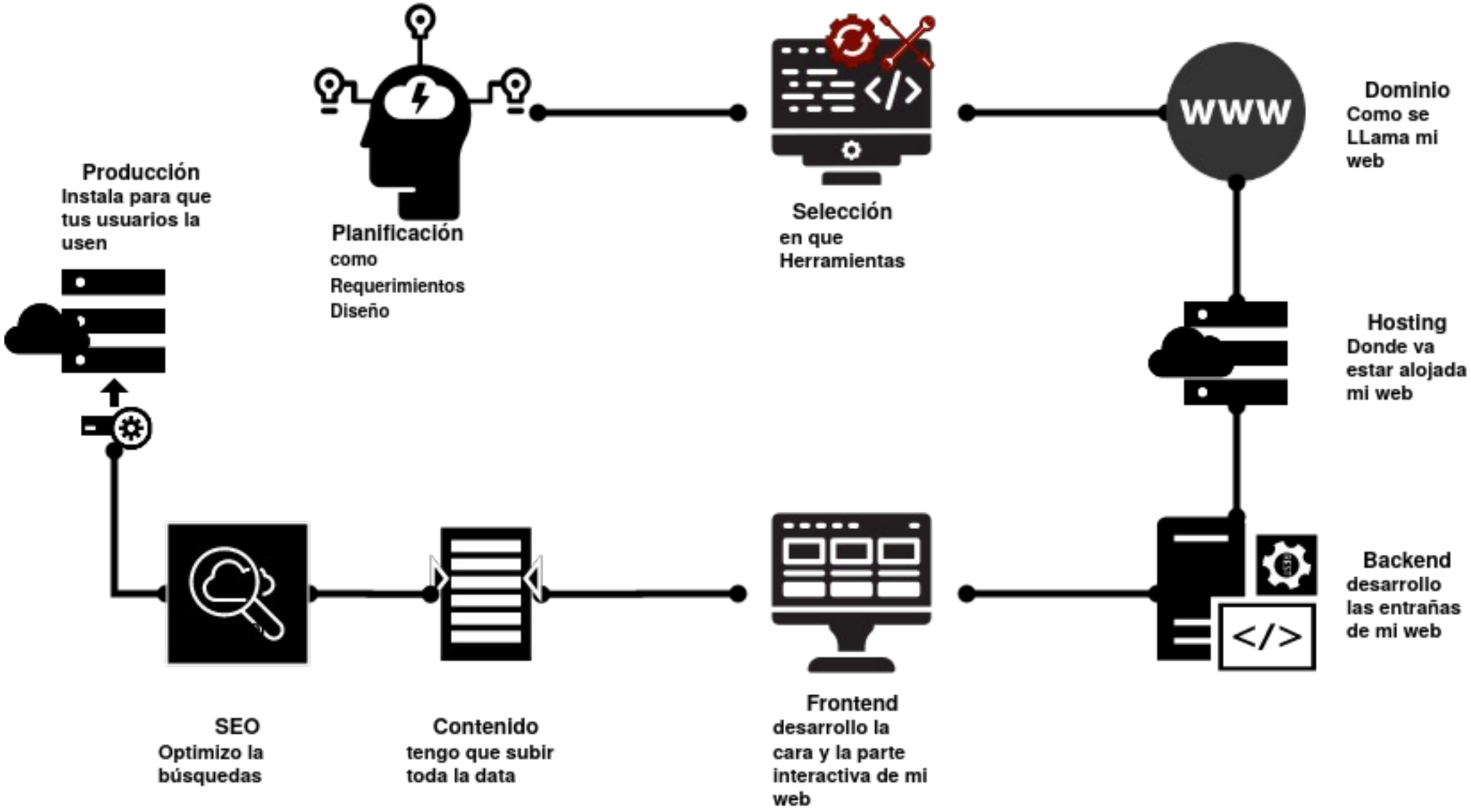
Universidad Industrial de Santander

# HTML
# CSS
# JAVASCRIPT


Programación en la Web
Escuela Ingeniería de Sistemas e Informática
Jathinson Meneses Mendoza
Jathinson@gmail.com

# Paginas Web


Producción
Instala para que tus usuarios la usen
Planificación
como
Requerimientos
Diseño
Selección
en que
Herramientas
www
Dominio
Como se LLama mi web
Hosting
Donde va estar alojada mi web
Backend
desarrollo las entrañas de mi web
</>
Frontend
desarrollo la cara y la parte interactiva de mi web
Contenido
tengo que subir toda la data
SEO
Optimizo la búsquedas

# Paginas Web

## FRONTEND

Programación que está a la vista del usuario,
Lo que se ejecuta en la parte del cliente
La capa de presentación o capa UI (User Interface).

### Tareas

La lógica de la presentación
El diseño de la interfaz de la aplicación
El diseño de interacción, Relación del usuario con la aplicación a través del navegador.
El diseño de la presentación de la información, según los objetivos de la aplicación.

### Add Conocimientos

Diferentes lenguajes, frameworks y librerías necesarios para el
desarrollo, un programador frontend , Conocimientos
de diseño para encontrar la mejor manera de presentar la información

### Perfiles

Diseñador gráfico,Diseñador UX,Maquetador, etc.

### Lenguajes Básicos

HTML, CSS y Javascript. A partir de aquí, se multiplican los frameworks y las herramientas, que se tornan cada vez más complejas.

## BACKEND

Programación que no es visible para el usuario.
Esta parte no se ejecuta en el cliente, sino en un servidor (físico o en la nube).

### Tareas

La lógica del negocio
Pprocesar toda la información,
Datos en el interior de la aplicación con las capas visuales que ha creado el Frontend

### Add Conocimientos

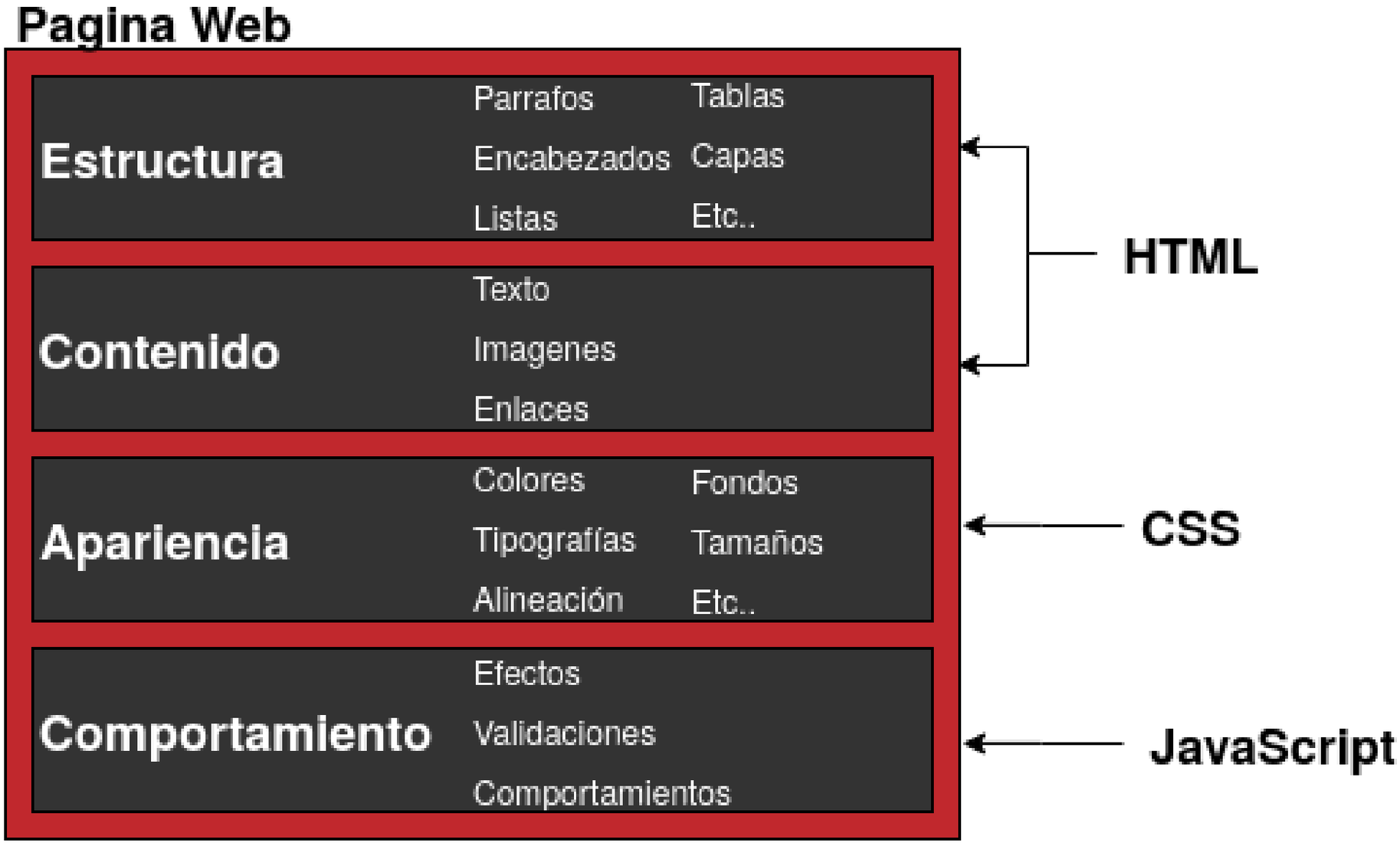
Conocimientos sobre los múltiples lenguajes y frameworks backend,
Bases de datos, protocolos http, diversas herramientas, librerías, interconectividad, APIs, Conceptos abstractos y lógica compleja.

