

redefine tu carrera, redefine tu vida

WEB FUNDAMENTALS

*HTML5
CSS3
JAVASCRIPT*

Adriana Milenka Paz Candia

Acerca de mí...

- Ingeniería de Sistemas – UMSS
- Diplomado en Educación superior – UMSS
- Máster Dirección de Sistemas de Información - USAL
- Máster Animación Digital – USAL
- FrontEnd Developer – TRUEXTEND
- Freelancer - NY

¡BIENVENIDOS!

:D

INTRODUCCION AL CURSO

- Diapositivas.
- Ejemplos prácticos.
- Ejemplos de páginas web.
- Cubrir elementos principales HTML, por ejemplo <div>, , etc.
- Uso de estilos y propiedades: align, width, position, top, font-size, etc.

OBJETIVOS

- Crear páginas web con diferentes elementos HTML.
- Cambiar el display y el comportamiento de los elementos HTML usando CSS y Javascript.
- Páginas modernas: Single page.

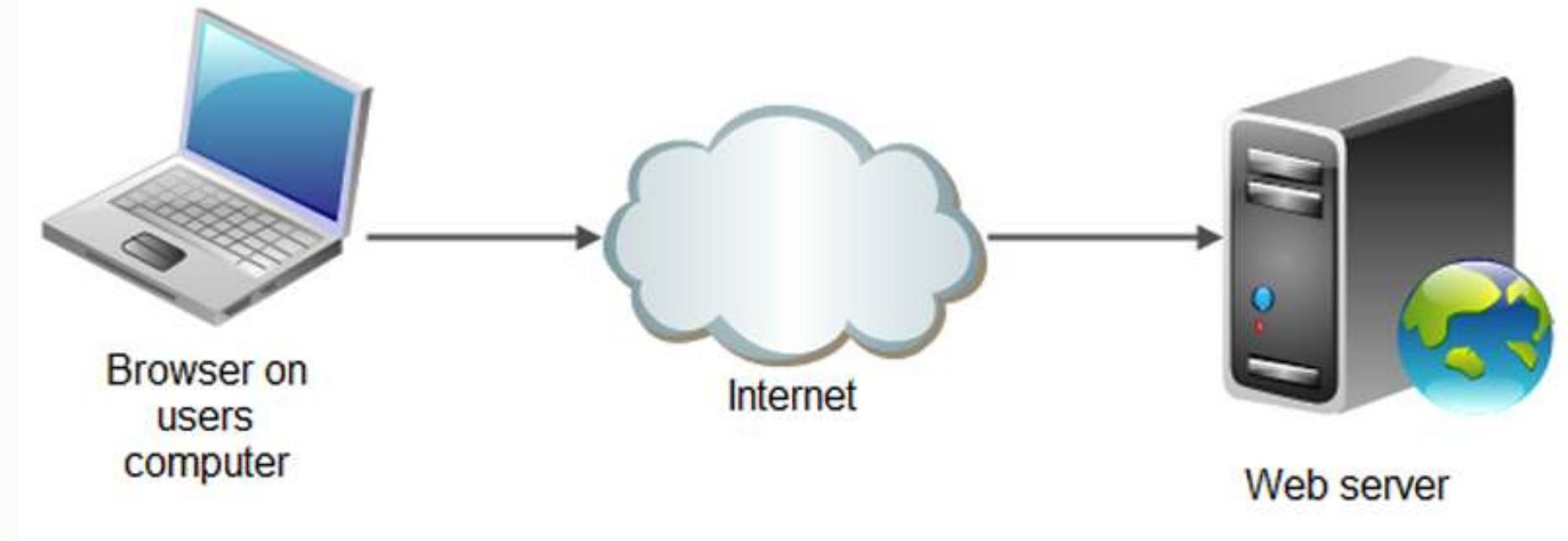
BASES DE EVALUACIÓN

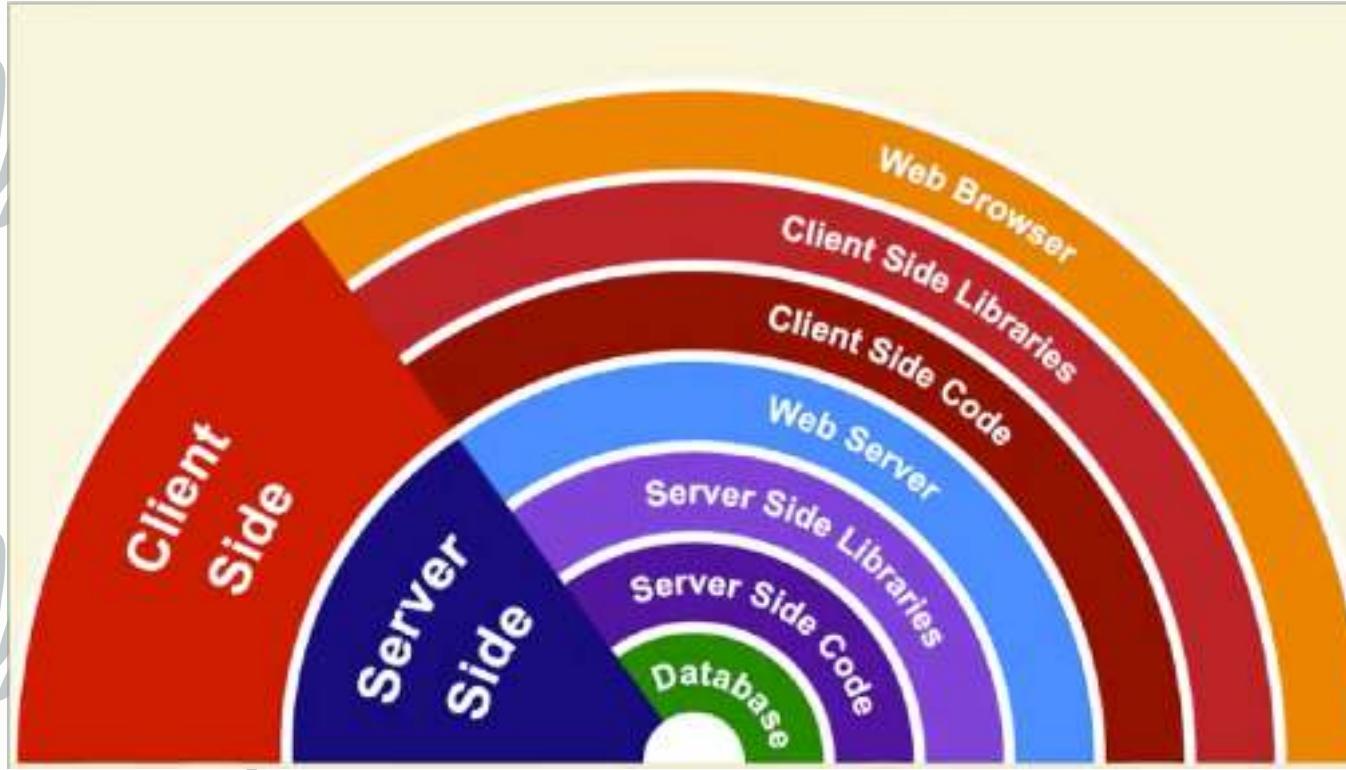
- Ejercicios semanales.
- Entrega de los 3 ejercicios al fin de curso.

HTML BÁSICO

HTML

INTERNET

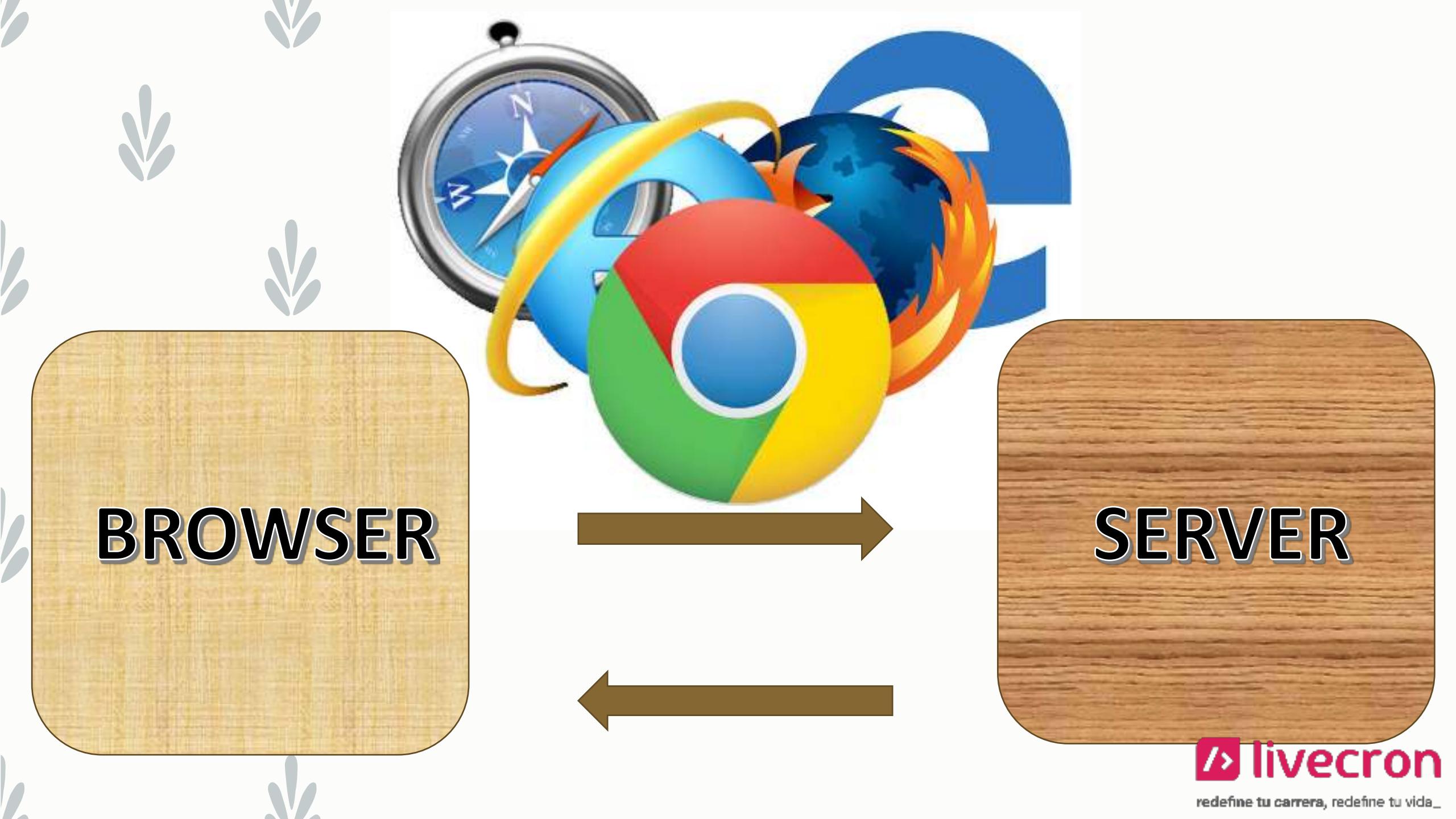




<http://html-css-js.com/>

Sobre HTML

- **HTML**, sigla en inglés de ***HyperText Markup Language*** (lenguaje de marcas de hipertexto), hace referencia al lenguaje de marcado para la elaboración de páginas web.
- Es un estándar a cargo del [World Wide Web Consortium\(W3C\)](#) o Consorcio WWW.
- Tiene diferentes versiones desde los 90s.
- La última versión es HTML5.
- Una alternativa es SVG. (Scalable Vector Graphics)



BROWSER

SERVER

Conociendo HTML

-
- El HTML se escribe en forma de «etiquetas», rodeadas por corchetes angulares (<,>,/)
 - Los comandos son llamados elementos.
 - Tienen un tag de inicio y uno de fin.

