

FEUD

Fundación de egresados U. Distrital
Construyendo Profesionales



Somos el Centro de Entrenamiento Autorizado por marcas representativas en Gobierno TI y empresa, con el portafolio más amplio en Latinoamérica:



Todas las marcas, nombres comerciales, marcas de servicios y logotipos a los que se hace referencia en el presente documento pertenecen a sus respectivas empresas. ITIL® es una marca registrada de AXELOS Limited. El Swirl logo™ es una marca de AXELOS Limited. PMP, PMI, PMI-RMP, PMI-ACP, PMI-SP, PgMP, CAPM, REP Logo son marcas registradas del Project Management Institute.

FEUD
Fundación de egresados U. Distrital
Construyendo Profesionales



HTML 5 + CSS3 + JavaScript

Diplomado Desarrollo de Software para móviles

Diego Alberto Rincón Yáñez MSc.

Julio Cesar Galeano Garcia

HTML5

THE HISTORY OF HTML5



2004

"WHAT" Working Group is born

Founded in 2004, The WHATWG, with members from Apple, the Mozilla Foundation, and Opera Software, sets out to develop HTML5.

Oct.
2006



World Wide Web Consortium (W3C) announces it will work with "WHAT" Working Group

W3C decides to stop working on XHTML and instead begins collaborating with "WHAT" Working Group to evolve HTML as a technology.

2008

First version of HTML5 is published

The first draft of HTML5, written by Ian Hickson, is introduced but changes are still coming. Experts say that HTML5 is a continually evolving technology that will never be absolutely "finished."

2008



Firefox 3 becomes HTML5 compatible

Firefox 3 takes steps to allow HTML5 to be viewed on the browser. Chrome, Safari and eventually IE will follow suit.



Jan.
2010



Safari and eventually IE will follow suit.

Jan.
2010



YouTube offers HTML5 Video Player

New player can only be activated through TestTube. A better video player will arrive in July 2010.

April
2010

Steve Jobs "trashes" Flash in an open letter

Jobs explains why Flash will never be allowed on Apple's smart devices. This triggers many companies to begin pursuing HTML5.



"Flash was designed for PCs using mice, not for touch screens using fingers."

May
2010

Scribd documents switch to HTML5

The online document sharing site, Scribd, switches to HTML5. This creates a better UI for users who are reading documents on tablets.

Scribd

Aug.
2010



Arcade Fire's HTML5-based interactive film is a hit

A year after its release, Arcade Fire's interactive film "The Wilderness Downtown" wins the Grand Prix award at the Cannes advertising awards in the Cyber category.



Dec.
2010

Chrome Web Store opens

Chrome opens its web store in HTML5, making non-Apple web apps easy to buy on tablets.

March
2011



March
2011

Disney

Disney buys HTML5 gaming start-up

Disney buys 'Rocket Pack', a Helsinki-based HTML5 gaming engine start-up. Its intention is clear: break the app-store monopoly and build games straight into the web using HTML5.



July
2011

Pandora begins moving to HTML5

Pandora begins making the switch to an HTML5 audio player. It is reviewed as being 'less clunky' than the Flash player: easier to load and faster.



Aug.
2011

kindle amazon.com

Amazon creates Kindle Cloud Reader

Amazon creates a new web-based version of the Kindle eBook reader app. The new HTML5 version allows customers to access their content offline directly from their browser.



Aug.
2011

Twitter rolls out new HTML5 version for iPad

Twitter revamps its iPad presence with a new HTML5-heavy version.



Sept.
2011

34% of top 100 sites use HTML5

As of Sept 2011, 34% of Alexa's Top-100 trafficked websites are using HTML5.

TOP100
SITES



Sept.
2011

Boston Globe

January
2013

HTML5 adoption
expected to boom as
over one billion
(1,000,000,000)
HTML5-compatible
smartphones will be
sold by this time.

Based on research by Strategy
Analytics whose predictions are
supported by other research
agencies



craigslist

**Craigslist
isn't currently
HTML5**

Craigslist isn't popular
for its beautiful UI
design but it has
announced that it is
hiring a designer for an
HTML5 redesign.



**eBay has
chosen the
route of the
native app**

For now, eBay is not
on HTML5. Its media
team says one is on
the way. The company
did however launch its
HTML5 Fashion app.

**LAGGING
BEHIND**

**Wikipedia isn't
currently
making full use
of HTML5
features**

The desktop version of
Wikipedia isn't in
HTML5 but the Wiki's
tech team is looking
into implementing a
variety of HTML5
features in the future.



**WordPress
homepage
isn't using
HTML5**

The blogging
platform's homepage
does however look
great on mobile
devices and tablets.
WordPress also has
multiple applications
designed for mobile
devices allowing users
to download apps best
suited for their needs.