### Perfiles

Ingenieros de Sistemas, Ingenieros de Software, Programadores Certificados

### Lenguajes Básicos

Java, PHP, Python, NodeJS, C#, .NET, Ruby, etc. nuevos lenguajes que van ganando alcance, como Kotlin y Go. etc.


Pagina Web
Estructura
Parrafos
Tablas
Encabezados
Capas
Listas
Etc..
Contenido
Texto
Imagenes
Enlaces
Apariencia
Colores
Fondos
Tipografías
Tamaños
Alineación
Etc..
Comportamiento
Efectos
Validaciones
Comportamientos
HTML
CSS
JavaScript

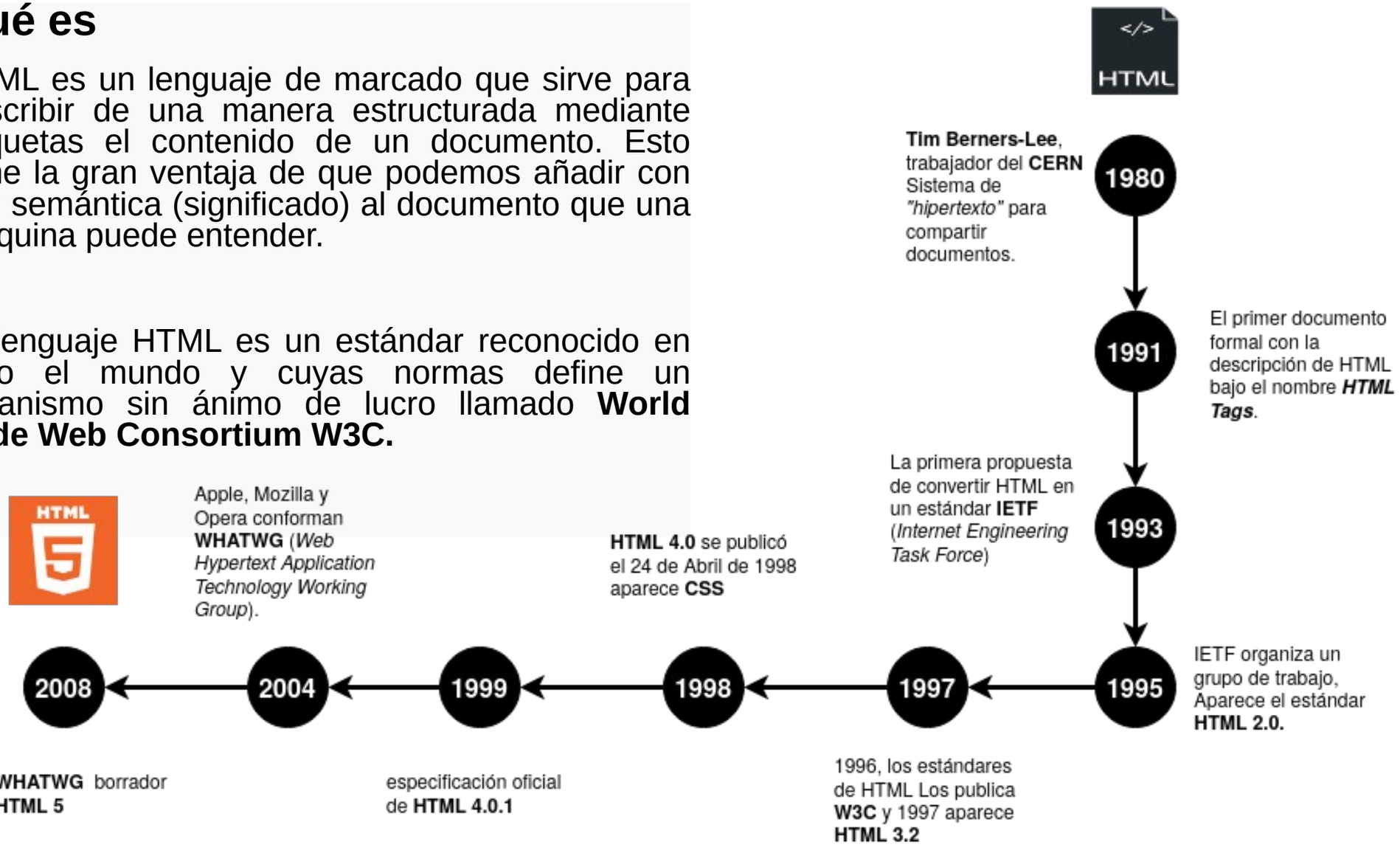
Universidad Industrial de Santander

# HTML

## Qué es

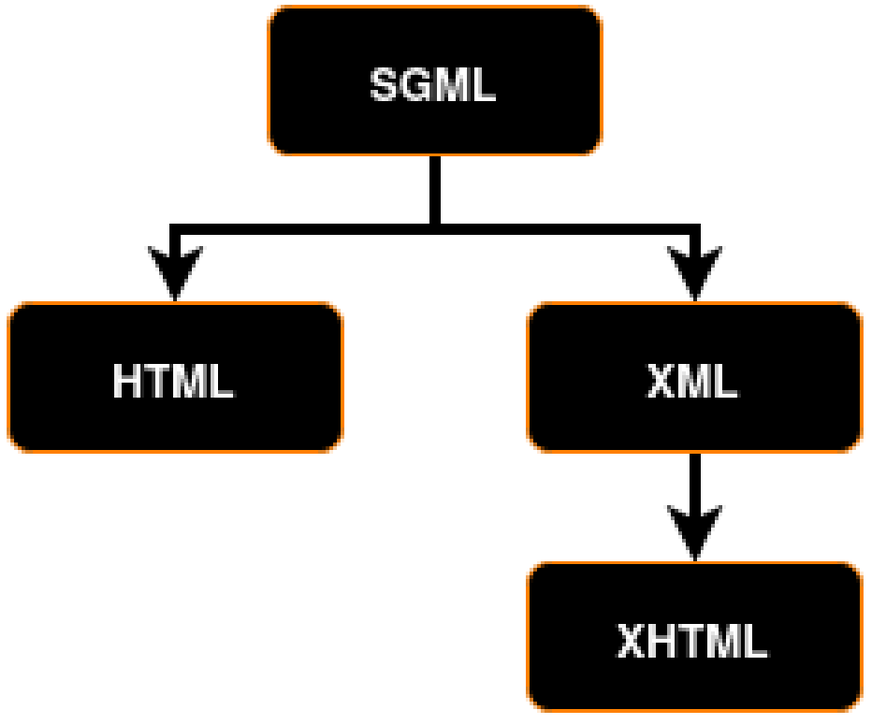
HTML es un lenguaje de marcado que sirve para describir de una manera estructurada mediante etiquetas el contenido de un documento. Esto tiene la gran ventaja de que podemos añadir con ello semántica (significado) al documento que una máquina puede entender.