<p> . . . </p>

Estructura de una página HTML

```
<!DOCTYPE html>
<html>
  <head>
    ...
  </head>
  <body>
    ...
  </body>
</html>
```

pagina web 1

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Mi primera Página Web</title>
5     <meta name="author" content="Adriana Paz">
6   </head>
7   <body>
8     <h1>Mi Web</h1>
9     <p>Aquí pondré lo que aprenderé en el Summer Bootcamp!</p>
10    </body>
11  </html>
12
```

Mi Web

Aquí pondré lo que aprenderé en el Summer Bootcamp!

Head: Elementos y atributos...

-
- <head> Información sobre la página web.
 - <title> Título de la página.
 - <meta> Mayor información.
 - <style> estilos
 - <link> referencia a un archivo
 - <script> código javascript
 - <base> a quien pertenecen.
 - Atributos: <meta name="author" content="Adriana Paz">

Body: H1 & P

-
- <h1> heading
 - <p> paragraph
 - Inspeccionemos...

Un editor para HTML

- Graphical User Interface
- **Lo único que necesitamos para crear páginas y aplicaciones web es un simple editor de texto plano y nuestra imaginación**
- Complementan el aprendizaje.
- <https://www.campusmvp.es/recursos/post/Los-10-mejores-editores-gratuitos-de-HTML-CSS-y-JavaScript.aspx>

Elementos más comunes

- Headings: <h1> <h2> <h3> <h4> <h5> <h6>
- Sections: <section>
- Lists: y junto con
- Comments: <!-- un comentario -->

Dando formato a textos

- ITALIC: *<i>* & EMPHASIS: **
- BOLD: **** & ****
- UNDERLINE: <u>
- BIG & SMALL: <big> & <small>
- HIGHLIGHTING: <mark>
- SUBSCRIPT & SUPERSCRIPT: <sub> & <sup>
- INSERTED AND DELETED: <ins> & ~~<deleted>~~

MULTIMEDIA



Imágenes

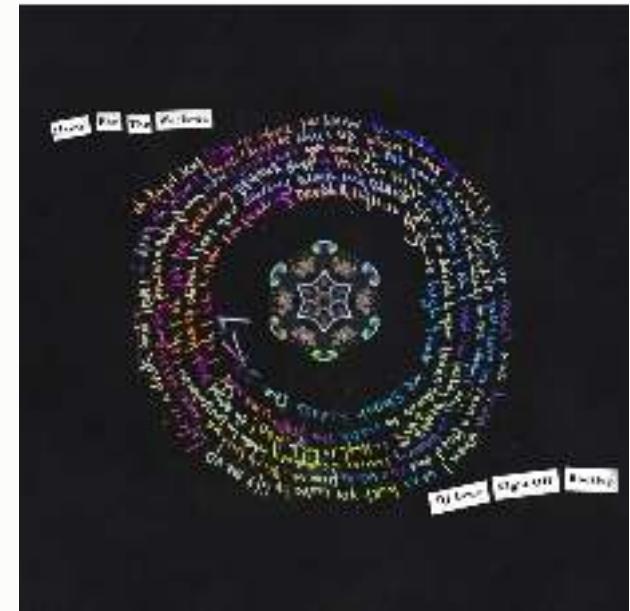
-
- Atributos:
 - *Source*
 - *Width*
 - *Height*

```
1 
```



Audio

- <audio>
- Atributos:
 - *src*
 - *autoplay*
 - *controls*
 - *loop*
 - *alt*



```
1 <audio src="coldplay_hymn_for_the_weekend.mp3"></audio>
```

Video

- <video>
- Atributos:
 - *src*
 - *autoplay*
 - *controls*
 - *loop*
 - *alt*



```
1 <video src="caminata.mp4"></video>
```

PROFUNDIZANDO HTML



Links

- <a> anchor
- Atributos:
 - *href*
 - *id*

Elige un icono!



Ya revisaste tu gmail el día de hoy?

Posiciones en la página web

1. Usamos el **id="here"**
2. Y luego usamos la referencia en el link:
3. **Ir a here**

<http://www.lipsum.com>
/

Elementos Void y Breaks

<wbr><hr>

– VOID ELEMENTS

- <start_tag> . . . </end_tag>
- Elementos void no tienen contenido
- No usan el “end_tag”



Ejemplo de algunos elementos void:

- <meta name="author" content="AMPC">
- Multimedia:
- Formularios: <input>
- Breaks:
 <wbr> <hr>

Estilos

-
- <link>
 - href
 - rel
 - type
 - <style>
 - <HTML_ELEMENT>
 - id
 - Style properties:
 - *color*
 - *background*

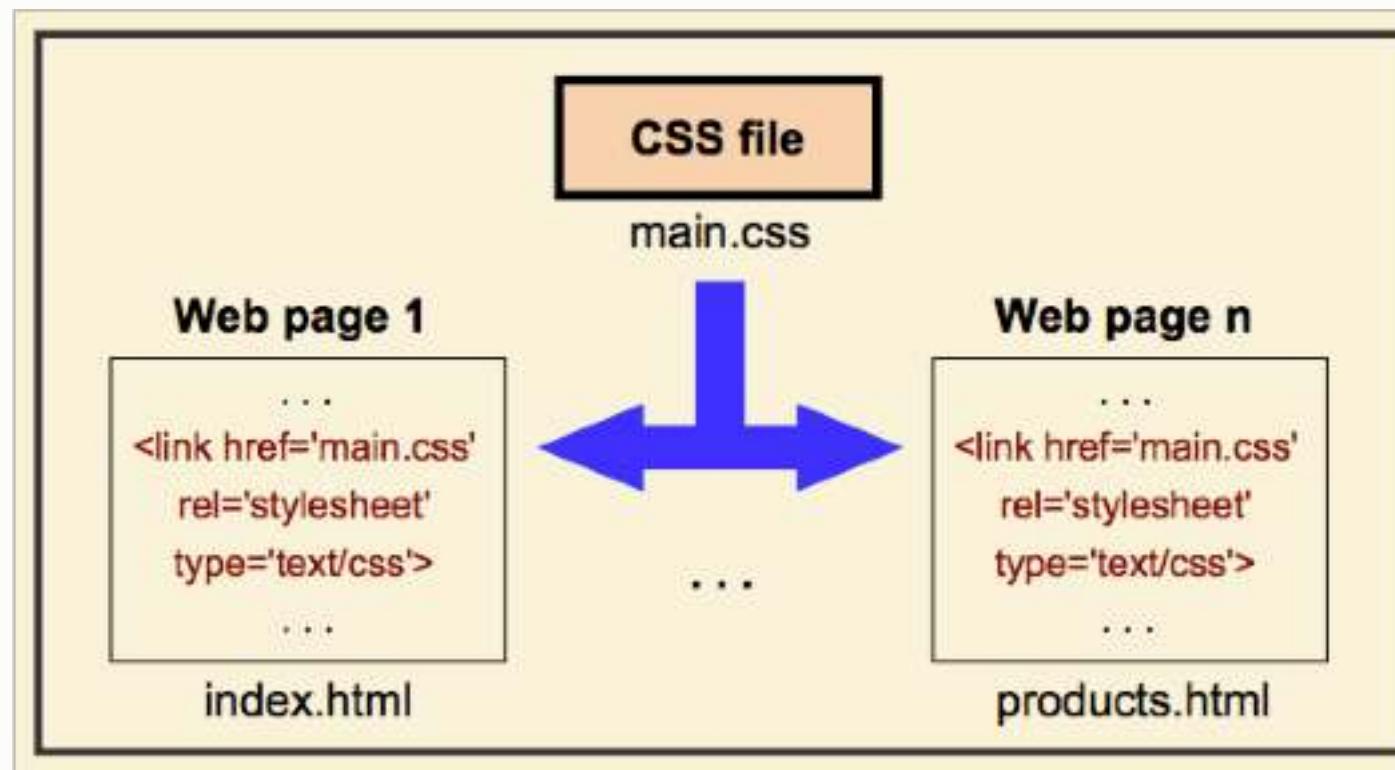
Necesitamos estilos!

- Sin los estilos la página es aburrida
- Con estilos también pueden ser controlados desde librerías JavaScript
- El lenguaje para aplicar estilos en las páginas web es: CSS, **Cascading Style Sheets**

En conceptos generales:

Información + Estilos = Salida visual

1 CSS, MUCHAS PÁGINAS WEB



Link a un file CSS

```
<!DOCTYPE html>
<html>
  <head>
    <title>Demostracion de link a un file de estilos</title>
    <link href="01_css_file.css" rel="stylesheet" type="text/css">
  </head>
  <body>
```

... elementos que usan las reglas de estilos van aquí...

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

... y el CSS es...

```
h1 { color:purple }
p { color:blue }
```

CSS dentro de HTML

```
<!DOCTYPE html>
<html>
  <head>
    <title>Demostracion de estilos dentro del file HTML</title>
    <style>
```

... los estilos van aquí...