EUD

de egresados U. Distrital
Construyendo Profesionales



NUEVOS ELEMENTOS

Nuevos elementos

- **Sintaxis simplificada:**
 - `<DOCTYPE HTML!>` para indicar el tipo de documento
- **Código más estructurado:**
 - `<header>` y `<footer>` (en vez de `<div>`)
 - `<section>` y `<article>`
 - `<menu>` y `<figure>`
 - `<audio>` y `<video>`

Nuevos elementos

- Cambio en los formularios
 - `<form>` y `<forminput>`
- **<Canvas>**: representación de gráficos (HTML5 + javascript reemplazando a Flash)
- Eliminación de las etiquetas:
 - `` y `` (manejo con CSS)
 - `<frame>`, `<center>` y `<big>`

SEMÁNTICA Y MICRODATOS

Semántica

- Elementos **con significado** tanto para el desarrollador como para el navegador:
 - **Sin semántica:** <div> y
(no dicen nada sobre el contenido, HTML4)
 - **Con semántica:** <form>, <table>,
(se reconoce de qué contenido se está hablando, HTML5)

Semántica

Describing the structure of a web page in HTML 4

<div id="header"> *This div element contains branding like the logo*

<div id="nav"> *This div element contains the site navigation*

<div id="content">

This div element contains the web page's main content

<div id="sidebar">

This div element contains extra information and related content/links

<div id="footer"> *This div element contains copyright information*

Describing the structure of a web page in HTML 5

<header> *This element contains branding like the logo*

<nav> *This element contains the site navigation*

<article>

This element contains the web page's main content

<aside>

This element contains extra information and related content/links

<footer> *This element contains copyright information*

```

<!DOCTYPE html>
<html>
<body>
<header>
  <h1>Header</h1>
  <p><time pubdate datetime="2013-08-24"></time></p> </header>
<nav>
<a href="/html/">HTML- nav 1</a> |
<a href="/css/">CSS- nav 2</a> | </nav>
<section>
  <h1>Sección 1</h1>
  <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p> </section>
<article>
  <h1>Artículo</h1>
  <p>Pellentesque ultricies porta mi sit amet molestie. Vestibulum tincidunt sem orci.</p>
</article>
<figure>
  
  <figcaption>Fig.1 -Qwerty.</figcaption> </figure>
<aside style="font-size:larger;font-style:italic;color:green;float:right;width:35%;padding-left:5px;">
  <h2>Aside Bar</h2>
  <ul>
    <li><a href="#">A link to the past</a></li>
    <li><a href="#">A link between worlds</a></li>
    <li><a href="#">Hey! Listen!</a></li>
  </ul>
</aside>
<footer>
  <p>Posted by: LG</p> </footer>
</body>
</html>

```



Resultado

Header

[HTML- nav 1](#) | [CSS- nav 2](#) |

Sección 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Articulo

Windows Internet Explorer 9 Pellentesque ultricies porta mi sit amet molestie...



Fig.1 -Norway.

Posted by: LG

Microdatos

- Grupos de pares nombre-valor. Estos grupos se denominan *items* y cada par nombre-valor es una *propiedad*.
- **Atributo itemscope:**
 - ámbito de un ítem
- **Atributo itemprop:**
 - propiedad del ítem

Microdatos

ITEM



```
<div itemscope>
```

```
Título: <span itemprop="título"> Blade Runner  
</span>
```

```
Dirección: <span itemprop="autor">Ridley  
Scott </span>
```

```
Reparto: <span itemprop="reparto"> Harrison  
Ford, Rutger Hauer, Sean Young, Edward James  
Olmos, Daryl Hannah </span>
```

```
Año: <span itemprop="años">1982.</span>
```

```
</div>
```



Microdatos

- **Schema.org**


- Para que estas etiquetas semánticas funcionen deben referenciarse en algún lugar y bajo un mismo término unificado
- Para contribuir a la web semántica:
Ej: autor, autoría, responsable, etc... se unifica a autor y todos utilizan el mismo término para el mismo concepto

Microdatos

- **Schema.org**

- Para usarlo: añadir donde se especifique el itemscope el **atributo itemtype** y como valor la URL.

Blade Runner se encuentra en esta URL



```
<div itemscope itemtype="http://schema.org/Movie">  
  Título: <span itemprop="título"> Blade Runner </span>  
  Dirección: <span itemprop="autor">Ridley Scott </span>  
  Reparto: <span itemprop="reparto"> Harrison Ford, ...</span>  
  Año: <span itemprop="años">1982.</span>  
</div>
```

MULTIMEDIA

Multimedia: <audio>

- HTML5 define un nuevo elemento que estandariza la manera de embeber archivos de audio en una página web

```
<!DOCTYPE html>  
<html>  
<body>
```

<audio controls>

```
  <source src="horse.ogg" type="audio/ogg">  
  <source src="horse.mp3" type="audio/mpeg">
```

Your browser does not support the audio element.