El lenguaje HTML es un estándar reconocido en todo el mundo y cuyas normas define un organismo sin ánimo de lucro llamado **World Wide Web Consortium W3C.**



## SGML

El lenguaje de marcado generalizado estándar o SGML derivado del anterior GML de IBM, es un estándar para definir lenguajes de marcado generalizados para documentos. ISO 8879 define el Anexo A.1 de marcado generalizado.



## Qué es etiqueta HTML

HTML es un markup language, lo que significa que está escrito con códigos que puede leer una persona sin que sea necesario compilarlo primero. Texto en una página web está «marcado» para dar instrucciones al navegador web sobre cómo mostrar el texto. Estas etiquetas de marcado son las propias etiquetas HTML.

- El carácter “menor que” **<**
- Una palabra o carácter que determina qué etiqueta se está escribiendo
- Cualquier número de atributos HTML que se quiera usar, escritos de la forma nombre=”valor”
- El carácter “mayor que” **>**

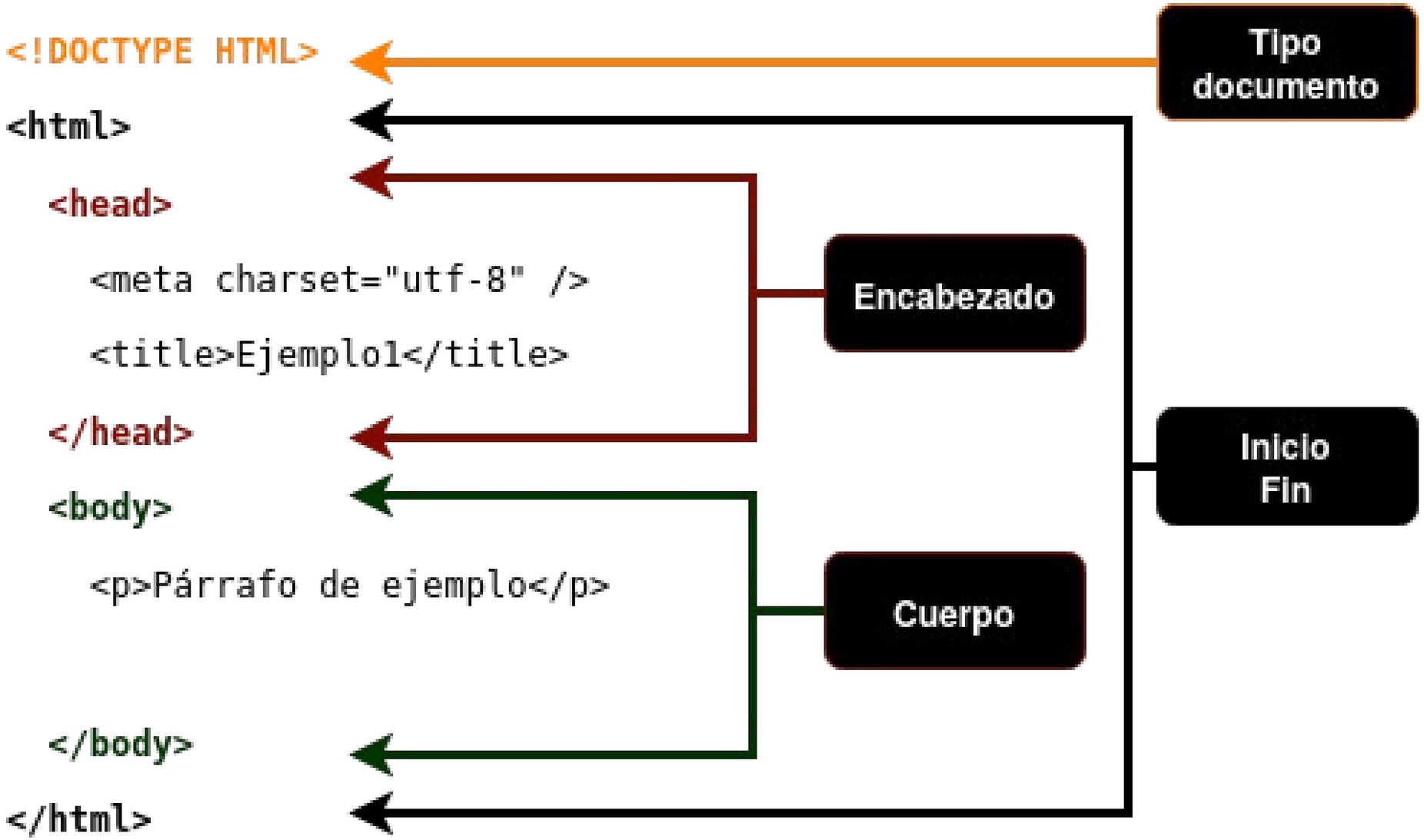
forman una estructura jerárquica, es decir, se pueden anidar entre ellas, excepto la etiqueta especial **<!DOCTYPE HTML>** en la primera línea de un documento HTML.

Siempre hay una etiqueta de apertura y otra de cierre que es la misma etiqueta, pero con una barra delante del nombre de la etiqueta.

**<title>** (etiqueta de apertura) y **</title>** (etiqueta de cierre)

### Que editor

HTML en cualquier editor, incluso editor de texto, pero es recomendado usar editor HTML especializado, que ayude a colocar las etiquetas.


<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8" />
<title>Ejemplo1</title>
</head>
<body>
Párrafo de ejemplo
</body>
</html>
Tipo documento
Encabezado
Cuerpo
Inicio
Fin

# Etiquetas

**<html>**: indica el comienzo y el final de un documento HTML. Ninguna etiqueta o contenido puede colocarse antes o después de la etiqueta **<html>** con una sola excepción. En el interior de la etiqueta **<html>** se definen la cabecera y el cuerpo del documento HTML y todo lo que se coloque fuera de la etiqueta **<html>** se ignora.

**<head>:** delimita la parte de la cabecera del documento. La cabecera contiene información sobre el propio documento, como su título y el idioma. Los contenidos indicados en la cabecera no son visibles para el usuario, con la excepción de **<title>**, que indica el título del documento y que los navegadores lo visualizan la superior izquierda de la ventana del navegador.

**<body>**: delimita el cuerpo del documento HTML. El cuerpo encierra todos los contenidos que se muestran al usuario (párrafos de texto, imágenes, tablas). En general, el **<body>** de un documento contiene cientos de etiquetas HTML, mientras que el **<head>** no contiene más que unas pocas.

## Etiquetas

HTML define **91** etiquetas que los diseñadores pueden utilizar para marcar los diferentes elementos que componen una página:

**a, abbr, acronym, address, applet, area, b, base, basefont, bdo, big, blockquote, body, br, button, caption, center, cite, code, col, colgroup, dd, del, dfn, dir, div, dl, dt, em, fieldset, font, form, frame, frameset, h1, h2, h3, h4, h5, h6, head, hr, html, i, iframe, img, input, ins, isindex, kbd, label, legend, li, link, map, menu, meta, noframes, noscript, object, ol, optgroup, option, p, param, pre, q, s, samp, script, select, small, span, strike, strong, style, sub, sup, table, tbody, td, textarea, tfoot, th, thead, title, tr, tt, u, ul, var.**

De todas las etiquetas disponibles, las siguientes se consideran obsoletas y no se pueden utilizar: **applet, basefont, center, dir, font, isindex, menu, s, strike, u**

cada una de las etiquetas HTML define sus propios atributos, algunos de los atributos son comunes a muchas o casi todas las etiquetas. De esta forma, es habitual explicar por separado los atributos comunes de las etiquetas para no tener que volver a hacerlo cada vez que se explica una nueva etiqueta. Los atributos comunes se dividen en cuatro grupos según su funcionalidad:

Atributos básicos: se pueden utilizar prácticamente en todas las etiquetas HTML.

