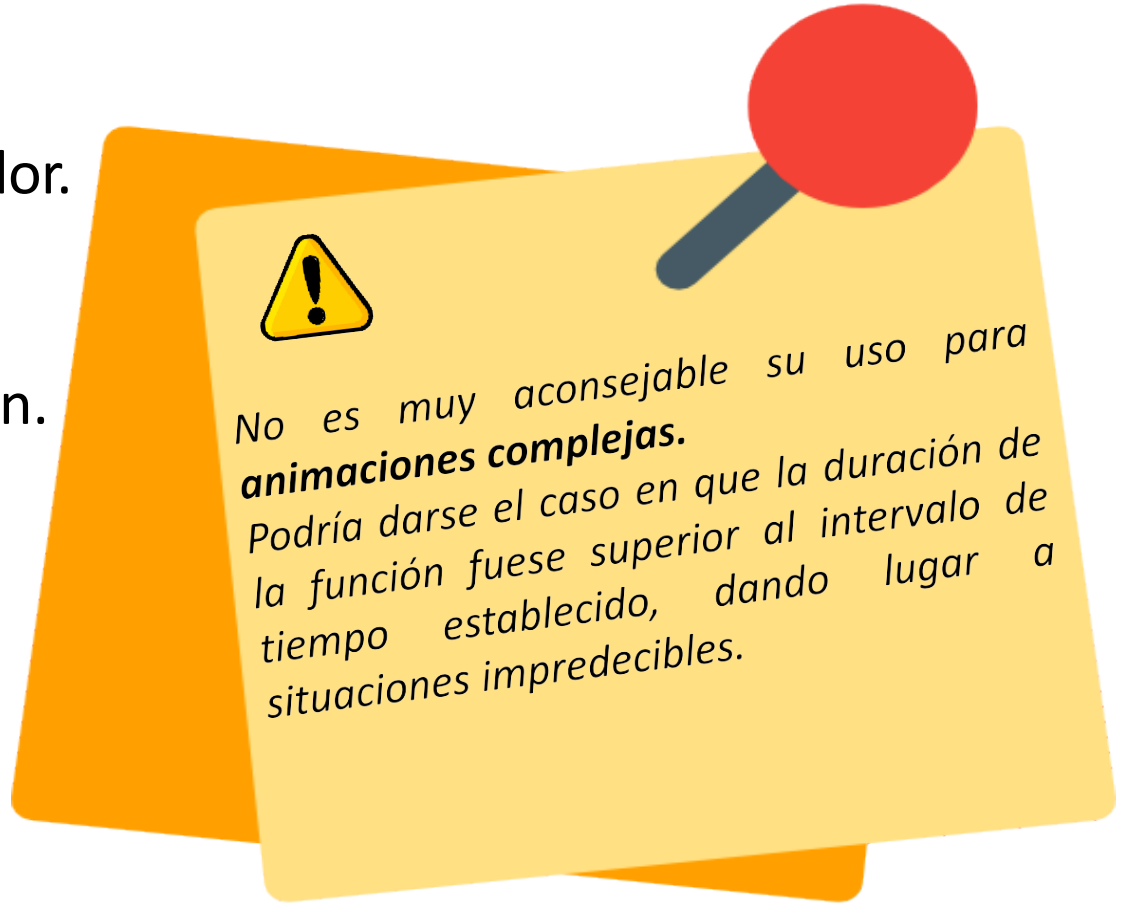
## Id=setInterval(función,ms)

→ Esta función llama a otra función cada intervalo de tiempo especificado en milisegundos.

Después de ejecutarse, devuelve un identificador.

## clearInterval(id)

→ Esta función se usa para detener una animación.



## Id=setTimeout(función,ms)

→ Nos permite ejecutar una función una vez, pasado un intervalo de tiempo dado.
Después de ejecutarse, devuelve un identificador.

## clearTimeout(id)

→ Esta función se usa para detener una animación.

## requestAnimationFrame(función)

→ Sustituye a setInterval().

**Ver ejemplo de requestAnimationFrame**

| Ventajas sobre setInterval() y setTimeout() |
| --- |
| El intervalo de ejecución es mucho más **preciso**. |
| Se pueden ejecutar **múltiples animaciones** sin sobrecargar el procesador. |
| Existen más parámetros para configurar intervalos de alta precisión en ciertas animaciones que lo requieran. |
| **Ahorro de batería** en dispositivos móviles gracias a la optimización en la CPU/GPU. |

# ETIQUETAS SEMANTICAS HTML 5

HTML 5 fue publicado el 28 de octubre de 2014 como la 5ª versión del formato usado para construir páginas webs y aplicaciones.