</audio>

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



Multimedia: <video>

- Estandariza la manera de embeber archivos de video, como un video clip o video stream, en una página web

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



GRAFICOS HTML5

Gráficos HTML 5

- Canvas
- SVG

Canvas

- Un canvas se utiliza para dibujar gráficos en tiempo de ejecución mediante scripts (usualmente javascript).
- Cuenta con métodos para agregar figuras como líneas, cuadrados, círculos, entre otros.
- Ideal para creación de juegos

Canvas

- Se crea usando el tag <canvas>
- `<canvas id="canvas" width="550" height="400"></canvas>`
- Un canvas no cuenta con borde o contenido inicial

Canvas

- Se crea usando el tag <canvas>
- `<canvas id="canvas" width="550" height="400"></canvas>`
- Un canvas no cuenta con borde o contenido inicial
- El id es importante pues es la referencia para el script

```

<!DOCTYPE html>
<html>

  <head>

    <style type="text/css">
      canvas{border:#666 1px solid;}
    </style>
    <script type="text/javascript">
      function draw(x,y){
        var canvas = document.getElementById('canvas');
        var ctx = canvas.getContext('2d');
        ctx.save();
        ctx.clearRect(0,0,550,400);
        ctx.fillStyle = "rgba(200,0,0,1)";
        ctx.fillRect (x, y, 50, 50);
        ctx.restore();
        x += 1;
        y += 1;
        var loopTimer = setTimeout('draw('+x+', '+y+)',30);
      }
    </script>

  </head>
  <body>

    <button onclick="draw(0,0)">Draw</button>
    <canvas id="canvas" width="550" height="400"></canvas>

  </body>

</html>

```

```
<!DOCTYPE html>
<html>

  <head>

    <style type="text/css">
      canvas{border:#666 1px solid;}
    </style>
    <script type="text/javascript">
      function draw(x,y){
        var canvas = document.getElementById('canvas');
        var ctx = canvas.getContext('2d');
        ctx.save();
        ctx.clearRect(0,0,550,400);
        ctx.fillStyle = "rgba(200,0,0,1)";
        ctx.fillRect (x, y, 50, 50);
        ctx.restore();
        x += 1;
        y += 1;
        var loopTimer = setTimeout('draw('+x+', '+y+')',30);
      }
    </script>

  </head>
  <body>

    <button onclick="draw(0,0)">Draw</button>
    <canvas id="canvas" width="550" height="400"></canvas>

  </body>

</html>
```

SVG

- Scalable Vector Graphics
- Define gráficos en XML
- Pueden ser animados
- Es pesado si es complejo
- No apto para juegos

SVG: Primitivas

- `<line x1 y1 x2 y2>`
- `<rect x y width height>`
- `<circle cx cy r>`
- `<path d>`
- `<image x y width height xlink:href>`

`<svg>`

`<rect x="5" y="5" width="140" height="140" stroke="#000000"
stroke-width="4" fill="#AAAAFF" opacity="1"/>`

`</svg>`

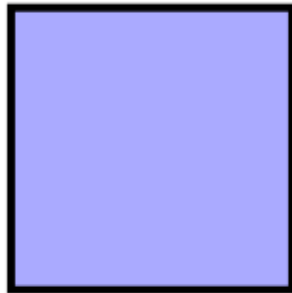
SVG: Ejemplo rectángulo

```
<svg>  
  <rect x="5" y="5" width="140" height="140" stroke="#000000"  
    stroke-width="4" fill="#AAAAFF" opacity="1">  
  </rect>  
</svg>
```



SVG: Ejemplo rectángulo

```
<svg>  
  <rect x="5" y="5" width="140" height="140" stroke="#000000"  
    stroke-width="4" fill="#AAAAFF" opacity="1">  
  </rect>  
</svg>
```



SVG: Animar

```
<animate dur="3s" attributeName="r"  
values="80; 150; 80"  
repeatCount="indefinite" />
```



SVG: Animar

`<animate dur="3s" attributeName="r" values="80; 150; 80" repeatCount="indefinite" />`

Duración

Atributo

Valores

Repeticiones

SVG: Circulo Animado

```
<circle cx="200" cy="205" r="80" >  
<animate dur="3s" attributeName="r"  
values="80; 150; 80"  
repeatCount="indefinite" /> </circle>
```