Atributos para internacionalización: los utilizan las páginas que muestran sus contenidos en varios idiomas.

Atributos de eventos: sólo se utilizan en las páginas web dinámicas creadas con JavaScript.

Atributos de foco: relacionados principalmente con la accesibilidad de los sitios web.

## 1 Atributos básicos

Los siguiente cuatro atributos se pueden aplicar prácticamente a todas las etiquetas HTML:

**id = "texto"** : Establece un identificador único a cada elemento dentro de una página HTML.

**class = "texto":** Establece la clase CSS que se aplica a los estilos del elemento.

**style = "texto"** : Establece de forma directa los estilos CSS de un elemento.

**title = "texto"** : Establece el título a un elemento (muestran cuando el usuario pasa el ratón por encima del elemento) .

id y class son realmente útiles cuando se trabaja con **CSS** y con **Javascript**. sólo pueden contener guiones medios (-), guiones bajos (_), letras y/o números, pero no pueden empezar por números. Además, los navegadores distinguen mayúsculas de minúsculas y no ñ ni acentos.

## 2 Atributos para internacionalización

Estos atributos son útiles para aquellas páginas que muestran sus contenidos en varios idiomas y para aquellas que quieren indicar de forma explítica el idioma de sus contenidos:

**lang = "codigo de idioma" :** Indica el idioma del elemento mediante un código predefinido

**xml:lang = "codigo de idioma"** : Indica el idioma del elemento mediante un código predefinido.

**dir** : Indica la dirección del texto (útil para los idiomas que escriben de derecha a izquierda)

En las páginas XHTML, el atributo xml:lang tiene más prioridad que lang y es obligatorio.

Como la palabra internacionalización es muy larga, se suele sustituir por la abreviatura i18n

# 3 Atributos de eventos

Estos atributos sólo se utilizan en las páginas web que incluyen código JavaScript para realizar acciones dinámicas.

**onblur** : Deseleccionar el elemento **<button>, <input>, <label>, <select>, <textarea>, <body>**.

**onchange :** Deseleccionar un elemento que se ha modificado **<input>, <select>, <textarea>**.

**onclick** : Pinchar y soltar el ratón, **todos**.

**ondblclick** : Pinchar dos veces seguidas con el ratón .**todos**.

**onfocus** : Seleccionar un elemento, **<button>, <input>, <label>, <select>, <textarea>, <body>**.

**onkeydown** : Pulsar una tecla (sin soltar), **elementos del <body>.**

**onkeypress :** Pulsar una tecla **elementos del <body>.**

**onkeyup** : Soltar una tecla pulsada **elementos del <body>**.

**onload** : La página se ha cargado completamente, **<body>.**

**onmousedown** : Pulsar (sin soltar) un botón del ratón **todos**.

**onmousemove** : Mover el ratón **todos**.

**onmouseout** : El ratón "sale" del elemento **todos**.

**onmouseover** : El ratón "entra" en el elemento **Todos**.

**onmouseup** : Soltar el botón que estaba pulsado en el ratón **Todos**.

## 3 Atributos de eventos

**onreset** : Inicializar el formulario (borrar todos sus datos) **<form>**.

**onresize :** Se ha modificado el tamaño de la ventana del navegador **<body>**.

**onselect** : Seleccionar un texto, **<input>, <textarea>**.

**onsubmit** : Enviar el formulario .**<form>**.

**onunload** : Se abandona la página (por ejemplo al cerrar el navegador), **<body>**.

## 4 Atributos de foco

Cuando el usuario selecciona un elemento en una aplicación.

**accesskey = "letra" :** Establece una tecla de acceso rápido a un elemento HTML**.**

**tabindex = "numero"** : Establece la posición del elemento en el orden de tabulación de la página. Su valor debe estar comprendido entre 0 y 32.767.

**onfocus, onblur** : Controlan los eventos JavaScript que se ejecutan cuando el elemento obtiene o pierde el foco**.**

El atributo tabindex permite alterar el orden en el que se seleccionan los elementos.

accesskey permite establecer una tecla para acceder de forma rápida a cualquier elemento, Explorer ALT + la tecla definida; en Firefox Alt + Shift + la tecla definida; Opera Shift + Esc + la tecla definida; Safari Ctrl + la tecla definida.

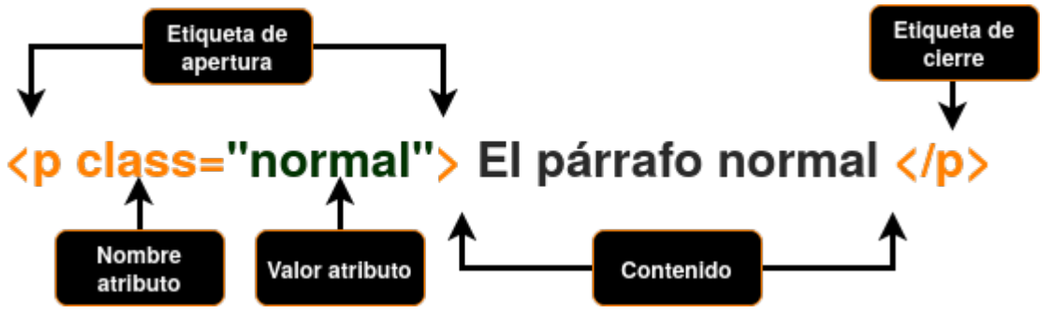
# Elementos HTML

HTML define el término elemento para referirse a las partes que componen los documentos HTML, un elemento HTML es mucho más que una etiqueta, ya que está formado por:

- Una etiqueta de apertura.
- Cero o más atributos.
- Texto encerrado por la etiqueta.
- Una etiqueta de cierre.

todos los elementos en dos grupos: elementos en línea (inline) y elementos de bloque (block ).

Los elementos de bloque siempre empiezan en una nueva línea y ocupan todo el espacio disponible hasta el final de la línea, aunque sus contenidos no lleguen hasta el final de la línea. Por su parte, los elementos en línea sólo ocupan el espacio necesario para mostrar sus contenidos



## 1 elementos en línea

Los elementos en línea definidos por HTML son: **a, abbr, acronym, b, basefont, bdo, big, br, cite, code, dfn, em, font, i, img, input, kbd, label, q, s, samp, select, small, span, strike, strong, sub, sup, textarea, tt, u, var.**

## 2 elementos en bloque

Los elementos de bloque definidos por HTML son: **address, blockquote, center, dir, div, dl, fieldset, form, h1, h2, h3, h4, h5, h6, hr, isindex, menu, noframes, nos-cript, ol, p, pre, table, ul.**

Los siguientes elementos también se considera que son de bloque: **dd, dt, frame-set, li, tbody, td, tfoot, th, thead, tr.**

Los siguientes elementos pueden ser en línea y de bloque según las circunstancias: **button, del, iframe, ins, map, object, script.**