```
    </style>
  </head>
  <body>
```

... elementos que usan las reglas de estilos van aquí...

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

Usando IDs únicos...

- Cada elemento puede tener un identificador único
- El id no tiene ningún efecto visual

```
<html>
  <body>
    <ul id="rainbowColors">
      <li id="red">Rojo</li>
      <li id="orange">Naranja</li>
      <li id="yellow">Amarillo</li>
      <li id="green">Verde</li>
      <li id="blue">Azul</li>
      <li id="indigo">Indigo</li>
      <li id="violet">Violeta</li>
    </ul>
  </body>
</html>
```

```
<style>
  #rainbowColors {background: grey}
  #red {background: red}
  #orange {background: orange}
  #yellow {background: yellow}
  #green {background: green}
  #blue {background: blue}
  #indigo {background: indigo}
  #violet {background: violet}
</style>
```

Usando Class...

- Puedes crear tus propias reglas y aplicarlas a lo que sea.
- Una regla puede ser aplicada a varios elementos al mismo tiempo

```
<html>
  <head>
    <style>
      .zappy {color:purple; background:yellow}
      .wow {color:blue; background:lightgrey}
    </style>
  </head>
  <body>
    <h1 class="zappy">Mi primer título</h1>
    <p class="wow">My primer párrafo</p>
    <h1 class="wow">Mi segundo título</h1>
    <p class="zappy">Mi segundo párrafo</p>
  </body>
</html>
```

Múltiples classes

- Un solo elemento puede utilizar múltiples classes:

```
<html>
  <head>
    <style>
      .zappy {color:blue}
      .spicy {color:red}
      .wow {background:lime}
      .lol {background:lightgrey}
    </style>
  </head>
  <body>
    <p class="zappy wow">Mi primer párrafo</p>
    <p class="zappy lol">Mi segundo párrafo</p>
    <p class="spicy wow">Mi tercer párrafo</p>
    <p class="spicy lol">Mi cuarto párrafo</p>
  </body>
</html>
```

100 cosas por hacer...

100 cosas por hacer antes de morir!!!



getloupe.com

1. Visitar al menos 10 países
2. Aprender a hablar 3 idiomas
3. Hacer un circuito largo (mínimo entre dos países) en moto
4. Migrar a un país a vivir en su totalidad
5. Recorrer el mundo de la mano de un fotógrafo
6. Escribir un libro
7. Ir a escalar rocas
8. Ser impulsivo al máximo, no controlar emociones
9. Estudiar un arte marcial
10. Competir en una carrera de aventura de varios días

Más sobre estilos...

-
- Inline Styles
 - Pseudo-classes
 - link
 - visited
 - hover
 - active
 - empty
 - Prioridades

Inline:

Se aplica a un elemento en particular

```
<p style="text-align: right">Bienvenido.</p>
```

Context control:

Se aplica a un contexto específico

```
ol li {background:green} /* context control */
```

Pseudo-class:

Hover: El mouse se encuentra sobre el elemento

```
h1:hover {color:red} /* pseudo-classes */
```

Link: Es un link

```
a:link {background:yellow} /* pseudo-classes */
```

Visited: link visitado

```
a:visited {background:pink} /* pseudo-classes */
```

Active : link activo

```
a:active {background:purple} /* pseudo-classes */
```

Empty: elemento vacío

```
li:empty {background:brown} /* pseudo-classes */
```

Tablas

- Son una manera de tener una estructura dentro de la página.
 - Se necesitan varios tags trabajando juntos.
-
- Estructura: <table> <thead> <tbody>
 - Cabecera: <th>
 - Cuerpo: <tr> <td>

Estilos para las tablas:

- Bordes: border
- Ancho: width
- Alto: height
- Alineación vertical: vertical-align
- Espacio de relleno: padding

Estructura

```
<table>
  <thead>
    <tr>
      <th>Skills</th>
      <th>Dificultad</th>
      <th>Mi nivel</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>HTML</td>
      <td>Fácil</td>
      <td>Algo</td>
    </tr>
    <tr>
      <td>CSS</td>
      <td>Medio</td>
      <td>Un poco</td>
    </tr>
    <tr>
      <td>JavaScript</td>
      <td>Difícil</td>
      <td>Nada</td>
    </tr>
  </tbody>
</table>
```

```
<html>
  <head>
    <style>
      table, td, th {
        border: 1px solid green;
        width:50px;
        text-align:center
      }
      .ganancia {
        text-align:left;
        background-color:lightblue
      }
      .cero {
        text-align:center;
        background-color:yellow
      }
      .perdida {
        text-align:right;
        background-color:red
      }
    </style>
  </head>
  <body>
    <table>
      <tr>
        <th>Producto</th>
        <th>Ganancia</th>
        <th>Costo</th>
        <th>Diferencia</th>
      </tr>
      <tr>
        <td>Laptop</td>
        <td>$300</td>
        <td>$100</td>
        <td class="ganancia">$200</td>
      </tr>
      <tr>
        <td>Escritorio</td>
        <td>$150</td>
        <td>$150</td>
        <td class="cero">$0</td>
      </tr>
      <tr>
        <td>Silla</td>
        <td>$50</td>
        <td>$300</td>
        <td class="perdida">$250</td>
      </tr>
    </table>
  </body>
</html>
```

Producto	Ganancia	Costo	Diferencia
Laptop	\$300	\$100	\$200
Escritorio	\$150	\$150	\$0
Silla	\$50	\$300	\$250

```
<html>
  <head>
    <style>
      table, td {border:1px solid black; width:80%; height:80%}
      td {width:33.33%; height:33.33%}
      .t {vertical-align:top}
      .m {vertical-align:middle}
      .b {vertical-align:bottom}
      .l {text-align:left}
      .c {text-align:center}
      .r {text-align:right}
    </style>
  </head>
  <body>
    <table>
      <tr>
        <td class="t l">1</td>
        <td class="t c">2</td>
        <td class="t r">3</td>
      </tr>
      <tr>
        <td class="m l">4</td><td class="m c">5</td>
        <td class="m r">6</td>
      </tr>
      <tr>
        <td class="b l">7</td><td class="b c">8</td>
        <td class="b r">9</td>
      </tr>
    </table>
  </body>
</html>
```

1		2	3
4		5	6
7		8	9

Div y Span

- <div> : Áreas grandes
- : Unas cuantas palabras

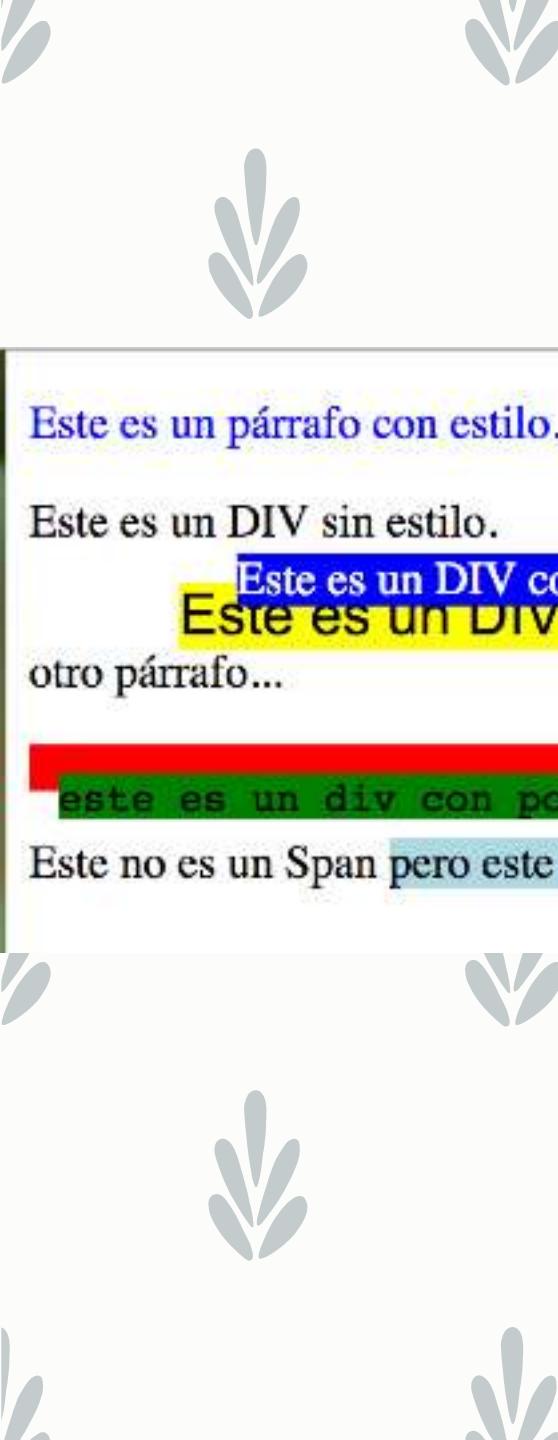
- Estilos:
 - *Font-size*
 - *Font-family*
 - *Background*
 - *Position*
 - *Top*
 - *Left*

DIV

- No tiene estilos por defecto
- No tiene un significado en especial
- Se lo usa para cualquier propósito
- Se lo puede poner en cualquier posición.
 - *position: absolute;*
 - El elemento toma como referencia la ventana del navegador o el elemento posicionado más cercano.
 - *position: relative;*
 - El elemento toma como referencia su posición por defecto.

Span

- No tiene estilos por defecto
- Se lo utiliza sólo para unas cuantas palabras.



Este es un párrafo con estilo.

Este es un DIV sin estilo.

Este es un DIV con posición absoluta
Este es un DIV con estilos

otro párrafo...

este es un div con posicion relativa

Este no es un Span pero este sí es pero este ya no es.

```
<!DOCTYPE html>
<html>
<head>
    <title></title>
</head>
<body>
    <p style="color: blue"> Este es un párrafo con estilo.</p>

    <div>Este es un DIV sin estilo.</div>

    <div style="background: yellow;
        font-size: 20px;
        font-family: Arial;
        position: absolute;
        top: 80px;
        left: 60px;
    ">
        Este es un DIV con estilos</div>
    <div style="background: blue;
        color: white;
        position: absolute;
        top: 70px;
        left: 80px;">Este es un DIV con posición absoluta</div>
    <br>
    <p>otro párrafo...</p>
    <div style="background: red">
        <div style="
            background: green;
            font-size: 14px;
            font-family: courier;
            position: relative;
            top: 10px;
            left: 10px;
        "> este es un div con posicion relativa</div>
    </div>
    <p> Este no es un Span <span style="background: lightblue">pero este sí es</span> pero este ya no es.</p>
</body>
</html>
```

Float

Duis autem vel eum iriure, conseque ad ipsa sed.
et iusto odio dignissim qui blanditiam enim
exercitationem. Nam libero tempore, lucilius
aliquam dolorem ipsum quia dolor sit amet, congue
explicatur. Aliquam fringilla urna, vestibulum
vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in
odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis
dictum nisi, sed ullamcorper ipsum dignissin

Duis autem vel eum iriure, conseque ad ipsa sed.
et iusto odio dignissim qui blanditiam enim
exercitationem. Nam libero tempore, lucilius
aliquam dolorem ipsum quia dolor sit amet, congue
explicatur. Aliquam fringilla urna, vestibulum
vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in
odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis
dictum nisi, sed ullamcorper ipsum dignissin

Duis autem vel eum iriure, conseque ad ipsa sed.
et iusto odio dignissim qui blanditiam enim
exercitationem. Nam libero tempore, lucilius
aliquam dolorem ipsum quia dolor sit amet, congue
explicatur. Aliquam fringilla urna, vestibulum
vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in
odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis
dictum nisi, sed ullamcorper ipsum dignissin