FORMULARIOS

Formularios

- Etiqueta **<form>**
- **<input>**: especifica el tipo de entrada para cada campo
- Cada campo se asocia a un **<label>**
- Propiedad **required**: para definir todos los campos como requeridos
- **placeholder** : el valor esperado de cada campo de entrada

```

<form class="contact_form" action="#" method="post">
<ul>
  <h2>Contáctanos</h2>

  <li>
    <label for="name">Nombre:</label>
    <input type="text" name="name" placeholder="Diego Rincon" required />  </li>
  <li>
    <label for="email">Email:</label>
    <input type="email" name="email" placeholder="d1egoprog@ejemplo.com"
required />  </li>
  <li>
    <label for="web">Sitio Web:</label>
    <input type="url" name="web" placeholder="http://www.d1egoprog.co"
required />  </li>
  <li>
    <label for="Mensaje">Mensaje:</label>
    <textarea name="Mensaje" cols="40" rows="6" required ></textarea> </li>

  <button class="submit" type="submit">Enviar</button>

</ul>
</form>

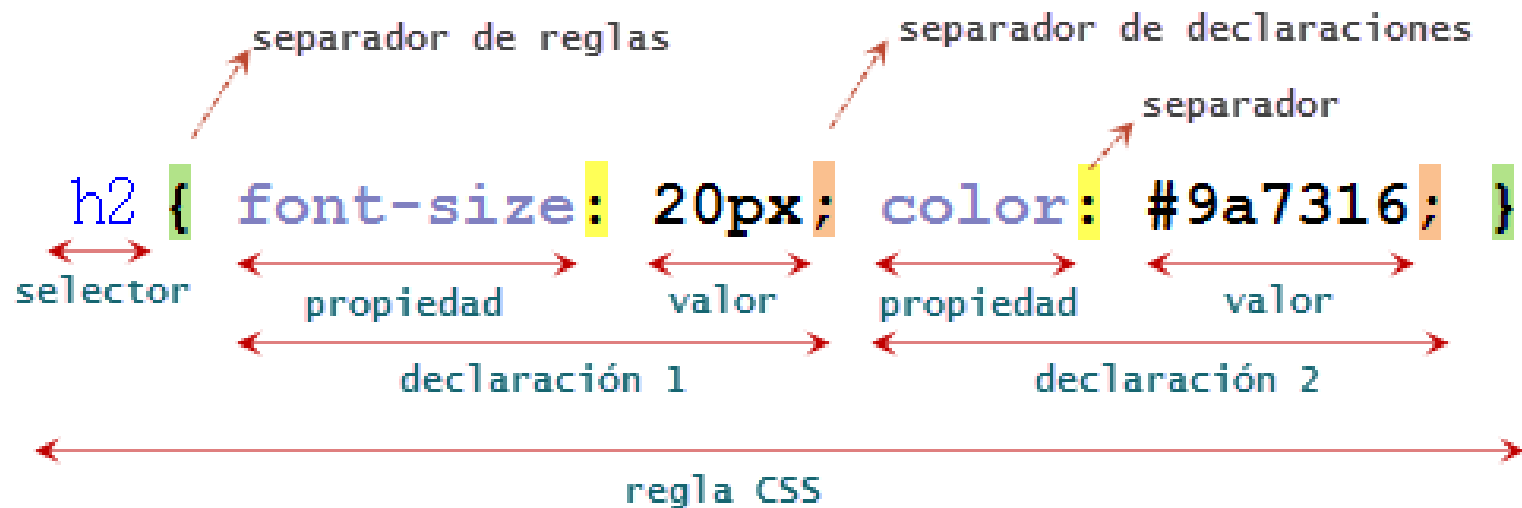
```

CSS

CSS

- Forma mas limpia de programación
- Código del diseño de una pagina web
- Separación de responsabilidades

Regla CSS



Regla CSS

- **Regla:** Cada uno de los estilos definidos
- **Selector:** Elementos a los que se les aplica la regla CSS
- **Declaración:** Estilos que se aplican, puede haber uno o más estilos
- **Propiedad:** Característica a modificar del elemento en el selector
- **Valor:** Valor de la propiedad

Selectores CSS

- * = Selector Universal
- div, p, a, h1 = Tipo o etiqueta
- p span = Selector descendente
- . = Selector de clase
- # = Selector de ID
- div.algo span.otro = Combinación de selectores
- tr:nth-child(even | odd) = hijo n de tr (par | impar)

CASCADA = HERENCIA

- Los estilos se aplican en cascada, es decir que el elemento “padre” hereda su estilo a todos los “hijos” “nietos” “bisnietos” “tataranietos” “choznos”...
- Ej: Si se agrega un estilo (regla) al cuerpo (body) cualquier etiqueta dentro de body heredará este estilo.