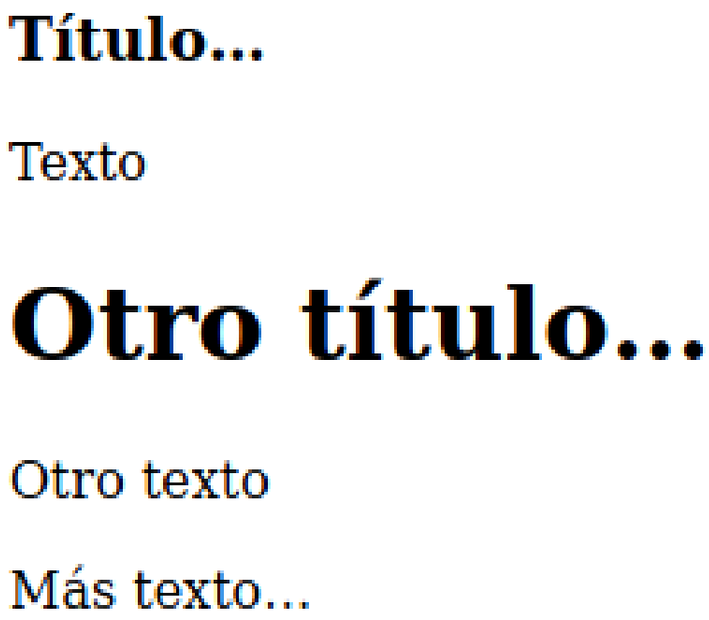
```
<!DOCTYPE html>
<html>

<head>
  <title>Ejemplo de elementos en línea y elementos de bloque</title>
</head>

<body>
<p>Los párrafos son elementos de bloque.</p>
<a href="http://www.google.com">Los enlaces son elementos en línea</a>
<p>Dentro de un párrafo, <a href="http://www.google.com">los enlaces</a>
siguen siendo elementos en línea.</p>
</body>

</html>
```

```
<!DOCTYPE HTML>
<html lang="es">
<head>
  <title>Título de la página…</title>
  <meta charset="UTF-8">
  <meta name="description" content="Descripción de la página…">
</head>
<body>
  <div class=”barra-lateral”>
    <h3>Título…</h3>
    <p>Texto</p>
  </div>
  <div class=”contenido”>
    <h1>Otro título…</h1>
    <p>Otro texto</p>
    <p>Más texto… </p>
  </div>
</body>
</html>
```



## Texto

HTML habituales está formado por texto, llegando a ser más del 90% del código de la página

**Párrafos** Delimita el contenido de un párrafo de texto.

**Secciones** Define los títulos de las secciones de mayor importancia de la página.**h1, h2, h3, h4, h5 y h6.**

**Marcado** tenemos dos **** Realza la importancia del texto que encierra y **** Realza con la máxima importancia el texto que encierra el texto que ha sido eliminado y el texto que ha sido añadido a un determinado texto original. Las etiquetas utilizadas son **<ins>** y **<del>** la etiqueta **<blockquote>** para incluir citas textuales en las páginas web

## Enlaces

Los enlaces se utilizan para establecer relaciones entre dos recursos. páginas web, imágenes, documentos y archivos. Dentro de los enlaces podemos hablar de dos clases absoluto o relativo.

**<a>** Para enlazar todo tipo de recursos name = "texto" - Permite nombrar, href = "url" - Indica la URL, hreflang = "codigo_idioma" – Idioma, type = "tipo_de_contenido" - Permite "avisar" al navegador sobre el tipo de contenido,rel = "tipo_de_relacion" recurso enlazado, rev = "tipo_de_relacion" documento,charset = "tipo_de_charset" -codificación del recurso enlazado.

**<script>** Se emplea para enlazar o definir un bloque de código (normalmente JavaScript)

<link> Se emplea para enlazar y establecer relaciones entre el documento y otros recursos

# Listas

listas para agrupar los elementos: listas no ordenadas, listas ordenadas y listas de definición

**Listas no ordenadas** La etiqueta <ul> encierra todos los elementos de la lista y la etiqueta <li> cada uno de sus elementos.

```
<ul>
  <li>Uno </li>
  <li>dos </li>
</ul>
```

**Listas ordenadas** se define mediante la etiqueta <ol>. Los elementos de la lista se definen mediante la etiqueta <li>.

```
<ol>
  <li>Enchufar correctamente</li>
  <li>Comprobar conexiones</li>
  <li>Encender el aparato</li>
</ol>
```

**Listas de definición** La etiqueta <dl> crea la lista de definición y las etiquetas <dt> y <dd> definen respectivamente el término y la descripción de cada elemento de la lista.

```
<dl>
  <dt>Titulo </dt>
  <dd>subtitulo</dd>

  <dt>Titulo</dt>
  <dd>subtitulos</dd>

  <dt>Titulo</dt>
  <dt>Titulo</dt>
  <dt>Titulo</dt>
  <dt>Titulo</dt>
  <dt>Titulo</dt>
  <dd>subtitulos</dd>
</dl>
```

## Imágenes y Objetos

**imagen <img>** Se emplea para incluir imágenes en los documentos.

**Mapas de imagen <map>** Se emplea para definir mapas de imagen **<area>** Se emplea para definir las distintas áreas que forman un mapa de imagen.

**Objetos <object>** Se emplea para embeber objetos en los documentos como applets de Java y vídeos en formato QuickTime o Flash , **<param>** Se emplea para indicar el valor de los parámetros del objeto, **<embed>** Se emplea para embeber objetos en los documentos .

## Tablas

Las tablas en HTML utilizan los mismos conceptos de filas, columnas, cabeceras y títulos .

**Tablas básicas** se definen con tres etiquetas: <table> para crear la tabla, <tr> para crear cada fila y <td> para crear cada columna.

```
<table>
<tr>
  <td><strong>Col1</strong></td>
  <td><strong>Col2</strong></td>
  <td><strong>Col3</strong></td>
</tr>
<tr>
  <td>N1</td>
  <td>N2</td>
  <td>N3</td>
</tr>
<tr>
  <td>N4</td>
  <td>N5</td>
  <td>N6</td>
</tr>
</table>
```

# Formularios

El estándar HTML/XHTML permite crear formularios para que los usuarios interactúen con las aplicaciones web.

**Formularios básicos** Los formularios más sencillos se pueden crear utilizando solamente dos etiquetas: **<form>** y **<input>**.

**<form>** Atributos:
**action = "url"** - Indica la URL que se encarga de procesar los datos del formulario
**method = "POST o GET"** - Método HTTP empleado al enviar el formulario
**enctype** = "application/x-www-form-urlencoded o multipart/form-data" - Tipo de codificación empleada al enviar el formulario al servidor
**accept = "tipo_de_contenido"** - Lista separada por comas de todos los tipos de archivos aceptados
**Otros**: accept-charset, onsubmit, onreset

```
<!DOCTYPE html>
<html>
<head><title>formulario sencillo</title></head>
<body>
<h3>Formulario</h3>
<form action="https://www.muylinux.com/" method="post">
  Nombre:
  <input type="text" name="nombre" value="" />
  <br/>
  <input type="submit" value="Enviar" />
</form>
</body>
</html>
```

# Formularios

**Elementos de formulario** Los elementos de formulario como botones y cuadros de texto también se denominan "campos de formulario" y "controles de formulario". La mayoría de controles se crean con la etiqueta <input>.

<input> Atributos

type = "text | password | checkbox | radio | submit | reset | file | hidden | image | button" - Indica el tipo de control que se incluye en el formulario

name = "texto" - Asigna un nombre al control (es imprescindible para que el servidor pueda procesar el formulario)

value = "texto" - Valor inicial del control

size = "unidad_de_medida" - Tamaño inicial del control (para los campos de texto y de password se refiere al número de caracteres, en el resto de controles se refiere a su tamaño en píxel)

maxlength = "numero" - Máximo número de caracteres para los controles de texto y de password

checked = "checked" - Para los controles checkbox y radiobutton permite indicar qué opción aparece preseleccionada

disabled = "disabled" - El control aparece deshabilitado y su valor no se envía al servidor junto con el resto de datos

readonly = "readonly" - El contenido del control no se puede modificar

src = "url" - Para el control que permite crear botones con imágenes, indica la URL de la imagen que se emplea como botón de formulario

alt = "texto" - Descripción del control


## Rellena tu CV

Nombre

Apellidos

Contraseña

Identificación

TI

Sexo

- Hombre
- Mujer

Incluir mi foto Examinar... No se ha seleccionado ningún archivo.

Suscribirme para hacer....