```
<div style="background: red; width: 50%; float: left;">  
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et  
    dictum interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis  
    dolor. Maecenas nisl est, ultrices nec congue eget, auctor vitae massa. Fusce luctus  
    vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in  
    odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis  
    dictum nisi, sed ullamcorper ipsum dignissin  
</div>  
<div style="background: blue; width: 25%; float: right;">  
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et  
    dictum interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis  
    dolor. Maecenas nisl est, ultrices nec congue eget, auctor vitae massa. Fusce luctus  
    vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in  
    odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis  
    dictum nisi, sed ullamcorper ipsum dignissin  
</div>  
<div style="background: green; width: 25%; clear: both;">  
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et  
    dictum interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis  
    dolor. Maecenas nisl est, ultrices nec congue eget, auctor vitae massa. Fusce luctus  
    vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in  
    odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis  
    dictum nisi, sed ullamcorper ipsum dignissin  
</div>
```

Mi Primera Página Web

Summer bootcamp My Primera Página Web

Home

Bienvenido a la primera página web del Summer bootcamp. Aquí puedes ver el trabajo de los participantes y sus profesores. Puedes acceder a las páginas de los diferentes grupos para ver sus trabajos. También puedes ver las páginas de los profesores.

Acerca de mí

Soy un estudiante de informática en la Universidad de Valencia. Me encanta programar y aprender nuevas tecnologías. Estoy interesado en el desarrollo web y la inteligencia artificial. Mi proyecto para este bootcamp es crear una aplicación que permita a los usuarios aprender a través de la programación. Mi objetivo es convertirme en un desarrollador web profesional y contribuir a la sociedad.

Proyectos

Proyecto 1: Una aplicación que permite a los usuarios aprender a través de la programación. El proyecto consiste en una interfaz web que muestra ejemplos de código y explicaciones detalladas. Los usuarios pueden ejecutar el código y ver los resultados.

Proyecto 2: Una aplicación que permite a los usuarios aprender a través de la programación. El proyecto consiste en una interfaz web que muestra ejemplos de código y explicaciones detalladas. Los usuarios pueden ejecutar el código y ver los resultados.

Proyecto 3: Una aplicación que permite a los usuarios aprender a través de la programación. El proyecto consiste en una interfaz web que muestra ejemplos de código y explicaciones detalladas. Los usuarios pueden ejecutar el código y ver los resultados.

Quién soy

Cada persona tiene su propia historia. Aquí te presento mi historia. Nació en Valencia, creció en Madrid y se mudó a Barcelona para estudiar. Es una persona amable, siempre dispuesta a ayudar a los demás. Tiene una gran pasión por la programación y la tecnología. Es un profesional dedicado y comprometido con su trabajo.



livecron

redefine tu carrera, redefine tu vida...

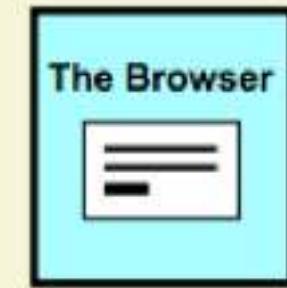
FORMULARIOS

Elementos y atributos:

- <form>
 - Method
 - Action
- <textarea>
- <input type="submit">

Forms are the simplest method

1. The user fills in a form and submits it
2. The browser sends the form data to the server



3. The server receives the data, processes it, sends a response
4. The browser displays the response

Formularios básicos

```
<form action="destination" method="get or post">  
    .... LOS ELEMENTOS DEL FORM VAN AQUÍ....  
    <input type="submit">  
</form>
```

Destination...

-
- Es el programa de destino donde irán los datos del formulario

```
<form action="http://www.server.com/directorio/programa.php">
```

```
<form action="directorio/programa.php">
```

```
<form action="programa.php">
```

Get o Post

-
- Método GET es el que está por defecto.
 - www.google.com
 - GET:
 - Se ven los datos en el URL
 - No se guardan secretos
 - Solamente datos cortos
 - POST:
 - No se ven los datos enviados
 - Datos más grandes

TextArea:

```
<form>  
    <p>Por favor ingresa tu comentario.</p>  
    <textarea rows="3" cols="60" name="feedback" >  
        Ingrresa el texto aquí.  
</textarea>  
</form>
```

Por favor ingresa tu comentario.

Ingrresa el texto aquí.

Más sobre formularios

- <select>
- <option>
- <input type="submit">
- <input type="text">
- <input type="password">
- <input type="checkbox">
- <input type="radio">
- *placeholder*
- *value*
- *autofocus*
- *required*

<textarea> & submit

Ingresá tu comentario.

Ingresá tu texto aquí.

Enviar

```
1 <form action="destination"
2   method="get">
3   <p>Ingresá tu comentario.</p>
4
5   <textarea rows="3" cols="60" name="feedback" >
6     Ingresá tu texto aquí.
7   </textarea>
8   <br>
9   <input type="submit" value="Enviar">
10 </form>
```

Text, checkbox & radio

Por favor ingresa tu nombre.

Por favor selecciona los objetos que tengas:

- Auto
- Osito
- Cepillo de dientes

Por favor ingresa tu nivel de inteligencia.

- Alto
- Medio
- Bajo

```
1 <form>
2   Por favor ingresa tu nombre. <br>
3   <input type="text" name="name"> <br> <br>
4
5   Por favor selecciona los objetos que tengas: <br>
6   <input type="checkbox" name="items" value="auto">Auto <br>
7   <input type="checkbox" name="items" value="osito">Osito <br>
8   <input type="checkbox" name="items" value="cepillo de dientes">Cepillo de
dientes <br> <br>
9
10  Por favor ingresa tu nivel de inteligencia. <br>
11  <input type="radio" name="iq" value="alto">Alto <br>
12  <input type="radio" name="iq" value="medio" checked>Medio <br>
13  <input type="radio" name="iq" value="bajo">Bajo <br> <br>
14 </form>
15
```

Password

Cuál es tu password?

```
1 <form>
2   <p>Cuál es tu password?</p>
3   <input type="password" name="userpassword"> <br>
4 </form>
5
```

Dropdown

Qué ciudad te gustaría visitar?

```
1 <form>
2   <p>Qué ciudad te gustaría visitar?</p>
3
4   <select name="ciudades">
5     <option value="hk">Hong Kong</option>
6     <option value="pa">París</option>
7     <option value="mi">Miami</option>
8   </select>
9 </form>
```

Atributos útiles

- Dar un valor por defecto:
 - value=“algo”
- Texto que desaparece:
 - placeholder = “algo”
- Enfoque inicial: autofocus
- Campos requeridos: required

Por favor llena el siguiente formulario:

Nombre: Adriana

Apellido: Aquí va tu apellido

Edad:

```
1 <form>
2   <p>Por favor llena el siguiente formulario:</p>
3
4   <label for="nombre">Nombre:</label>
5   <input id="nombre" type="text" name="nombre" value="Adriana" autofocus>
6   <br>
7   <label for="apellido">Apellido:</label>
8   <input id="apellido" type="text" name="apellido" placeholder="Aquí va tu
9     apellido">
10  <br>
11  <label for="edad">Edad:</label>
12  <input type="text" name="edad" required>
13  <br>
14  <input type="submit" value="Enviar">
15  </form>
```

Subida de archivos

- El programa del servidor guarda, procesa, mueve y retorna datos.

```
<form method="post" enctype="multipart/form-data"
      action="destination">

    <p>Selecciona el archivo que deseas subir</p>
    <input type="file" name="archivoASubir">

    <p>Presiona el boton para enviar el archivo</p>
    <input type="submit" value="Subir Archivo">

</form>
```

Nuevos elementos HTML5

- <input type="number">
- <input type="date">
- <input type="color">
- <input type="range">
- <input type="time">
- Div span ejemplos
checkbox y password

The form contains the following fields:

- Tu edad:
- Tu fecha de nacimiento:
- A qué hora suena tu despertador?:
- Tu color favorito:
- Tu humor: Malo Bueno
- Enviar!

Tu edad: 18

Tu fecha de nacimiento: dd/mm/aaaa

A qué hora suena tu despertador?: --:--

Tu color favorito:

Tu humor: Malo Bueno

```
1 <form action="destination">
2   <label for="edad">Tu edad:</label>
3   <input type="number" min="0" max="99" step="1"
4     value="18" id="edad" name="edad" required>
5   <br>
6   <label for="cumpleanos">Tu fecha de nacimiento:</label>
7   <input type="date" id="cumpleanos" name="cumpleanos">
8   <br>
9   <label for="despertador">A qué hora suena tu despertador?:</label>
10  <input type="time" id="despertador" name="despertador">
11  <br>
12  <label for="color">Tu color favorito:</label>
13  <input type="color" id="color" name="color">
14  <br>
15  <label for="humor">Tu humor:</label>
16  Malo <input type="range" min="0" max="100" step="5" value="50" id="humor" name="humor"> Bueno
17  <br>
18  <input type="submit" value="Enviar!">
19  </form>
```

AGRUPACIÓN DE ELEMENTOS

Agrupación de elementos: Fieldset

- <fieldset>
- <legend>
- Permite organizar en grupos los campos de un formulario.
- Representa el conjunto de controles en un formulario, agrupados bajo un mismo nombre.
- Usado para formularios grandes.

Formulario 1

Por favor llena los siguientes datos:

Nombre: Adriana

Apellido: Aqui va tu apellido

Edad:

Selecciona una imagen de perfil:

Ningún archivo seleccionado