Archivo css/estilo.css

```
h1{  
    color:blue;  
}  
  
body{  
    font-family: Verdana, sans-serif;  
}  
  
table {  
    border: 1px solid #000;  
    border-collapse: collapse;  
    border-spacing: 0px;  
}  
tr:nth-child(even){  
    background: #AAAAFF;  
}  
th{  
    background: #aaa;  
    border: 1px solid #000;  
}  
td{  
    border: 1px solid #000;  
}
```

Vinculando Archivos

- Agregar a la cabecera la siguiente linea:

```
<head>
```

```
  <meta charset = "UTF-8">
```

```
  <title>DEMO HTML</title>
```

```
  <link rel="stylesheet" href="css/estilo.css">
```

```
</head>
```

Clases y ID

Clases

- Las clases pueden ser reutilizadas en muchos elementos
- Útil cuando se requiere agrupar un conjunto de estilos y aplicarlo a varios elementos

ID

- Los id identifican a un único elemento
- Útil cuando se requiere acceder a el desde otro archivo



Asignar un ID a un elemento

```
<p id="unParrafo">Parrafo # 1</p>
```

Asignar una Clase a un elemento

```
<p class="clasePrueba">Parrafo # 1</p>
```

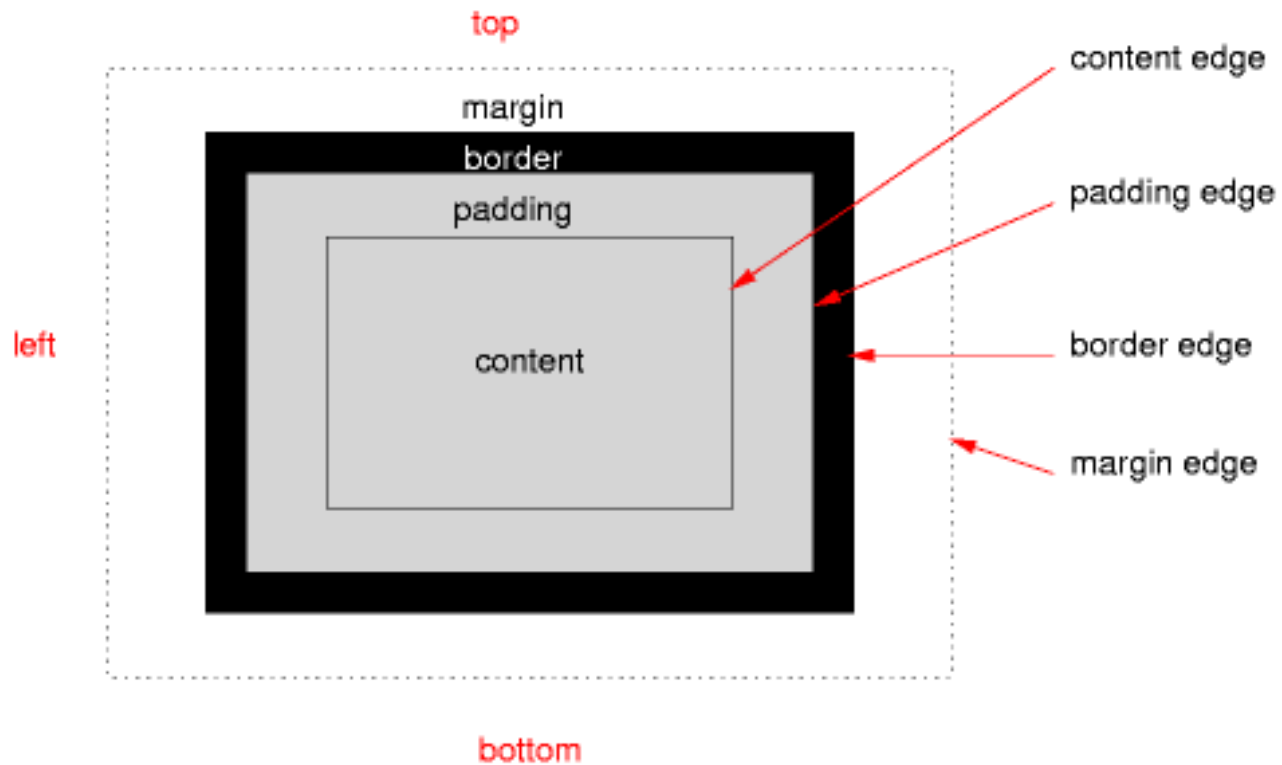
Ejercicio 1: Usar CSS

- Agregar color a las filas impares de la tabla
- Colocar un nuevo párrafo
- Crear una clase llamada “centrada” y asignarla a dicho párrafo y a la tabla
- En el CSS asignar un color rojo y una alineación centrada a la clase centrada
- Asignar al titulo un id de “Titulo1”
- En el css darle fondo al “Titulo1”

CSS: Alineación

- Se considera cada elemento de html como una caja (Box)
- Existen 4 “cascarones” en cada elemento de un html, los cuales definen su alineación.
- Esto se llama el “Modelo de caja” (Box Model) de CSS

CSS: Alineación



CSS: Alineación

- **Contenido:** El elemento como tal; Ej: Un texto o una imagen
- **Padding:** Espacio dentro del elemento que lo rodea
- **Border:** Espacio entre el Margin y el Padding
- **Margin:** Espacio fuera del elemento