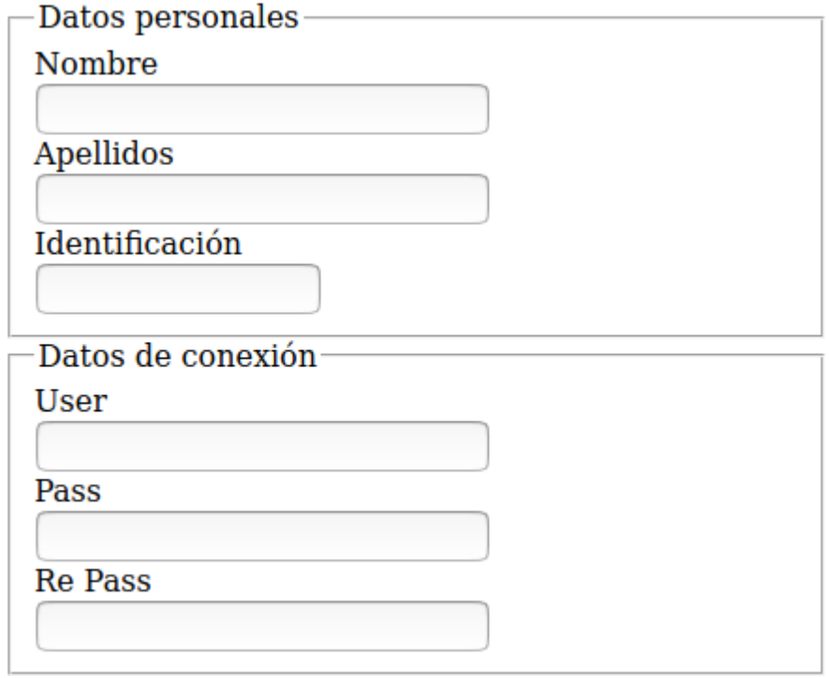
Guardar cambios Borrar los datos introducidos

```
<!DOCTYPE html>
<html xmlns="http://www.uis.edu.co" lang="es" xml:lang="es">
 head>
 meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
 title>Rellena tu CV</title>
 /head>
 body>
 h3>Rellena tu CV</h3>
 form action="/cv/dataCV" method="post" enctype="multipart/form-data">
 ombre <br/>
 input type="text" name="nombre" value="" size="20" maxlength="30" />
 br/>
 pellidos <br/>
 input type="text" name="apellidos" value="" size="50" maxlength="80" />
 br/>
 ontraseña <br/>
 input type="password" name="contrasena" value="" maxlength="10" />
 br/>
 dentificación <br/>
 select name="select">
 <option value="CC">CC</option>
 <option value="TI" selected>TI</option>
 <option value="CE">CE</option>
 /select>
 input type="text" name="idsuj" value="" size="10" maxlength="9" />
 br/>
 exo <br/>
 input  type="radio"  name="sexo"  value="hombre"  checked="checked"  /> Hombre
 br/>
 input type="radio" name="sexo" value="mujer" /> Mujer
 br/><br/>
 ncluir mi foto <input type="file" name="foto" />
 br/><br/>
 input  name="suscribir"  type="checkbox"  value="suscribir"  checked="checked"/>
 uscribirme para hacer....
 br/><br/>
 input type="submit" name="enviar" value="Guardar cambios" />
 input type="reset" name="limpiar" value="Borrar los datos introducidos" />
 /form>
 /body>
 /html>
```

# Formularios

**Formularios avanzados** HTML define algunos elementos adicionales para mejorar la estructura de los formularios creados. La etiqueta <fieldset> agrupa campos del formulario y la etiqueta <legend> asigna un nombre a cada grupo.

```
<!DOCTYPE html>
<html>
<head>
    <title></title>
</head>
<body>
<form action="maneja_formulario.php" method="post">
  <fieldset>
    <legend>Datos personales</legend>
    Nombre <br/>
    <input type="text" name="nombre" value="" />
    <br/>
    Apellidos <br/>
    <input type="text" name="apellidos" value="" />
    <br/>
    Identificación <br/>
    <input type="text" name="dni" value="" size="10" maxlength="9" />
  </fieldset>
  <fieldset>
    <legend>Datos de conexión</legend>
    User<br/>
    <input type="text" name="nombre" value="" maxlength="10" />
    <br/>
    Pass<br/>
    <input type="password" name="password" value="" maxlength="10" />
    <br/>
    Re Pass<br/>
    <input type="password" name="password2" value="" maxlength="10" />
  </fieldset>
</form>
</body>
</html>
```



# Formularios

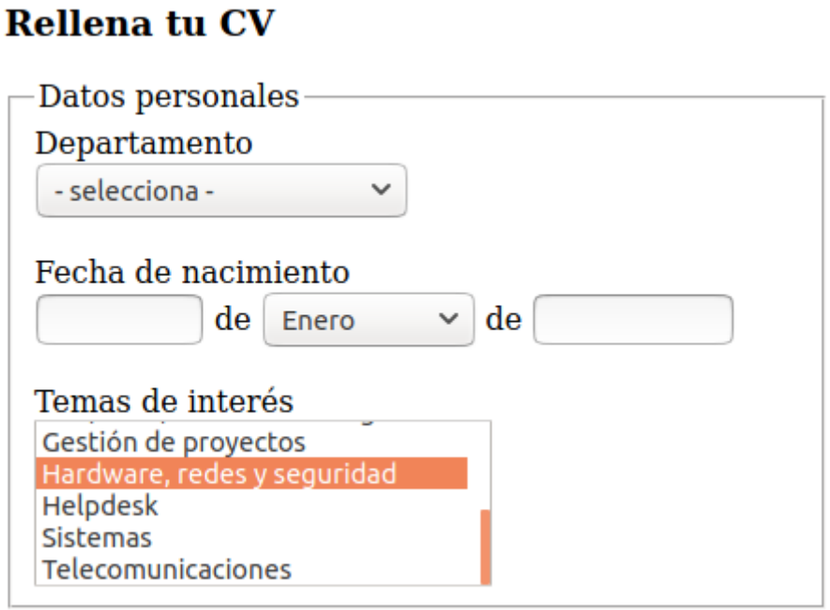
## Formularios Otros elementos

**<textarea>** Se emplea para incluir un área de texto en un formulario, **Atributos** rows = "numero" filas , cols = "numero" número de caracteres, Otros: name, disabled, readonly, onselect, onchange, onfocus, onblur

**<select>** Se emplea para incluir una lista desplegable en un formulario, size = "numero" Número de filas, multiple = "multiple" seleccionar más de un elemento, Otros: name, disabled, onchange, onfocus, onblur.

**<option>** Se emplea para definir cada elemento de una lista desplegable, selected = "selected" elemento seleccionado, value = "texto" valor de elección.

**<optgroup>** Se emplea para definir una agrupación lógica de opciones de una lista desplegable label = "texto", Otros: disabled, selected.



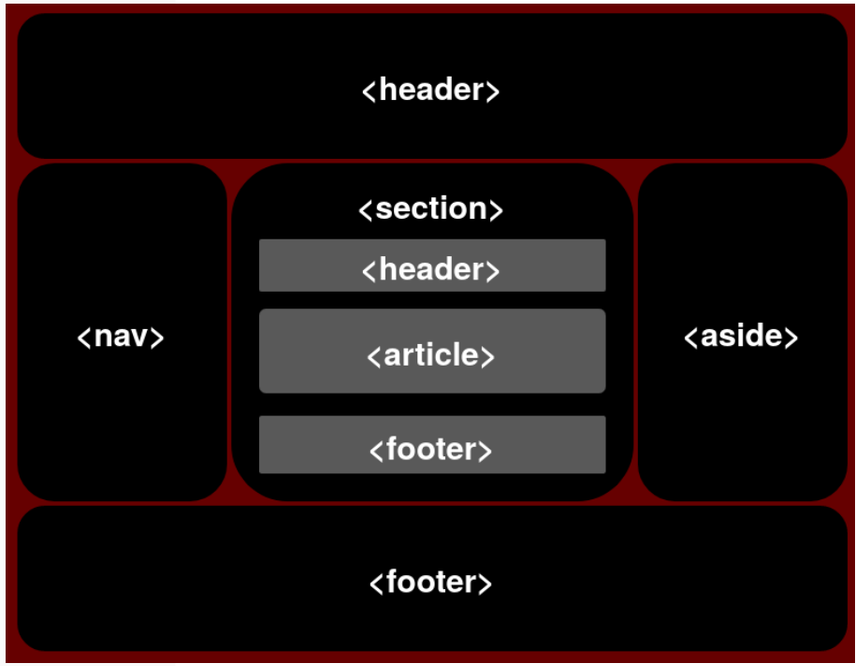
## Estructura y layout

Para agrupar los elementos que forman cada zona o división de la página se utiliza la etiqueta :

```
<div id="wrapper">
  <div id="header">
    ...
  </div>

  <div id="content">
    <div id="menu">
      ..
    </div>
    ...
  </div>

  <div id="footer">
    ...
  </div>
</div>
```