```
1 <fieldset>
2   <legend>Formulario 1</legend>
3   <form>
4     <p>Por favor llena los siguientes datos:</p>
5     <label for="nombre">Nombre:</label>
6     <input id="nombre" type="text" name="nombre" value="Adriana" autofocus>
7     <br><br>
8     <label for="apellido">Apellido:</label>
9     <input id="apellido" type="text" name="apellido" placeholder="Aqui va tu
apellido">
10    <br><br>
11    <label for="edad">Edad:</label>
12    <input type="text" name="edad" required>
13    <br><br>
14    <p>Selecciona una imagen de perfil:</p>
15    <input type="file" name="archivoASubir">
16    <br>
17    <input type="submit" value="Enviar">
18  </form>
19</fieldset>
20
```

TRABAJANDO EN EL PROYECTO

FORMULARIO WEB

Por favor ingresa tus datos para ser mi amigo:

Tu foto

Tu imagen: Ningún archivo seleccionado

Preview de la Imagen:

Tus datos generales

Nombre:

Género: Masculino Femenino

Edad:

Fecha de nacimiento: / /

Color favorito:

País de procedencia: Ninguno

Algunas cosas importantes...

Altura: Bajo Alto

Personalidad: Negativa Positiva

TU Información de contacto

Email:

Celular:

Dirección:

Cómo te puedo contactar: Email Whatsapp Facebook

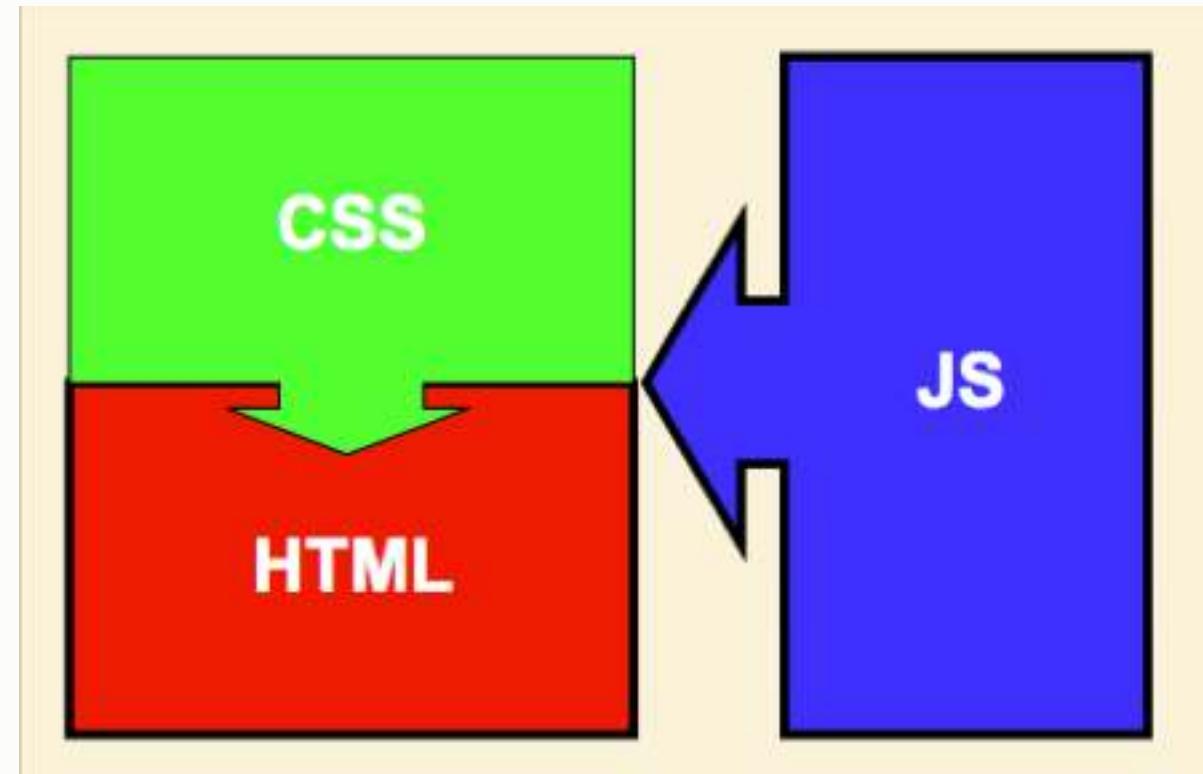
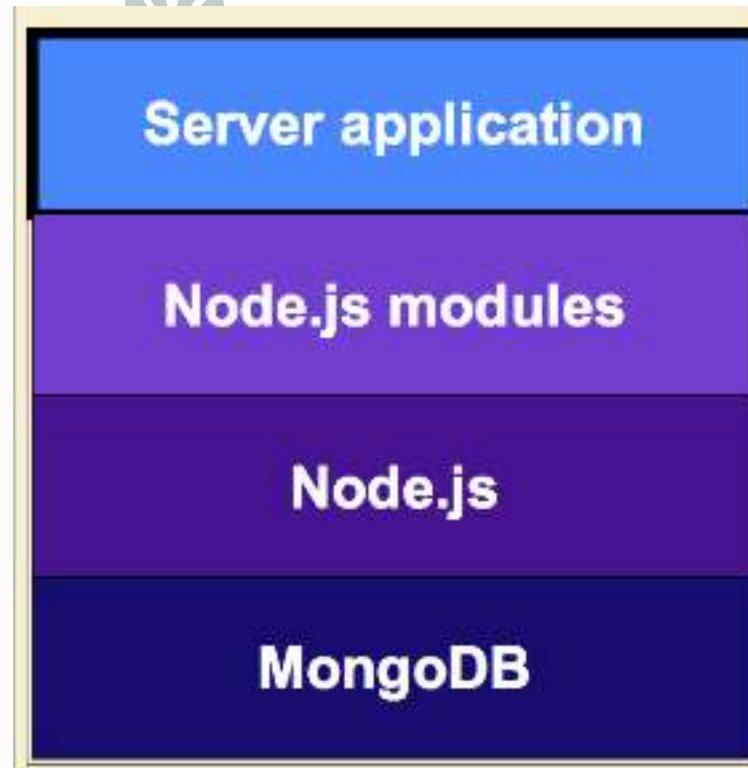
Enviar

INTRODUCCIÓN A JAVASCRIPT

Javascript

- Es el lenguaje de programación Web muy popular y con mayor documentación.
 - Ej: Stack Overflow
- Puede estar tanto del lado del cliente como en el lado del servidor.

Server Side & Client Side



Dónde poner el código JS?

-
- Puede ir en cualquier lugar del código HTML, pero también hay patrones.
 - El código generalmente va al final antes de cerrar el </body>

```
<script>
    function sorpresa() {
        alert("Hello!");
    }
</script>
```

Las librerías JS se cargan en el
<head>

```
<script src="miCodigo.js">
```

```
function sorpresa() {  
    alert("Hola!");  
}
```

Funciones pop-up: alert()

```
<!doctype html>
<html>
  <head>
    <title>Ejemplo de alert()</title>
    <script>
      alert("Bienvenido a mi página!");
    </script>
  </head>
</html>
```

Funciones pop-up: confirm()

```
<!doctype html>
<html>
  <head>
    <title>Ejemplo de confirm()</title>
    <script>
      if (confirm("Te gustaría visitar Disneyland?"))
        document.location.href
        ="https://disneyland.disney.go.com/";
    </script>
  </head>
</html>
```

Funciones pop-up: prompt()

```
<!doctype html>
<html>
  <head>
    <title>Ejemplo de prompt()</title>
    <script>
      var user_name;

      user_name = prompt("Cuál es tu nombre?");
      document.write("Bienvenido a mi página "
                    + user_name + "!" );
    </script>
  </head>
</html>
```

Variable

- Las variables son contenedores
- Puedes crear una variable y poner algo dentro:

```
var costoTotal = 7000;
```

- Se puede obtener los valores del contenedor.
- Se puede cambiar su contenido.
- Por ejemplo en user_name en prompt() ←

BASES DE JAVASCRIPT

Más sobre variables:

- var
- typeof
- TIPOS DE DATOS:
 - Number
 - String
 - Boolean
 - Otros. Ej: Object

Number

- Sólo existe un tipo de número
- Se puede usar decimales

```
var numero1 = 12.97;  
var numero2 = 100;
```

- Se puede usar notación científica

```
var numero_grande = 123e5; //12300000  
var numero_pequeno = 123e-5; //0.00123
```

String

- Es texto
- Se pueden usar comillas dobles o simples

```
var name = "Juan";
var title = 'Dentista';
```

- Podemos usar comillas simples en el texto dentro de un String definido con comillas dobles y viceversa.

```
var message = "I'm Adriana";
```

Boolean

- True o False
- No confundir con Strings

```
var condicion1 = true;  
var condicion2 = false;  
var texto = "true";
```

Las variables pueden cambiar de tipo

```
var valor1 = "Holaaa";  
  
valor1 = 1234;
```

Typeof

Esta página dice:

"Adrita" es del tipo: string

3.14 es del tipo: number

false es del tipo: boolean

Aceptar

```
<!doctype html>
<html>
  <head>
    <title>Ejemplo typeof</title>
  </head>
  <body>
    <script>
      alert( "Adrita" es del tipo: ' + typeof "Adrita" + "\n\n"
        + "3.14 es del tipo: " + typeof 3.14 + "\n\n"
        + "false es del tipo: " + typeof false );
    </script>
  </body>
</html>
```

Cambios comunes

Código	
contador = contador + 1	contador++
contador = contador - 1	contador--
contador = contador + 10	contador += 10
hola = hola + “!”	hola += “!”
numero = numero - 20	numero -= 20
numero = numero * 4	numero *= 4
dulces = dulces / ninios	dulces /= ninios

Convertir de un tipo a otro

Función	Significado
<code>parseInt()</code>	Convierte a un entero
<code>parseFloat()</code>	Convierte a un numero con decimales
<code>String()</code>	Convierte el valor del objeto a String

Eventos y funciones

- Events
- onload
- Functions
- function
- return

Eventos

- Un evento es cuando algo sucede
- Por ejemplo:
 - Hacer un click sobre un elemento
 - Mover el mouse
 - Presionar una tecla
 - Se puede ejecutar partes de código cuando un evento ocurre

Evento: onload

- El evento es lanzado cuando el objeto ha cargado

```
<!doctype html>
<html>
  <body onload="alert('Hola!')">
    <p>Este mensaje se muestra tan pronto como se cargue la página.</p>
  </body>
</html>
```

Funciones

- Una función es un grupo de código:

```
<!doctype html>
<html>
  <head>
    <title>Ejemplo de función</title>
    <script>
      function do_something(){
        ... el código va aquí ...
      }
    </script>
  </head>
  <body onload="do_something()">
  </body>
</html>
```

Parámetros de funciones

```
<!doctype html>
<html>
  <body>
    <script>
      function comprar( gatos ) {
        ... aqui se usa la variable gatos ...
      }

      ... más codigo ...

      comprar(10); //llamando a la funcion comprar
    </script>
  </body>
</html>
```

Respuesta de funciones

```
function hacer_algo() {  
    return "algo";  
}  
  
... código ...  
  
result = hacer_algo(); //llamar a la función |
```

Funciones recursivas

```
<!doctype html>
<html>
  <body>
    <script>
      function hacer_algo( valor_de_control ) {
        ... hago la llamada a la funcion hacer_algo las veces necesarias ...
      }

      ... más código ...

      resultado = hacer_algo(10); //llamar a la función
    </script>
  </body>
</html>
```

Manejando bugs & console.log()



DECISIONES Y LOOPS

Tomando decisiones...

- if
- if ... else
- if ... else ... if
- if ... else ... if ... else
- switch ... case
- default

Operadores condicionales

- < es menor que
- <= es menor o igual que
- > es mayor que
- >= es mayor o igual que
- == es igual a
- != es diferente a

```
<html>
  <head>
    <script>
      var user_name;

      user_name=prompt("Cual es tu nombre?");
      if (user_name == "adri")
        alert("Que lindo nombre!");

    </script>
  </head>
</html>
```

If else

```
<!doctype html>
<html>
  <head>
    <script>
      var user_name;

      user_name=prompt("Cual es tu nombre?");
      if (user_name == "maria filomena")
        alert("que lindo nombre!");
      else
        alert("uhmm tu nombre no es tan lindo...");
    </script>
  </head>
</html>
```

If ... else if ... else

```
<!doctype html>
<html>
  <head>
    <script>
      var user_name;

      user_name=prompt("Cual es tu nombre?");
      if (user_name == "adriana")
        alert("que lindo nombre!");
      else if (user_name == "milenka")
        alert("ese es un nombre muy bonito!");
      else if (user_name == "paola")
        alert("Excelente nombre!");
      else
        alert("uhmm... tu nombre no es muy bueno, ni modo...");
    </script>
  </head>
</html>
```

Switch

```
<!doctype html>
<html>
  <head>
    <script>
      var user_name = prompt("Cual es tu nombre?");

      switch(user_name) {
        case "adri":
          alert("que buen nombre!");
          break;
        case "milenka":
          alert("un nombre muy lindo!");
          break;
        default:
          alert("uhmm... tu nombre no es tan bueno, no importa...");
      }
    </script>
  </head>
</html>
```

Manejando LOOP WHILE

- Un loop repite el código una y otra vez de acuerdo a una condición.
- Los loops que veremos:
 - While
 - Do ... while

While & indexof()

```
<!doctype html>
<html>
  <head>
    <title>Ejemplo de while()</title>
    <script>
      var response, finished;
      finished=false;
      alert("La UMSS es la mejor universidad.");
      while (!finished){
        response=prompt("Estas de acuerdo?");
        if (response.indexOf("s")=0)
          finished=true;
      }
    </script>
  </head>
</html>
```

do ... while: ejecuta el código por lo menos una vez

```
<!doctype html>
<html>
  <head>
    <title>Ejemplo de do .. while()</title>
    <script>
      var response, finished;
      finished=false;
      alert("La UMSS es la mejor universidad.");
      do {
        response=prompt("Estas de acuerdo?");
        if (response.indexOf("s")==0)
          finished=true;
      } while (!finished);
    </script>
  </head>
</html>
```

MANEJANDO INFORMACIÓN

Más sobre variables: Locales

- Declaradas dentro de una función
- Sólo se puede acceder a ellas dentro de la función.