Visualización de la Alineación

- Crear en un documento un elemento DIV con id “caja”
- En el CSS agregar lo siguiente:

```
#caja {  
    /* Mover Elementos */  
    border: 5px dotted red;  
    margin: 0px;  
    padding: 13px;  
    background: gray;  
}
```

Ejercicio 2: Alineación

- Con la siguiente pagina:

```
<!DOCTYPE html>
<html>
<head>
  <title>Resultado</title>
  <link rel="stylesheet" href="css/estilo.css" />
</head>
<body>
<div id="uno">
  Un primer div!
</div>
<div id="dos">
  Un segundo div, <span>usando</span> <span>spans</span>!
</div>
<div id="tres">
  Un tercer div, <span>también con</span> <span>spans</span>!
</div>
</body>
</html>
```



Ejercicio 2: Alineación

- Darle al primer div un color rojo y un borde de 1 pixel solido de color negro
- Darle al segundo div un color verde y un borde de 1 pixel punteado de color negro
- Darle al tercer div un color azul y un borde de 1 pixel lineado de color negro
- Darle a todos los span un borde rojo

Ejercicio 2: Alineación

- Darle un margen al primer div de 15px en todos los lados
- Darle un padding al segundo div de 20px de todos los lados
- Darle un padding al tercer div de -20px

Elementos Flotantes

- Sirve para colocar imágenes o elementos alrededor del texto en una estructura no lineal.
- Se usa float:right o float:left como propiedad del elemento
- Ejemplo:

```
#primero {  
    float:left;  
}  
#segundo {  
    float:right;  
}
```

Elementos Flotantes

- Intentar con un par de imágenes

Índice Flotante

- Agregar un índice a la página con un div:

```
<div id="navbar">  
  <ul>  
    <li>Home</li>  
    <li>About</li>  
    <li>Products</li>  
    <li>Contact</li>  
  </ul>  
</div>
```



Ejercicio 3: Barra de navegación

- Coloque la barra de navegación flotante hacia la izquierda

Media Queries

Una media query consiste en un tipo de medio y al menos una consulta que limita las hojas de estilo utilizando características del medio como ancho, alto y color. Se entiende como un módulo CSS3 que permite adaptar la representación del contenido a características del dispositivo. Añadido en CSS3, las media queries dejan que la presentación del contenido se adapte a un rango específico de dispositivos de salida sin tener que cambiar el contenido en sí.

Media Queries

Utiliza el media query para incluir un bloque de propiedades CSS sólo si una determinada condición es verdadera.

```
@media (min-width: 500px) {  
    h1{  
        margin: 1%;  
    }  
}
```

Media Queries

Operadores lógicos para las Media Queries

Para combinar diversas condiciones podemos usar operadores lógicos

Operador and: las dos condiciones deben cumplirse para que se evalúe como verdadera.

Operador not: es una negación de una condición. Cuando esa condición no se cumpla se aplicarán las media queries.

Media Queries

Operador only: se aplican las reglas solo en el caso que se cumpla cierta circunstancia.

Operador or: no existe como tal, pero puedes poner varias condiciones separadas por comas y cuando se cumpla cualquiera de ellas, se aplicarán los estilos de las media queries.

```
@media only screen and (min-width: 480px) and (max-width: 767px) {  
    body {  
        background-color: lightblue;  
    }  
}
```



Media Queries

Orientacion

Tambien puedes definir la orientación en los media queries

```
@media (max-width: 600px) and (orientation: landscape)
{
    h1{
        color: red;
    }
}
```



Media Queries

¿Que es el Viewport?

Es una etiqueta HTML5, sirve para definir que área de pantalla está disponible al renderizar un documento, cuál es el nivel de escalado que puede realizar el usuario, así como si el navegador debe mostrarla con algún zoom inicial.

```
<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1">
```


Javascript

- Lenguaje de Programación interpretado.
- Utilizado en el Navegador.
- Sintaxis basada en C/C++.
- Sensible entre mayúsculas/minúsculas
- Débilmente tipado.

Uso de Javascript

- Declaración de variables

```
var cadena = "PruebaCadena";  
var numero = 54;  
numero = "Otra Cadena";
```

- Declaración de funciones

```
function nombreFuncion1(param1,param2){  
    //..TO-DO  
}
```

```
var nombreFuncion2 = function (param1,param2){  
    //..TO-DO  
}
```

Uso de Javascript

- Condicionales

```
if(condicion){
```

```
}
```

```
else{
```

```
}
```

```
for(var i=0;i<longitud;i++){
```

```
}
```

```
while(condicion){
```

```
}
```

```
do{
```

```
}while(condicion);
```

```
switch(n)
```

```
{
```

```
case 1:
```

```
//Algo
```

```
break;
```

```
case 2:
```

```
//Algo
```

```
break;
```

```
default:
```

```
//Algo si fallan los anteriores
```

```
}
```