Algunas de las mejores que introdujeron fueron las siguientes:

- Capacidad para construir aplicaciones basadas en la web.
- Soporte para video y audio.
- Gráficos y efectos de estilo.
- Completo conjunto de APIs.
- Adaptable a cualquier dispositivo (escritorio, móvil, Tablet, TV…)
- Se añade semántica a la web.
- Se estructura la web para adaptarla a los nuevos diseños.

# “HTML 5 busca la simplicidad”

- Hay una **nueva definición de Doctype**.
- Algunos **atributos** son **opcionales** (pe. el atributo ***type*** en los elementos ***<link>*** *o* ***<script>***).
- Restricciones de sintaxis menos estrictas.
  - Se pueden **omitir las comillas** en la mayoría de los atributos.
  - **No es Key Sensitive**, puede mezclar mayúsculas y minúsculas.
- Se han incorporado **nuevos elementos de estructura** demandados por los desarrolladores.

## HTML 4.01

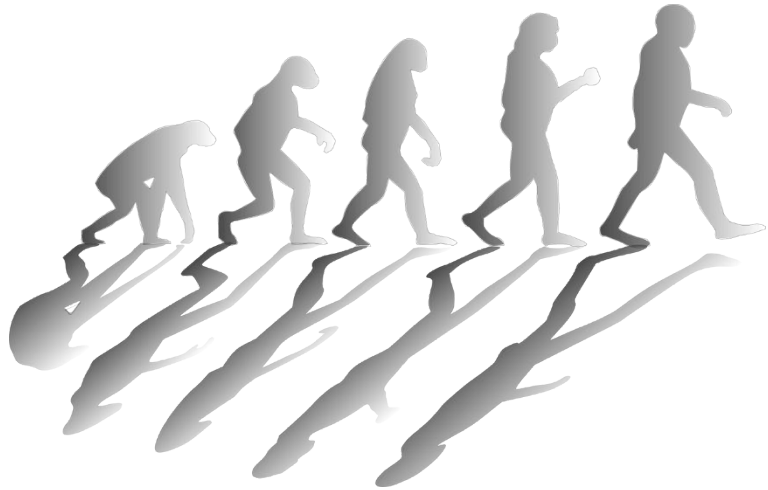
```
1. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/strict.dtd">
2. <html lang="en">
3. <head>
4.     <meta http-equiv="content-type"
   content="text/html" charset="utf-8">
5.     <title>title</title>
6.     <link rel="stylesheet" type="text/css"
   href="style.css">
7.     <script type="text/javascript" src="script.js">
   </script>
8. </head>
9. <body>
10. ...
11. </body>
12. </html>
```

## HTML 5

```
1. <!DOCTYPE html>
2. <html lang="en">
3. <head>
4.    <meta charset="utf-8">
5.    <title>Page Title</title>
6.    <link rel="stylesheet" href="style.css">
7.    <script src="script.js"></script>
8. </head>
9. <body>
10. ... <!-- The rest is content -->
11. </body>
12. </html>
```