```
<!doctype html>
<html>
  <body>
    <script>
      function show_money() {
        var money = 2;

        alert("En la funcion, el valor es: "+money);
      }
      money = 99;

      alert("En el main, el valor es: "+money);
      show_money();
      alert("En el main, el valor es: "+money);

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

Más sobre variables: Globales

- Declaradas dentro del main
- Se puede acceder a ellas dentro y fuera de las funciones.

```
<!doctype html>
<html>
  <body>
    <script>
      function show_money() {
        alert("En la funcion, el valor es: "+ money);
      }
      var money = 99;

      alert("En la parte del main, el valor es: "+ money);
      show_money();
      alert("En la parte del main, el valor es: "+ money);
    </script>
  </body>
</html>
```

Variables globales y locales con el mismo nombre.

- JavaScript le dará prioridad a la variable local dentro de la función

```
<!doctype html>
<html>
<body>
  <script>
    function show_money() {
      var money = 2;

      alert("En la funcion, el valor es: "+money);
    }
    money = 99;
    alert("En la parte del main, el valor es: "+money);
    show_money();
    alert("En la parte del main, el valor es: "+money);
  </script>
</body>
</html>
```

Creando Variables Globales dentro de funciones.

- Si asignamos un valor a una variable que no ha sido declarada, automáticamente se convertirá en una variable global

```
<!doctype html>
<html>
  <body>
    <script>
      function show_money() {
        money = 2;

        alert("En la funcion, el valor es: "+money);
      }
      show_money();
      alert("En la parte del main, el valor es: "+money);
    </script>
  </body>
</html>
```

Operadores lógicos

- Valores Booleanos:

- True
 - False

- Operadores lógicos:

- && (and)
 - || (or)
 - ! (not)

AND - &&

a	b	a && b
false	false	false
false	true	false
true	false	false
true	true	true

```
<!doctype html>
<html>
  <body>
    <script>
      var tienes_dinero = false;
      var tienes_amor = true;
      var tienes_salud = true;
      var tienes_la_mejor_vida = tienes_dinero
        && tienes_amor
        && tienes_salud;

      alert("tener una buena vida es posible: " +
            tienes_la_mejor_vida);
      tienes_dinero = true;
      tienes_la_mejor_vida = tienes_dinero
        && tienes_amor
        && tienes_salud;
      alert("la vida es mejor ahora: " +
            tienes_la_mejor_vida);

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

JavaScript evalúa el primer input dentro de una comparación con &&

```
<!doctype html>
<html>
  <body>
    <script>
      function primera_funcion() {
        alert("primera_funcion() esta corriendo!");
        return true;
      }
      function segunda_funcion() {
        alert("segunda_funcion() esta corriendo!");
        return false;
      }
      var test_funcion =
        primera_funcion() && segunda_funcion();
    </script>
  </body>
</html>
```

Cambiando el orden...

```
<!doctype html>
<html>
  <body>
    <script>
      function primera_funcion() {
        alert("primera_funcion() esta corriendo!");
        return true;
      }
      function segunda_funcion() {
        alert("segunda_funcion() esta corriendo!");
        return false;
      }
      var test_function_cambiada =
        segunda_funcion() && primera_funcion();
    </script>
  </body>
</html>
```

OR - ||

a	b	a b
false	false	false
false	true	true
true	false	true
true	true	true

```
<!doctype html>
<html>
  <body>
    <script>
      var tienes_dinero = false;
      var tienes_amor = true;
      var tienes_salud = false;
      var tienes_la_mejor_vida = tienes_dinero
                                || tienes_amor
                                || tienes_salud;

      alert("tener una buena vida es posible: " + tienes_la_mejor_vida);
      tienes_amor = false;
      tienes_la_mejor_vida = tienes_dinero
                            || tienes_amor
                            || tienes_salud;
      alert("la vida es mejor ahora: " + tienes_la_mejor_vida);

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

JavaScript evalúa el primer input dentro de una comparación con ||

```
<!doctype html>
<html>
  <body>
    <script>
      function primera_funcion() {
        alert("primera_funcion() esta corriendo!");
        return true;
      }
      function segunda_funcion() {
        alert("segunda_funcion() esta corriendo!");
        return false;
      }
      var test_function =
        primera_funcion() || segunda_funcion();
    </script>
  </body>
</html>
```

NOT !

a	!a
false	true
true	false

```
<!doctype html>
<html>
  <head>
    <title>Ejemplo operador Negativo</title>
  </head>
  <body>
    <script>
      var eres_hombre = true;
      var eres_mujer = !eres_hombre;

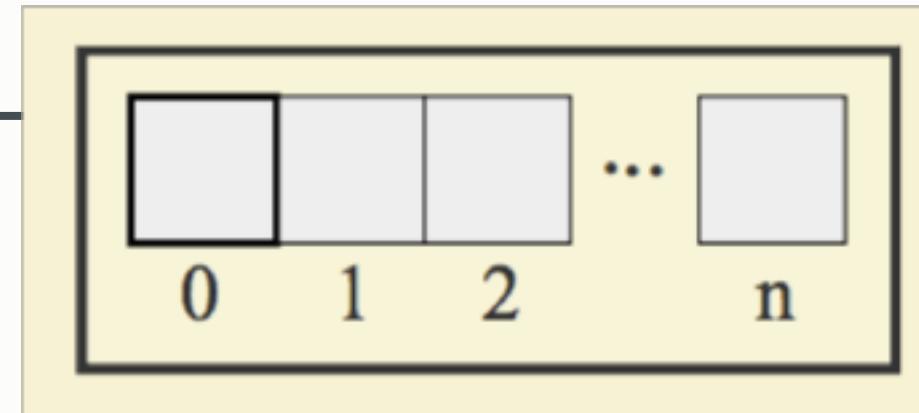
      alert("eres_hombre is " + eres_hombre);
      alert("eres_mujer is " + eres_mujer);
    </script>
  </body>
</html>
```

Arreglos

– Funciones de los arreglos:

- []
- push()
- concat()
- length
- shift()
- join()
- pop()
- unshift()

Arreglo



- Conjunto de elementos ordenados
- Cada elemento de un arreglo es una entidad única
- Cada elemento es accedido por un index.
- El index de la primera posición es 0.

Creando un arreglo:

- Un arreglo de 3 elementos definidos:

```
var mascotas = ["perro", "gato", "conejo"];
```

- Arreglo de 10 elementos:

```
var mascotas = new Array(10);
```

- Un arreglo puede contener objetos de cualquier tipo.

Join()

- Join es un separador que sirve para convertir el arreglo en un string

```
var mascotas = ["perro", "gato", "conejo"];
alert(mascotas.join(" y "));
// Esto muestra "perro y gato y conejo"
```

- El separador por defecto es “,”

```
var mascotas = ["perro", "gato", "conejo"];
alert(mascotas.join());
// Esto muestra "perro,gato,conejo"
```

Obteniendo valores

- Con este array:

```
var mascotas = ["perro", "gato", "conejo"];
```

- Podemos obtener un valor de una casilla específica:

```
alert(mascotas[2]); //esto muestra "conejo"
```

Cambiando valores

- Con este array:

```
var mascotas = ["perro", "gato", "conejo"];
```

- Podemos cambiar el valor de uno de sus elementos:

```
mascotas[2] = "gallina";
// ahora mascotas es ["perro", "gato", "gallina"]
```

Tamaño del arreglo

- Podemos obtener el tamaño de un arreglo usando array.length:

```
var mascotas = ["perro", "gato", "conejo"];
alert(mascotas.length); // esto muestra 3
```

Añadiendo elementos al final

- Se pueden agregar elementos al arreglo con array.push()

```
var mascotas = ["perro", "gato", "conejo"];
mascotas.push("gallina");
// ahora mascotas es ["perro", "gato", "conejo",
"gallina"]
```

- Los index seran actualizados automaticamente

Añadiendo elementos al inicio

- Se pueden agregar elementos al inicio del arreglo con array.unshift()

```
var mascotas = ["perro", "gato", "conejo"];
mascotas.unshift("gallina");
// ahora mascotas es ["gallina", "perro", "gato",
"conejo"]
```

- Los index son actualizados automáticamente

Eliminando del final

- Para eliminar elementos del final se usa array.pop()

```
var mascotas = ["perro", "gato", "conejo"];
var result = mascotas.pop();
// ahora mascotas es ["perro", "gato"]
```

- Pop() devuelve el elemento eliminado

Eliminando del inicio

- Array.shift() elimina el primer elemento:

```
var mascotas = ["perro", "gato", "conejo"];
var result = mascotas.shift();
// ahora mascotas es ["gato", "conejo"]
```

- Shift() devuelve el elemento eliminado
- Los index son actualizados automaticamente

Combinando 2 arreglos

- Array1.concat(array2) combina 2 arreglos en 1

```
var mascotas = ["perro", "gato", "conejo"];
var edades = [2, 3, 5];
var result = mascotas.concat(edades);
// ahora mascotas es ["perro", "gato", "conejo",
2, 3, 5]
```

GENERACIÓN DE NÚMEROS RANDÓMICOS

Math.random()

- Math.random()
- El resultado es un número en el rango [0,1)
- 1 no va a ser generado
- Se multiplica por el máximo valor para obtener el nuevo rango.

```
<!doctype html>
<html>
  <body>
    <script>
      var random_number;

      random_number = Math.random() * 8;
      alert("Un numero randomico entre 0 y 7.9999999 es:\n" +
            random_number );
    </script>
  </body>
</html>
```

Math.floor()