Uso con HTML: Seleccionar elementos

- getElementById
- getElementsByTagName
- querySelector
- querySelectorAll

Uso con HTML: Agregar eventos

- HTML con:
 - onclick
 - onblur
 - onfocus
- Programáticamente
 - addEventListener

Crear una carpeta js y un archivo script.js

- En el archivo .js colocar el siguiente código:

Crear una carpeta js y un archivo script.js

```
window.onload=function(){
    console.log("Pagina Cargada!");
    var primerParrafo = document.getElementById("primer-parrafo");
    primerParrafo.addEventListener("click",function(){
        mostrarMsj("click sobre el parrafo");
    });
    console.log(primerParrafo);
};

var mostrarMsj = function(texto){
    alert(texto);
    prompt("Como va todo?");
};
```



Crear una carpeta js y un archivo script.js

```
window.onload=function(){  
    console.log("Pagina Cargada!");  
    var primerParrafo = document.getElementById("primer-parrafo");  
    primerParrafo.addEventListener("click",function(){  
        mostrarMsj("click sobre el parrafo");  
    });  
    console.log(primerParrafo);  
};  
  
var mostrarMsj = function(texto){  
    alert(texto);  
    prompt("Como va todo?");  
};
```



Crear una carpeta js y un archivo script.js

```
window.onload=function(){  
    console.log("Pagina Cargada!");  
    var primerParrafo = document.getElementById("primer-parrafo");  
    primerParrafo.addEventListener("click",function(){  
        mostrarMsj("click sobre el parrafo");  
    });  
    console.log(primerParrafo);  
};  
  
var mostrarMsj = function(texto){  
    alert(texto);  
    prompt("Como va todo?");  
};
```



Crear una carpeta js y un archivo script.js


```
window.onload=function(){  
    console.log("Pagina Cargada!");  
    var primerParrafo = document.getElementById("primer-parrafo");  
    primerParrafo.addEventListener("click",function(){  
        mostrarMsj("click sobre el parrafo");  
    });  
    console.log(primerParrafo);  
};  
  
var mostrarMsj = function(texto){  
    alert(texto);  
    prompt("Como va todo?");  
};
```



Crear una carpeta js y un archivo script.js

```
window.onload=function(){
    console.log("Pagina Cargada!");
    var primerParrafo = document.getElementById("primer-parrafo");
    primerParrafo.addEventListener("click",function(){
        mostrarMsj("click sobre el parrafo");
    });
    console.log(primerParrafo);
};

var mostrarMsj = function(texto){
    alert(texto);
    prompt("Como va todo?");
};
```



Función Anónima

Vinculando HTML con JS

<head>

<meta charset = "UTF-8">

<title>DEMO HTML</title>

<link rel="stylesheet" href="css/estilo.css">

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

</head>

Prueba de Ejecución

- Windows
 - Para ver la consola en Chrome:
 - Ctrl + Shift+ J
 - Para ver la consola en Firefox:
 - Shift + F2 / Ctrl+Shift+I
- Mac
 - Para ver la consola en Chrome:
 - Command+Alt+ J

Usando QuerySelector

```
window.onload=function(){  
    console.log("ENTRO!");  
    var primerParrafo = document.querySelector("p");  
    primerParrafo.addEventListener("click",function(){  
        mostrarMsj("click sobre el parrafo");  
    });  
    console.log(primerParrafo);  
};  
  
var mostrarMsj = function(texto){  
    alert(texto);  
    prompt("Como va todo?");  
};
```

Haciendo un efecto “hover”

```
primerParrafo.addEventListener("mouseover",function(){
    cambiarColor(this, "#FF0000");
});
primerParrafo.addEventListener("mouseout",function(){
    cambiarColor(this, "#000000");
});
```



Ejercicio: Hacer función cambiarColor

- Recibe por parametro el elemento y el color

Ejercicio: Hacer función cambiarColor

- Recibe por parámetro el elemento y el color

```
function cambiarColor(elemento,color){  
    elemento.style.color=color;  
}
```

Ejercicio: Que el color sea aleatorio

- Crear un arreglo con los números (en string) del 0 al 9 y las letras desde la a hasta la f
- Utilizar función `Math.floor((Math.random()*16));` para generar un número aleatorio de 0 a 16.
- Este número será el índice de una posición en el arreglo

Ejercicio: Que el color sea aleatorio

- Crear una variable por color (rojo, verde, azul)
- Agregar dos elementos aleatorios a cada una
 - Ej: rojo = F4
 - Verde = 03
 - Azul = 64
- Finalmente agregar a una variable llamada color los tres componentes anteriores precedidos del carácter “#”