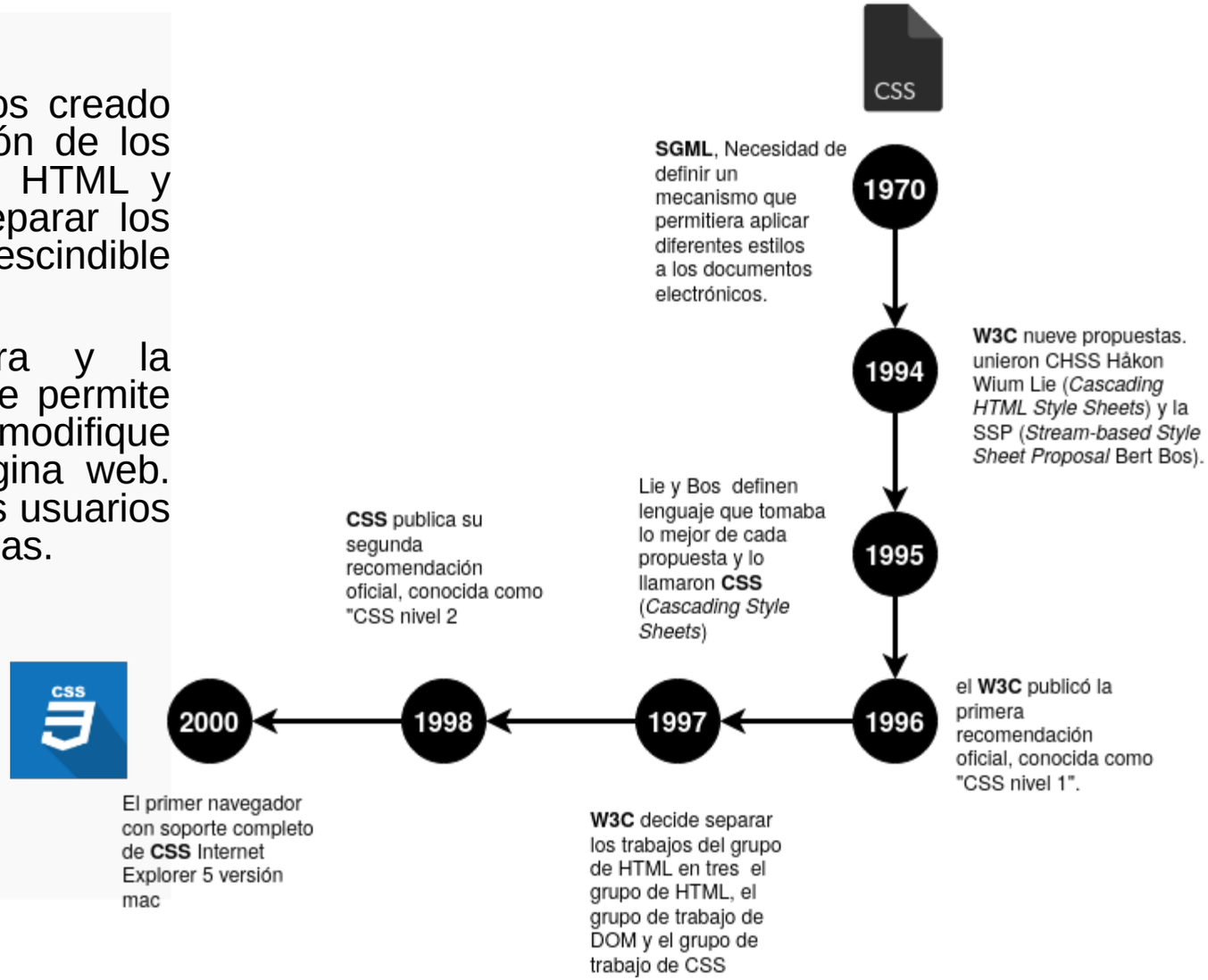
Universidad Industrial de Santander

# CSS

## Que es

CSS es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. CSS es la mejor forma de separar los contenidos y su presentación y es imprescindible para crear páginas web complejas.

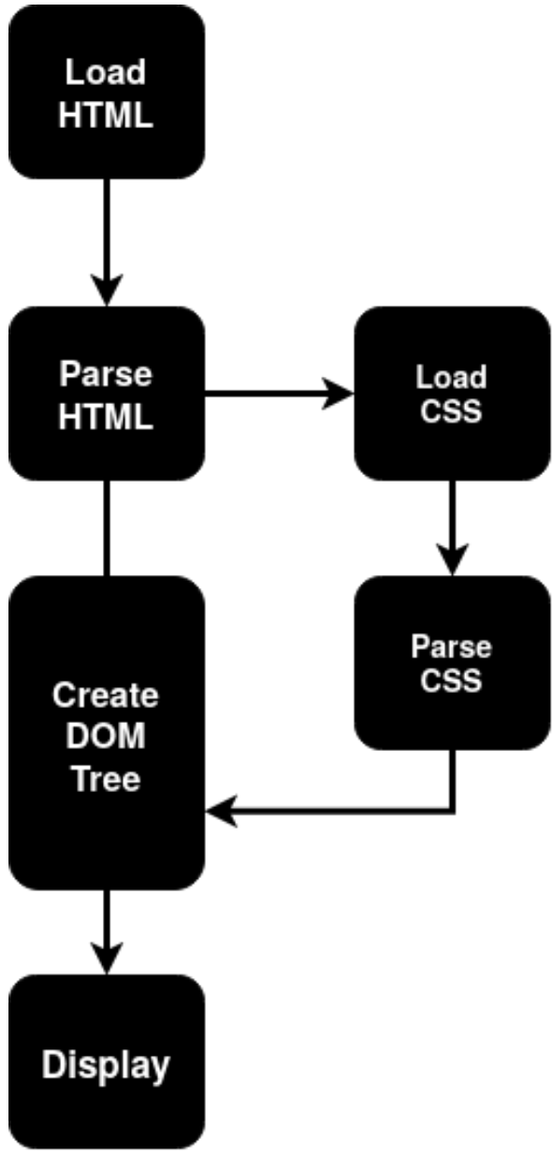
Esta separación entre la estructura y la presentación es muy importante, ya que permite que sólo cambiando los CSS se modifique completamente el aspecto de una página web. Esto posibilita, entre otras cosas, que los usuarios puedan usar hojas de estilo personalizadas.



## Como funciona

El navegador carga el HTML.

- Convierte el HTML en un DOM (Modelo de objetos del documento). El DOM representa el documento en la memoria del ordenador.
- El navegador va a buscar la mayor parte de los recursos vinculados al documento HTML, como las imágenes y los videos. ¡y también el CSS vinculado! JavaScript aparece más adelante.
- El navegador analiza el CSS y ordena en diferentes «cubos» las diferentes reglas según el tipo de selector. Para cada tipo de selector se calcula qué reglas deben aplicarse y a qué nodos en el DOM se les aplica el estilo según corresponda.
- El árbol de renderización presenta la estructura en que los nodos deben aparecer después de aplicarle las reglas.
- En la pantalla se muestra el aspecto visual de la página.



# Incluir el CSS en HTML

**En el mismo Documento** Se emplea la etiqueta <style> de HTML y solamente se pueden incluir en la cabecera del documento (sólo dentro de la sección <head>).

```
<!DOCTYPE html>
<html xmlns="http://www.uis.edu.co">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Ejemplo de estilos CSS en el propio documento</title>
<style type="text/css">
  p { color: red; font-family: Verdana; }
</style>
</head>
<body>
<p>Un párrafo de texto...</p>
</body>
</html>
```

**CSS en los elementos HTML** TML es el peor y el menos utilizado, ya que tiene los mismos problemas que la utilización de las etiquetas <font>.