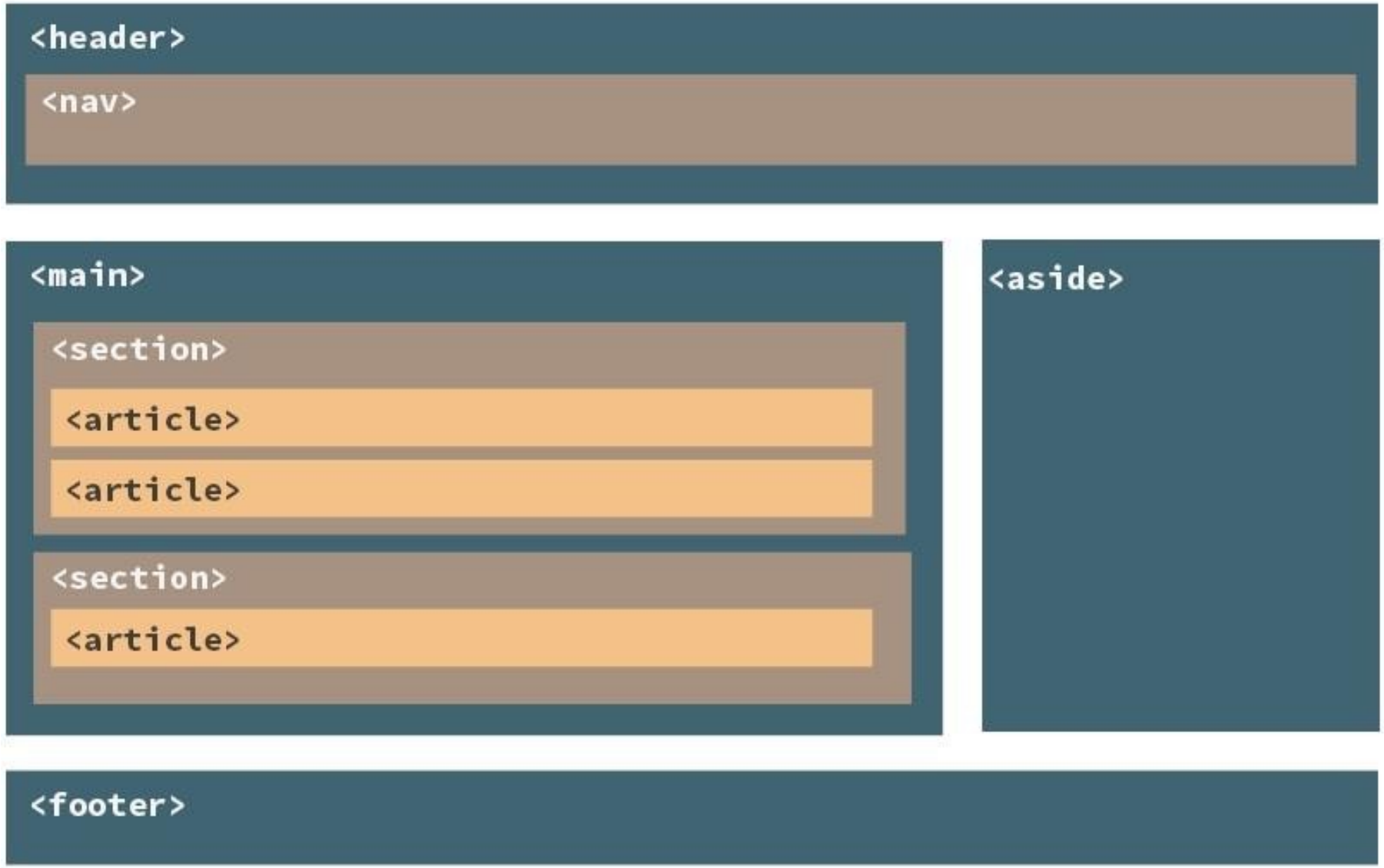
- La **web ha ido evolucionando** y los típicos elementos como listas, párrafos, tablas… son muy básicos
  para los diseños actuales.
  → Se **demandan nuevas opciones** como menús de navegación, paneles de fichas, encabezados, pies de páginas, etc.

- Muchos de estos elementos, ya se podían implementar con anteriores versiones de HTML, pero usando
  cantidades inmensas de código CSS y JavaScript, llevando a los siguientes inconvenientes:
  - ***id*** y ***class*** (usados para referenciar los elementos de HTML en JavaScript o CSS) varían de un desarrollador a otro, de un país a otro… No hay un estándar.
  - Pueden haber ***id*** y ***class*** que solapen entre ellos.
  - Librerías JavaScript cada vez más pesadas y difíciles de mantener.



# Entre los nuevos elementos para HTML5 destacan los siguientes:

| | |
|---|---|
| **<header>** | Introducción a “elementos de seccionado”: Un artículo, una sección, el documento entero. |
| **<footer>** | Contiene el pie del sitio web, de un largo *<articule>,* o de una larga *<section>.* |
| **<nav>** | Sección que contiene los links de navegación principales (dentro del documento o a otras páginas). |
| **<article>** | Contenido independiente, el cuál puede ser extraído de la web sin que penalice su comprensión. |
| **<section>** | Sección genérica usada para agrupar diferentes artículos para diferentes propósitos. |
| ***<time>*** | Usado para marcar horas y fechas. |
| **<aside>** | Sección con contenido no necesariamente relacionado con el tema principal, pero que puede añadir información adicional. |
| ***<figure>*** ***<figcaption>*** | Usado para encapsular una figura (imagen, video, etc) como un único ítem que contiene una descripción asociada. |
| ***<main>*** | Representa el contenido principal del cuerpo del documento. Se expone el contenido que directamente está relacionado con el tema central de la web.<br>Solamente puede haber un <**main**> en el documento. |