Respuesta

```
function cambiarColor(elemento){  
    var posibilidades=["0","1","2","3","4","5","6","7","8","9","A","B","C","D","E","F"];  
  
    var rojo = posibilidades[aleatorio()+posibilidades[aleatorio()];  
    var verde = posibilidades[aleatorio()+posibilidades[aleatorio()];  
    var azul = posibilidades[aleatorio()+posibilidades[aleatorio()];  
    var color = "#"+rojo+verde+azul;  
    console.log("El color es: "+color);  
    elemento.style.color=color;  
}  
  
function aleatorio(){  
    return Math.floor((Math.random()*16));  
}
```



CLASES EN JAVASCRIPT

Formas de definir una clase

- Existen 3 formas de definición de una clase Javascript
 - Función
 - Literal
 - Función anónima (Singleton)

Función

```
function MiObjeto(){  
    var att2 = 'variable privada';  
    this.att1 = 'variable publica';  
    this.mtd1 = function(){  
        alert('Esto es '+this.att1);  
    };  
}
```

Función

```
function MiObjeto(){  
    var att2 = 'variable privada';  
    this.att1 = 'variable publica';  
    this.mtd1 = function(){  
        alert('Esto es '+this.att1);  
    };  
}
```


Función

```
function MiObjeto(){  
    var att2 = 'variable privada';  
    this.att1 = 'variable publica';  
    this.mtd1 = function(){  
        alert('Esto es '+this.att1);  
    };  
}
```

Literal

```
var MiObjeto = {  
    att1 : 'un objeto',  
    mtd1 : function(){  
        alert('Esto es '+this.att1);  
    }  
}
```

Diferencias en definición

Función

- Requiere de la palabra “this” para definir atributos y métodos.
- Para asignar valores se utiliza el signo “=”.
- Al final de cada elemento se coloca “;”

Literal

- No requiere de la palabra “this” para definir atributos y métodos.
- Para asignar valores se utiliza el signo “:”.
- Al final de cada elemento se coloca “,”



Diferencias en declaración

Función

- Requiere del uso de un constructor (new)
- Uso:
 - `var nuevoObjeto = new MiObjeto();`

Literal

- Es un “singleton” ya que es una sola instancia
- Uso:
 - `MiObjeto.mtd1();`



XMLHttpRequest

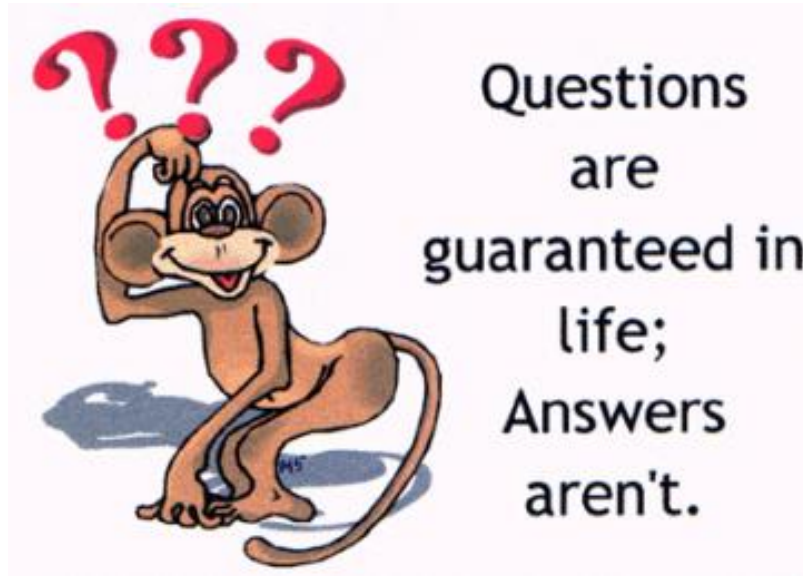
XMLHttpRequest es un objeto JavaScript que fue diseñado por Microsoft y adoptado por Mozilla, Apple y Google. Actualmente es un estándar de la W3C. Proporciona una forma fácil de obtener información de una URL sin tener que recargar la página completa. Una página web puede actualizar sólo una parte de la página sin interrumpir lo que el usuario está haciendo. XMLHttpRequest es ampliamente usado en la programación AJAX

AJAX: es un acrónimo de Asynchronous JavaScript and XML, es decir: JavaScript y XML asincrónico. AJAX se define como una técnica para el desarrollo de páginas (sitios) web que implementan aplicaciones interactivas.

JavaScript: Lenguaje de programación conocido por ser interpretado por los navegadores de páginas web.

XML: es un lenguaje de descripción de datos pensado fundamentalmente para el intercambio de datos entre aplicaciones, más que entre personas.

Asíncrono: En el contexto de las comunicaciones (y la visualización de una pagina web no deja de ser un acto de comunicación entre un servidor y un cliente) significa que el emisor emite un mensaje y continúa con su trabajo, dado que no sabe (ni necesita saberlo) cuándo le llegará el mensaje al receptor



¿Preguntas?

Diego Alberto Rincón Yáñez MSc.

Julio Cesar Galeano Garcia