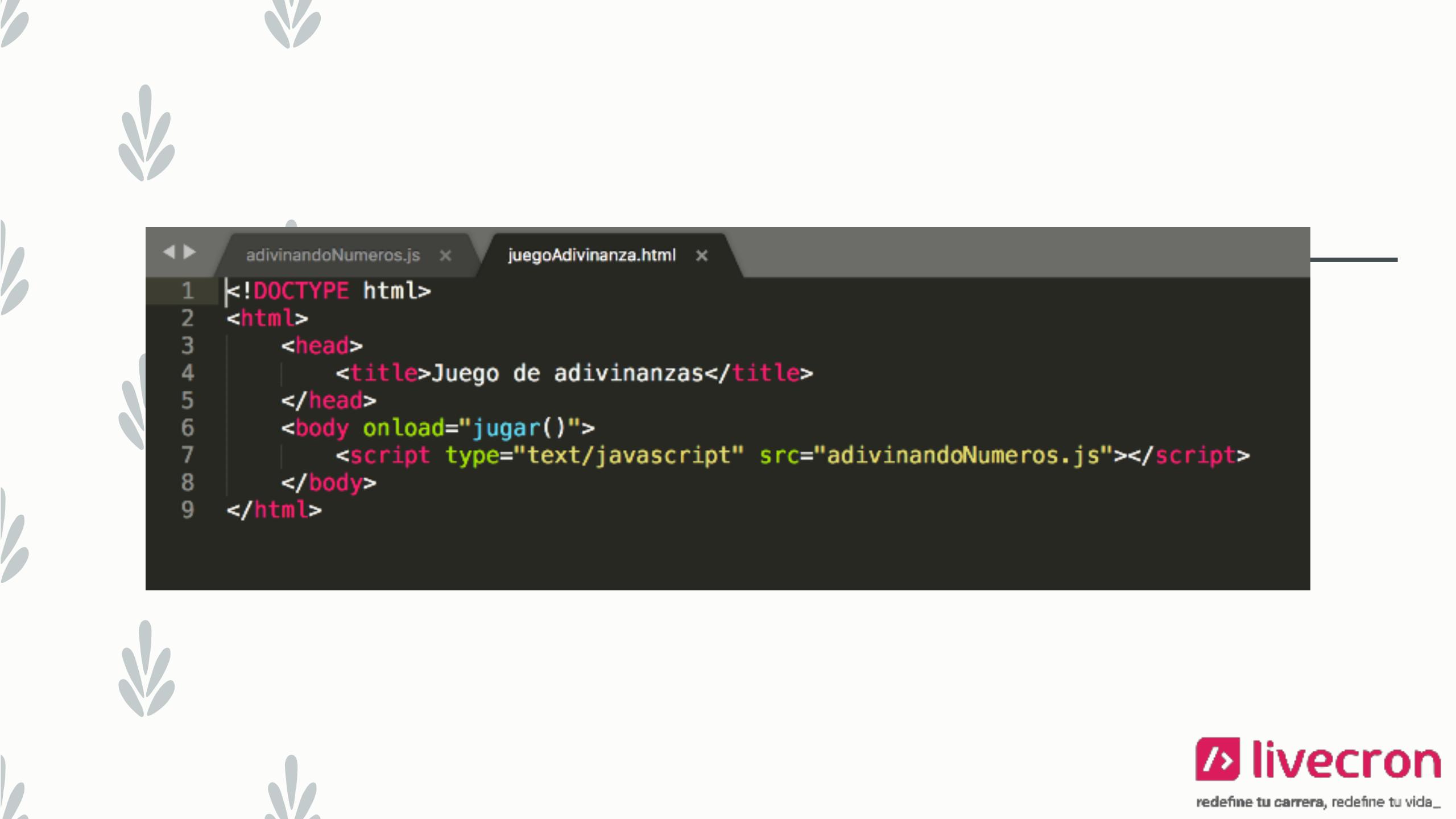
- Elimina los decimales de un número.
- Si tengo el número 3.821468, se convertirá en 3

```
<!doctype html>
<html>
  <body>
    <script>
      var random_number;

      random_number = Math.random() * 50;
      random_number = Math.floor( random_number );
      alert("Un número randómico entre 0 y 49: " +
            random_number);

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

EJEMPLO DE UN PROYECTO JAVASCRIPT



```
adivinandoNumeros.js x juegoAdivinanza.html x
1 <!DOCTYPE html>
2 <html>
3     <head>
4         <title>Juego de adivinanzas</title>
5     </head>
6     <body onload="jugar()">
7         <script type="text/javascript" src="adivinandoNumeros.js"></script>
8     </body>
9 </html>
```

```
adivinandoNumeros.js x juegoAdivinanza.html x
1 var objetivo;
2 var terminar = false;
3 var adivinanza_texto;
4 var adivinanza_numero;
5 var intentos = 0;
6
7 function jugar() {
8     var numero_randomico = Math.random() * 101;
9     var numero_randomico_entero = Math.floor(numero_randomico);
10    objetivo = numero_randomico_entero;
11    console.log("numero randomico: "+objetivo);
12    while(!terminar) {
13        adivinanza_texto = prompt("Estoy pensando en un numero en el rango de 1 a 100. \n\n"+
14                                "Cual es el numero??");
15        adivinanza_numero = parseInt(adivinanza_texto);
16        intentos++;
17        terminar = verificarNumero();
18    }
19}
20
21 function verificarNumero() {
22    if ( isNaN(adivinanza_numero)){
23        alert("No pusiste un numero. Por favor ingresa un numero entre 1 y 100.");
24        return false;
25    }
26    if( (adivinanza_numero < 1) || (adivinanza_numero>100)){
27        alert("Por favor ingresa un numero entre 1 y 100.");
28        return false;
29    }
30    if( adivinanza_numero < objetivo ) {
31        alert("Tu numero es muy pequenio!!!");
32        return false;
33    }
34    if( adivinanza_numero > objetivo ) {
35        alert("Tu numero es muy grande!!!");
36        return false;
37    }
38    alert("Lo hiciste!! el numero era: "+ objetivo +"\n\n Te tomo: "+ intentos +
39          " intentos adivinar el numero.");
40    document.body.style.background = "orange";
41    return true;
42 }
```

MÁS SOBRE LOOPS

Manejando LOOP FOR

-
- For
 - For ... in
 - For ... of

FOR

- Utiliza la posición inicial y final para realizar el recorrido
- Se lo utiliza para realizar recorridos en series de datos
- Data_structure.length nos indica el número de ítems dentro de nuestra estructura.

For

```
<!doctype html>
<html>
  <head>
    <script>
      var continentes = ["Australia", "Africa",
                          "Antartica", "Europa", "America"];
      var respuesta, count = 0;

      for (var index=0; index < continentes.length;
            index++) {
        respuesta = confirm("Visitaste " +
                            continentes[index] + "?");
        if (respuesta) count++;
      }
      alert("Visitaste " + count +
            " continentes!");
    </script>
  </head>
</html>
```

For ... in: Devuelve el índice de cada elemento

```
<!doctype html>
<html>
  <head>
    <script>
      var continentes = ["Australia", "Africa",
                          "Antartica", "Europa", "America"];
      var respuesta, contador=0;

      for (var index in continentes) {
        respuesta = confirm("Visitaste "
                            + continentes[index] + "?");
        if (respuesta) contador++;
      }
      alert("Visitaste " + contador +
            " continentes!");
    </script>
  </head>
</html>
```

For ... in: Puede ser usado para acceder al contenido de una estructura de datos

```
<!doctype html>
<html>
  <head>
    <title>Ejemplo de for in</title>
    <script>
      var persona = { iniciales:"AP", edad:15,
                      trabajo:"Ingeniera" };

      for (var propiedad in persona) {
        alert(propiedad + "=" + persona[propiedad]);
      }
    </script>
  </head>
</html>
```

For ... of: Devuelve cada ítem

```
<!doctype html>
<html>
  <head>
    <title>Ejemplo de for of</title>
    <script>
      var continentes = ["Australia", "Africa",
                          "Antartica", "Europa", "America"];
      var respuesta, contador = 0;

      for (var continente of continentes) {
        respuesta = confirm("Visitaste " +
                            continente + "?");
        if (respuesta) contador++;
      }
      alert("Visitaste " + contador + " continentes!");
    </script>
  </head>
</html>
```

Omitiendo partes:

- Las 3 partes de la cabecera del for pueden ser omitidas.
- Por ejemplo este seria un loop infinito:

```
for ( ; ; ){
    alert ("bienvenidos!");
}
```

Omitiendo partes:

```
var numero = 1;
for( ; numero <= 12; numero++) {
    alert(numero + " veces 5 = ", numero * 5);
}

-----
for (var conejos = 2, generaciones = 1;
generaciones <= 12;
generaciones++, conejos *= 2 ){
    alert("generaciones: "+ generaciones + " total: "+ conejos);
}
```

Control de LOOPs: Break & Continue

- Break: Detiene el loop en su totalidad
- Continue: Detiene la iteración actual.

MÁS SOBRE ARRAYS

sort()

indexOf()

slice()

reverse()

lastIndexOf()

splice()

Manejando Arrays: sort()

- Ordena los elementos en un Array

```
var girls = ["cecilia", "adriana", "paola", "ximena", "laura"];
girls.sort();
//ahora girls es: ["adriana", "cecilia", "laura", "paola", "ximena"]
```

Manejando Arrays: reverse()

- Devuelve el array en orden reverso

```
var girls = ["cecilia", "adriana", "paola", "ximena", "laura"];
girls.reverse();
//devuelve: ["laura", "ximena", "paola", "adriana", "cecilia"]
```

Manejando Arrays: sort() & reverse()

- Combinando sort() y reverse() se puede ordenar listas en orden descendente

```
var girls = ["cecilia", "adriana", "paola", "ximena", "laura"];
girls.sort().reverse();
//devuelve: ["ximena", "paola", "laura", "cecilia", "adriana"]
```

Manejando Arrays: indexOf()

- Se usa array.indexOf(Objetivo) para encontrar el index del primer elemento objetivo en un arreglo.

```
var girls = ["cecilia", "adriana", "paola", "ximena"];
alert(girls.indexOf("paola"));
//devuelve: 2
```

- Si el objetivo no esta dentro del arreglo, devuelve -1

Manejando Arrays: indexOf(objetivo, posición)

- Se puede enviar un segundo valor a indexOf() para controlar dónde empezar la búsqueda.

```
array.indexOf(target, posicionInicial)
```

Ejemplo búsqueda con indexOf()

```
var frutas = ["manzana", "durazno", "pera", "manzana", "uva", "manzana"];

var posicionesManzana = [], empezarBusquedaEn = 0;

do {
    encontradaEn = frutas.indexOf("manzana", empezarBusquedaEn);
    if(encontradaEn != -1) {
        posicionesManzana.push(encontradaEn);
        empezarBusquedaEn = encontradaEn + 1;
    }
} while(encontradaEn != -1);

alert(posicionesManzana); // Devuelve 0,3,5
```

Manejando Arrays: lastIndexOf()

- Se usa array.lastIndexOf(Objetivo) para encontrar el objetivo en un arreglo, empezando desde el último elemento.

```
var frutas = ["manzana", "durazno", "pera", "manzana", "uva"];
alert(frutas.lastIndexOf("manzana")); //Devuelve 3
```

Manejando Arrays: slice()

- Se puede obtener parte de un arreglo usando slice() con la posición donde empezará.

```
var frutas = ["manzana", "durazno", "pera", "uva"];
var frutero = frutas.slice(1) //Devuelve ["durazno", "pera", "uva"]
```

- También se puede obtener el arreglo indicando la posición de inicio y la posición final.

```
var frutas = ["manzana", "durazno", "pera", "uva"];
```

```
var frutero = frutas.slice(1,3); //Devuelve ["durazno", "pera"]
```

Manejando Arrays: splice()

- Splice() se utiliza para eliminar elementos de cualquier posición dentro de un arreglo.
- Se envían como parámetros la posición y la cantidad de elementos.