<header>
<nav>
<main>
<section>
<article>
<article>
<section>
<article>
<aside>
<footer>

No está claro si un elemento ***<section>*** puede contener uno o varios **<article>** o si un elemento **<article>** puede contener uno o varios ***<section>.***

**Ambas opciones son correctas.**

```
<article>
    <section>
        <h2>Introducción</h2>
    </section>
    <section>
        <h2>Mi viaje a India</h2>
    </section>
    <section>
        <h2>Vuelta a España</h2>
    </section>
</article>
```

```
<section>
    <article>
        <h2>¡HTML 5 ha llegado!</h2>
    </article>
    <article>
        <h2>Windows 10 está de moda</h2>
    </article>
    <article>
        <h2>Nueva release de Ubuntu</h2>
    </article>
</section>
```

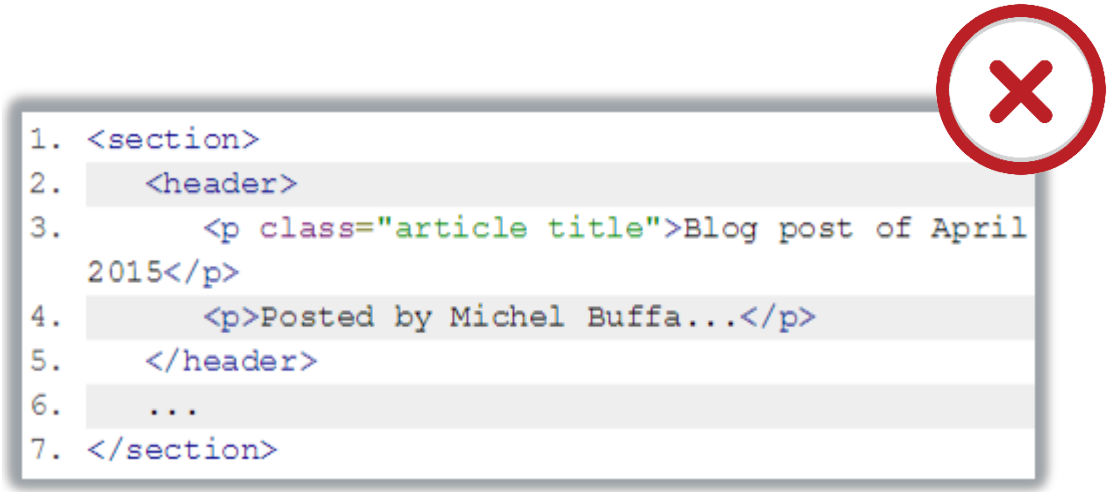
Los elementos ***<section>, <article>, <nav>, <header>, <footer>*** **y** ***<aside>*** son llamados **elementos de seccionado**, porque dividen el documento en partes semánticas.

La especificación HTML5 indica que cada **elemento de seleccionado** debe tener un **encabezado asociado** (<h1>,<h2>, <h3>…).

```
1  <body>
2      <h1>Título del documento</h1>
3      ...
4      <section>
5          <h1>Título de la sección</h1>
6          ...
7      </section>
8  </body>
```

```
1. <section>
2.     <h1>Blog post of April 2015</h1>
3.     ...
4. </section>
```

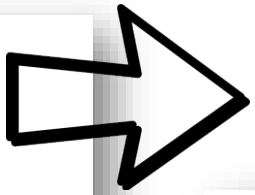
```
1. <section>
2.    <header>
3.       <h1>Blog post of April 2015</h1>
4.       <p>Posted by Michel Buffa...</p>
5.    </header>
6. ...
7. </section>
```



```
1. <section>
2.    <header>
3.       <p class="article title">Blog post of April
   2015</p>
4.       <p>Posted by Michel Buffa...</p>
5.    </header>
6.    ...
7. </section>
```

Debes intentar usar los **elementos de seleccionado** en lugar de solo los de encabezado.

```
1  <body>
2      <h1>Apples</h1>
3      <p>Apples are fruit.</p>
4      <h2>Taste</h2>
5      <p>They taste lovely.</p>
6      <h3>Sweet</h3>
7      <p>Red apples are sweeter than green ones.</p>
8      <h2>Color</h2>
9      <p>Apples come in various colors.</p>
10 </body>
```



```
1.  <body>
2.  <h1>Apples</h1>
3.  <p>Apples are fruit.</p>
4.  <section>
5.      <h2>Taste</h2>
6.      <p>They taste lovely.</p>
7.      <section>
8.          <h3>Sweet</h3>
9.          <p>Red apples are sweeter than green ones.</p>
10.     </section>
11. </section>
12. <section>
13.     <h2>Color</h2>
14.     <p>Apples come in various colors.</p>
15. </section>
16. </body>
```

*Obviamente, ambos son correctos. Pero el de la derecha, es más correcto según la norma HTML5.*

Debes intentar usar los **elementos de seleccionado** en lugar de solo los de encabezado.

*Esta forma de codificar es mucho más fácil de mantener que el anterior.*

***¡ATENCIÓN!*** *Puede que aún algunos navegadores no implementen el algoritmo "outline" y traten todos los <h1> como elementos principales del documento.*

```
1.  <body>
2.  <h1>Apples</h1>
3.  <p>Apples are fruit.</p>
4.  <section>
5.      <h2>Taste</h2>
6.      <p>They taste lovely.</p>
7.      <section>
8.          <h3>Sweet</h3>
9.          <p>Red apples are sweeter than green ones.</p>
10.     </section>
11. </section>
12. <section>
13.     <h2>Color</h2>
14.     <p>Apples come in various colors.</p>
15. </section>
16. </body>
```

# ¿QUÉ ES EL ALGORITMO OUTLINE?

En **HTML4** la jerarquía de encabezados de una página viene dada por las etiquetas **h1, h2, h3, h4, h5 y h6.**

En **HTML5 es distinto**, el nivel que ocupa cada encabezado en la jerarquía no viene dado principalmente por el número que acompaña a cada "h", sino por el **lugar que ocupa dentro del documento** cada encabezado y las etiquetas que los envuelven.

Esto se logra con los **elementos de seccionado.**

El elemento ***<main>*** sirve para identificar el **contenido principal** del documento.
Es muy útil para que la tecnología de accesibilidad sepa cuál es el contenido principal de la web.

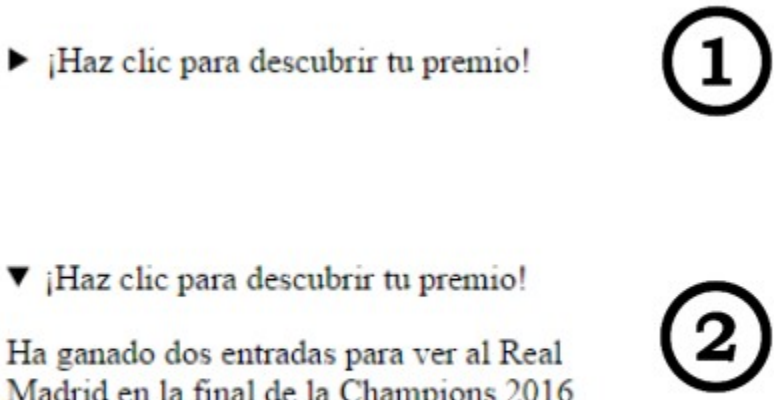
Tiene ciertas **restricciones** a tener en cuenta:

- Solo puede existir un elemento *<main>* en el documento.
- No puede ser descendiente de los **elementos de seleccionado *<article>, <nav>, <header>, <footer> o <aside>***.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Mi página web</title>
  </head>
  <body>
    <header>
      <nav>
        <ul>
          <li><>Inicio</a></li>
        </ul>
      </nav>
    </header>
    <main>
      <h1>Bienvenido a mi página web</h1>
      <p>Este es el contenido principal de mi página web</p>
    </main>
    <footer>
      <p>Copyright © 2021 Mi Página Web</p>
    </footer>
  </body>
</html>
```

Los elementos ***<details>*** y ***<summary>*** sirven para mostrar una **zona plegable** en el documento HTML, siendo **<summary>** quien contiene el título de la zona desplegable.