```
var frutas = ["manzana", "durazno", "pera", "uva"];
var frutero = frutas.splice(1,1);
//frutas ["manzana", "pera", "uva"]
//frutero es ["durazno"]
```

- Splice() devuelve los elementos eliminados

Manejando Arrays: splice()

- También se usa splice() para añadir elementos
- Se utiliza: array.splice(posicion, 0, elemento)

```
var frutas = ["manzana", "durazno", "uva"];
var frutero = frutas.splice(2, 0, "pera");
//frutas ["manzana", "durazno", "pera", "uva"]
//frutero es []
```

- Como no se elimina nada, el resultado es vacío

Manejando Arrays: reemplazando elementos

- Para reemplazar elementos se usa:
 - array.splice(posición, cantidad, elemento(s))

```
var frutas = ["manzana", "durazno", "uva"];  
  
var frutero = frutas.splice(1, 1, "pera", "sandia");  
//frutas ["manzana", "pera", "sandia", "uva"]  
//frutero es ["durazno"]
```

Funciones de Arrays: forEach()

- Se puede acceder a los elementos de un arreglo usando el loop (for/while)

```
var frutas = ["manzana", "durazno", "uva"];  
  
for(var i = 0; i < frutas.length; i++){  
    alert(frutas[i]);  
}
```

- También se puede acceder a los elementos utilizando array.forEach(function)

```
var frutas = ["manzana", "durazno", "uva"];  
  
frutas.forEach(alert);
```

Funciones de Arrays: forEach()

- Se puede pensar al forEach() de esta manera:

```
function forEach(arreglo, fn){  
    for(var i = 0; i < arreglo.length; i++){  
        fn(arreglo[i], i, arreglo);  
    }  
}
```

- Entonces, la función necesita 3 parámetros:

```
function tuFuncion(elemento, index, arreglo){}
```

Ejemplo:

```
<!doctype html>
<html>
  <body>
    <script>
      var numeros = [1, 2, 3, 4, 5];

      numeros.forEach( function(elem, idx, arr) {
          arr[idx] = elem * elem;
      });
      alert(numeros); // esto muestra [1,4,9,16,25];
    </script>
  </body>
</html>
```

Funciones de Arrays: map()

- La función map(función) guarda el resultado de cada ejecución de la función en un arreglo, el cual será devuelto.

```
function map(arreglo, fn){  
    var resultados = [];  
    for(var i = 0; i < arreglo.length; i++){  
        resultados.push(fn(arreglo[i], i, arreglo));  
    }  
    return resultados;  
}
```

Funciones de Arrays: map()

```
<!doctype html>
<html>
  <body>
    <script>
      var cuadrado = function(num) { return num * num; }
      var numeros = [1, 2, 3, 4, 5];
      var resultados = numeros.map(cuadrado);

      alert(resultados); // muestra [1,4,9,16,25];
    </script>
  </body>
</html>
```

EL DOM

Conceptos básicos

- DOM: Significa Document Object Model
- Todo lo que cargamos en el Browser es convertido a una estructura DOM.
- El primer nodo es la raíz.
- De la raíz salen cuantas ramas sean necesarias.
- Un nodo puede tener nodos hijos y un nodo padre.

Ejemplo:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Una página web básica</title>
    <meta name="author" content="AMPC">
  </head>
  <body>
    <h1>Mi pagina web</h1>
    <p>mi super pagina !!</p>
  </body>
</html>
```

Ejemplo 2:

```
<!DOCTYPE html>
<html>
  <body>
    <table>
      <tbody>
        <tr>
          <td>Adri</td>
          <td>15</td>
        </tr>
        <tr>
          <td>Mile</td>
          <td>16</td>
        </tr>
      </tbody>
    </table>
  </body>
</html>
```

Ejemplo 3:

```
<!DOCTYPE html>
<html>
  <body id="theBody"><p id="firstP">
    Hola!
    </p>
    Qué tal?
    <br>
    <p id="secondP">
      Ten un buen dia :))!
    </p>
  </body>
</html>
```

Ejemplo 4:

```
<!DOCTYPE html>
<html>
  <body id="theBody">
    <p id="firstP">
      Hola!
    </p>
    Que tal?
  </body>
</html>
```

Relaciones de Nodos

- Desde un nodo se puede acceder a:
 - parentNode
 - childNodes[]
 - firstChild
 - lastChild
 - previousSibling
 - nextSibling

Encontrando el PATH de un Nodo

1. La función comienza con un nodo
2. El tipo de nodo es añadido a un String
3. El código se va al padre del nodo actual
4. Si el nodo tiene un parente, se repite 2. y 3.

El código:

```
function handleClick(event) {  
    event.stopPropagation();  
  
    var node = event.target  
    var thisPath = node.nodeName;  
  
    while (node.parentNode) {  
        node = node.parentNode;  
        thisPath = node.nodeName + " > " + thisPath;  
    }  
    alert(thisPath);  
}
```

Insertando el handler:

```
// Asigna click event handler para todos los nodos
function attachHandler(node) {
    if(node == null) return;
    node.onclick = handleClick;

    for(var i = 0; i < node.childNodes.length; ++i) {
        visit(node.childNodes[i]);
    }
}

// Llama a la funcion que anade handler para cada elemento.
function init() {
    attachHandler(document.getElementsByTagName("body")[0]);
}
```

Localizar nodos

-
- Podemos añadir/copiar/cambiar cualquier nodo dentro del DOM
 - Usamos:
 - getElementsByTagName()
 - getElementById()
 - setAttribute()

Localizar nodos:

- Método 1: Usar el path exacto en el DOM (no es muy seguro y puede cambiar de browser a browser)
- Método 2: Usar el tipo con `getElementsByName()` (puede haber mas de un elemento del mismo tipo y debemos estar seguros de saber cual es exactamente)
- Método 3: Usar el valor del id con `getElementById()`

Ejemplo:

```
<html>
  <head>
    <script>
      function cambiar_color1() {
        document.childNodes[1].childNodes[2].childNodes[1].style.
          color="red";
      }

      function cambiar_color2() {
        document.getElementsByTagName("h2")[0].style.color="yellow";
      }

      function cambiar_color3() {
        document.getElementById("cute_text").style.color="blue";
      }
    </script>
  </head>

  <body>
    <h2 style="color:black" id="cute_text">
      Haz click en el boton para cambiar color al texto
    </h2>
    <form>
      <input onclick="cambiar_color1()" type="button" value="Cambiar
      color usando metodo 1">
      <input onclick="cambiar_color2()" type="button" value="Cambiar
      color usando metodo 2">
      <input onclick="cambiar_color3()" type="button" value="Cambiar
      color usando metodo 3">
    </form>
  </body>
</html>
```

Crear y añadir Nodos

- Primero se crea el nodo que queramos agregar con:
 - createElement()
 - createTextNode()
- Y luego se agrega el nodo al DOM en el lugar deseado con:
 - insertBefore()
 - appendChild()

Ejemplo insertBefore():

```
<!DOCTYPE html>
<html>
  <head>
    <script>
      function insert_new_text() {
        var newItem = document.createElement("hr");
        var destParent = document.getElementsByTagName("body")[0];
        destParent.insertBefore(newItem, destParent.firstChild);
      }
    </script>
  </head>

  <body onclick="insert_new_text()">
    <h1 id="my_text">Por favor haz click en la pagina.</h1>
  </body>
</html>
```

Ejemplo appendChild():

```
<!DOCTYPE html>
<html>
  <head>
    <script>
      function insert_new_text (){
        var newText = document.createTextNode(
          "Este texto es aniadido dinamicamente!");
        var textpart = document.getElementById("my_text");
        textpart.appendChild(newText);
      }
    </script>
  </head>

  <body onclick="insert_new_text()">
    <h1 id="my_text">Por favor haz click en la pagina.</h1>
  </body>
</html>
```

Eliminar Nodos

- Hay 3 formas de eliminar un nodo específico:
-

```
function eliminar1()
{
    var the_node=document.getElementById("firstP");
    the_node.parentNode.removeChild(the_node);
}
function eliminar2()
{
    var the_node=document.getElementsByTagName("p")[0];
    the_node.parentNode.removeChild(the_node);
}
function eliminar3()
{
    var the_parent=document.getElementById("theBody");
    the_parent.removeChild(the_parent.firstChild);
}
```

Eliminar todos los Nodos:

```
<html>
  <head>
    <script>
      function delete_all_children() {
        var theNode = document.getElementById("theBody");
        while (theNode.firstChild)
          theNode.removeChild(theNode.firstChild);
      }
    </script>
  </head>

  <body id="theBody">
    Que tal?
    <br>
    <p id="secondP">Que tengas un buen dia!</p>
    <button type="button" onclick="delete_all_children()">
      Eliminar children
    </button>
  </body>
</html>
```

Clonar Nodos: cloneNode() && appendChild()

```
<!DOCTYPE html>
<html>
  <body>
    <script>
      function myFunction() {
        var the_node=document.getElementById("myList").lastChild;
        var the_clone=the_node.cloneNode();
        document.getElementById("myList").appendChild(the_clone);
      }
    </script>
    <ul id="myList"><li>Hola</li><li>Buenos dias</li></ul>
    <p>Haz click en el boton para: cloneNode()</p>
    <button onclick="myFunction()">Copiar!</button>
  </body>
</html>
```

Clonar Ramas: cloneNode(true) && appendChild()

```
<html>
  <body>
    <script>
      function myFunction() {
        var the_node = document.getElementById("myList").lastChild;
        var the_clone = the_node.cloneNode(true);
        document.getElementById("myList").appendChild(the_clone);
      }
    </script>
    <ul id="myList"><li>Hola</li><li>Buenos dias</li></ul>
    <p>Haz click en el boton para: cloneNode(true)</p>
    <button onclick="myFunction()">
      Copiar!
    </button>
  </body>
</html>
```

MÁS SOBRE EVENTOS

Eventos del mouse: onclick , onmousedown, onmouseup

```
<!DOCTYPE html>
<html>
  <body>
    <script>
      function good_choice() {
        alert("Buena elección!");
      }
      function bad_choice() {
        alert("No estoy de acuerdo!");
      }
    </script>
    <h1>Haz Click en la mejor red social...</h1>
    
    
    
  </body>
</html>
```

Eventos del mouse: onmouseover, onmouseout

```
<!DOCTYPE html>
<html>
  <body>
    <script>
      function change_colour( new_colour ) {
        document.getElementById("myDiv")
          .style.background = new_colour;
      }
    </script>
    <div id="myDiv"
      style="position:absolute; background:yellow;
              left:300; top:100; width:300; font-size:
              52pt"
      onmouseover="change_colour('red');"
      onmouseout="change_colour('yellow');">
      Pon el mouse encima...
      retira el mouse...
    </div>
  </body>
</html>
```

Eventos con Timer

-
- Se inicia un timer de esta manera:
 - `var the_timer = setTimeout(do_something, 1000)`
 - Se detiene un timer de esta manera:
 - `clearTimeout(the_timer)`
 - Para repetir un intervalo de tiempo:
 - `The_timer = setInterval(do_something, 2000);`
 - Para detener el intervalo:
 - `clearInterval(the_timer);`

Añadir eventos usando Javascript

- Añadir un handler con HTML
- Añadir un handler usando Javascript (2 maneras)
- Eliminar un handler event

Más sobre funciones

- Pasando una función a otra función
- Devolviendo una función de una función

TRABAJANDO EN EL PROYECTO

JUEGO INTERACTIVO

JavaScript Alert

X

Congratulations! You have guessed the color!

It took you 4 guesses to finish the game!

You can see the colour in the background.

OK