```
1   <!DOCTYPE html>
2   <html>
3   <head></head>
4   <body>
5       <details>
6           <summary>¡Haz clic para descubrir tu premio!</summary>
7           <p>Ha ganado dos entradas para ver al Real Madrid en
8           la final de la Champions 2016</p>
9       </details>
10  </body>
11  </html>
```



La etiqueta **<hgroup>** pretende contener un grupo de etiquetas **<hx>** (al menos dos ). Es una formula derivada de la etiqueta **<header>.**

```
 8  <body>
 9      <header>
10          <hgroup>
11              <h1>Primer encabezado</h1>
12              <h2>Segundo encabezado</h2>
13          </hgroup>
14      </header>
15  </body>
16  </html>
```

La etiqueta ***<progress>*** indica la progresión de una tarea, como las barras de progreso. Solo tiene sentido si se utiliza junto javascript. Los atributos de esta etiqueta son:

value: El valor del progreso
max: Valor máximo posible.

```
8  <body>
9      <p>Descarga:</p>
10     <progress value="70" max="100"></progress>
11 </body>
12 </html>
```

La etiqueta ***<figure>*** puede usarse para agrupar elementos tales como imágenes, videos o incluso texto que forma parte de una ilustración del contenido principal.

La etiqueta ***<figcaption>*** usada de forma conjunta con la etiqueta anterior, proporciona una leyenda a los elementos así agrupados.

```
8   <body>
9       <figure>
10          <figcaption>Personajes de Dragon Ball Z</figcaption>
11          <img src="fase0.jpeg" alt="" width="300px" height="300px">
12          <img src="fase3.jpg" alt="" width="300px" height="300px">
13
14      </figure>
15  </body>
16  </html>
```

El elemento **<mark>** sirve para subrayar texto como lo haría un rotulador.

```
1  <!DOCTYPE html>
2  <html>
3  <head></head>
4  <body>
5      <p>Aprender <mark>HTML 5</mark> es muy útil.</p>
6  </body>
7  </html>
```

# ¿QUÉ ES MICRODATA?

El objetivo principal de microdata es la **optimización de los motores de búsqueda**.

- Las máquinas lectoras son capaces de entender el contenido de la web y el tema que se trata.
- Existen tres opciones de proporcionar contenido leíble a las máquinas:
  - **Microdata**.
  - HTML+RDFa.
  - Microformats.
- Esta información no es visible por los humanos, solo por las máquinas. Es pura información semántica.

Añadir microdata a una página HTML requiere solo de tres atributos:

- *Itemscope.*
- *Itemtype.*
- *Itemprop.*

## Itemscope

**Crea** un objeto (***item***) para el cuál se definen unas propiedades.

## Itemtype

Es usado para **especificar el vocabulario** válido (el más usado es *schema.org*) que describe al ***item*** y sus propiedades en ese contexto.

## Itemprop

Es usado para **añadir propiedades**, a través de pares nombre-valor, a un ***item***.
Cada par nombre-valor es una ***propiedad,*** por lo tanto, un conjunto de UNA o MÁS propiedades conformar un ***item***.

- *Itemscope.*
- *Itemtype.*
- *Itemprop.*

```
<div itemscope itemtype="https://schema.org/Movie">
  <h1 itemprop="name">Avatar</h1>
  <span>Director: <span itemprop="director">James Cameron</span> (born August 16,
    1954)</span>
  <span itemprop="genre">Science fiction</span>
  <a href="https://youtu.be/0AY1XIkX7bY" itemprop="trailer">Trailer</a>
</div>
```

El vocabulario más usado es Schema.org… Este vocabulario define un conjunto de nombres de tipos y **propiedades**.

*Por ejemplo,* https://schema.org/MusicEvent es el espacio de nombres para conciertos. Incluye las propiedades como *startDate (fecha de inicio)*, *location (localización),* entre muchos otros.

Es importante tener en cuenta que los vocabularios pueden también ser creados por el programador web. A continuación, una lista de enlaces a los vocabularios más usados:

https://schema.org/CreativeWork
https://schema.org/Book
https://schema.org/Movie
https://schema.org/MusicRecording
https://schema.org/TVSeries
https://schema.org/Event
https://schema.org/Person
https://schema.org/Organization
https://schema.org/Product
https://schema.org/Place
https://schema.org/Restaurant
https://schema.org/Review
https://schema.org/Offer
https://schema.org/AggregateOffer
https://developer.mozilla.org/es/docs/Web/HTML/Microdata

# Muchas gracias por vuestra atención