## estilos.css

```
p { color: black; font-family: Verdana; }
```

```
<!DOCTYPE html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Ejemplo de estilos CSS en el propio documento</title>
</head>

<body>
<p style="color: black; font-family: Verdana;">Un párrafo de texto.</p>
</body>
</html>
```

**Archivo externo** CSS se incluyen en un archivo de tipo CSS que las páginas HTML enlazan mediante la etiqueta <link>. Un archivo de tipo CSS no es más que un archivo simple de texto cuya extensión es .css.

**estilos.css**

```
p { color: black; font-family: Verdana; }
```

```
<!DOCTYPE html>
<html xmlns="http://www.uis.edu.co">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Ejemplo de estilos CSS en un archivo externo</title>
<link rel="stylesheet" type="text/css" href="/css/estilos.css" media="screen" />
</head>

<body>
<p>Un párrafo de texto.</p>
</body>
</html>
```

**Archivo externo** La forma alternativa los estilos definidos en archivos CSS externos se utiliza una regla especial de tipo @import. Las reglas de tipo @import siempre preceden a cualquier otra regla CSS (con la única excepción de la regla @charset).

```
<!DOCTYPE html>
<html xmlns="http://www.uis.edu.co">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Ejemplo de estilos CSS en un archivo externo</title>
<style type="text/css" media="screen">
  @import '/css/estilos.css';
</style>
</head>

<body>
<p>Un párrafo de texto.</p>
</body>
</html>
```

## Definición

permiten describir cada una de las partes que componen los estilos CSS.

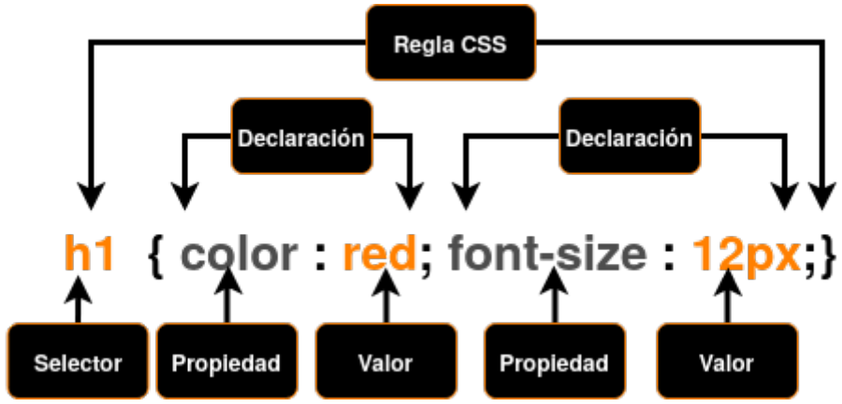
Regla: cada uno de los estilos que componen una hoja de estilos CSS. Cada regla está compuesta de una parte de "selectores", un símbolo de "llave de apertura" ({), otra parte denominada "declaración" y por último, un símbolo de "llave de cierre" (}).

**Selector**: indica el elemento o elementos HTML a los que se aplica la regla CSS.

**Declaración**: especifica los estilos que se aplican a los elementos. Está compuesta por una o más propiedades CSS.

**Propiedad**: característica que se modifica en el elemento seleccionado, como por ejemplo su tamaño de letra, su color de fondo, etc.

**Valor**: establece el nuevo valor de la característica modificada en el elemento.



## Medios

CSS es que permiten definir diferentes estilos para diferentes medios o dispositivos.

**all**: Todos los medios definidos.

**Braille**: Dispositivos táctiles que emplean el sistema braille.

**Embosed**: Impresoras braille

**Handheld**: Dispositivos de mano: móviles, PDA, etc.

**Print**: Impresoras y navegadores en el modo "Vista Previa para Imprimir"

**Projection**: Proyectores y dispositivos para presentaciones

**Screen**: Pantallas de ordenador

**Speech**: Sintetizadores para navegadores de voz utilizados por personas discapacitadas

**Tty**: Dispositivos textuales limitados como teletipos y terminales de texto

**Tv**: Televisores y dispositivos con resolución baja

**especificaciones** especificar el medio en el que se aplican los estilos.

**@media**

```
@media print {
  body { font-size: 10pt }
}
```

**@import** para enlazar archivos CSS externos.

```
@import url("estilos_basicos.css") screen;
```

**<link>** para enlazar los archivos CSS externos.

```
<link rel="stylesheet" type="text/css" media="screen" href="basico.css" />
```

**mezclando varios métodos**

```
<link rel="stylesheet" type="text/css" media="screen" href="basico.css" />
@import url("estilos_seccion.css") screen;
@media print {
  /* Estilos específicos para impresora */
}
```

# Selectores

## Básicos

El selector **CSS** es el nexo de unión entre la hoja de estilos y los documentos a los que se aplique dicha hoja.

**Selector universal**: Se utiliza para seleccionar todos los elementos de la página se indica mediante un asterisco (*) .

```
* {
  margin: 0;
  padding: 0;
}
```

**tipo o etiqueta**: Selecciona todos los elementos de la página cuya etiqueta HTML coincide con el valor del selector.

```
h1, h2, h3 {
  color: #8A8E27;
  font-weight: normal;
  font-family: Arial, Helvetica, sans-serif;
}
```

**descendente** Selecciona los elementos que se encuentran dentro de otros elementos. El selector selecciona todos los elementos  que se encuentren dentro de un elemento.

```
span { background-color: white; }
div span { background-color: DodgerBlue;
}
<div>
  <span>Span 1.
    <span>Span 2.</span>
  </span>
</div>
<span>Span 3.</span>
```

**de clase:** se utilizar el atributo class de HTML sobre ese elemento para indicar directamente la regla CSS que se le debe aplicar

```
.destacado { color: red; }

   <p class="destacado">Lorem ipsum dolor sit amet...</p>
```

# Selectores

**de ID**: El selector de ID permite seleccionar un elemento de la página a través del valor de su atributo id. La sintaxis de los selectores de ID es muy parecida a la de los selectores de clase, salvo que se utiliza el símbolo de la almohadilla (#) en vez del punto (.).

**#destacado { color: red; }**

** Sin color **
** Con color **
** Tercer párrafo }**

**Combinación de selectores básicos** :CSS permite la combinación de uno o más tipos de selectores para restringir el alcance de las reglas CSS.

selecciona aquellos elementos con un class="especial" que se encuentren dentro de cualquier elemento con un class="aviso".

**.aviso .especial { ... }**

Elementos de tipo  con un atributo class="especial" que estén dentro de cualquier elemento de tipo  que tenga un atributo class="aviso".

**div.aviso span.especial { ... }**

Enlace con un atributo id igual a inicio que se encuentra dentro de un elemento de tipo <li> con un atributo class igual a destacado, que forma parte de una lista <ul> con un atributo id igual a menuPrincipal.

**ul#menuPrincipal li.destacado a#inicio { .. }**

# Selectores

**Selector de atributos**: selectores avanzados lo forman los selectores de atributos, que permiten seleccionar elementos HTML en función de sus atributos y/o valores de esos atributos.

Los cuatro tipos de selectores de atributos:

**[nombre_atributo]** selecciona los elementos que tienen establecido el atributo llamado nombre_atributo, independientemente de su valor.

**[nombre_atributo=valor]** selecciona los elementos que tienen establecido un atributo llamado nombre_atributo con un valor igual a valor.

**[nombre_atributo~=valor]** selecciona los elementos que tienen establecido un atributo llamado nombre_atributo y al menos uno de los valores del atributo es valor.

**[nombre_atributo|=valor]** selecciona los elementos que tienen establecido un atributo llamado nombre_atributo y cuyo valor es una serie de palabras separadas con guiones, pero que comienza con valor. Este tipo de selector sólo es útil para los atributos de tipo lang que indican el idioma del contenido del elemento.

## Agrupación de reglas

CSS complejos con decenas, es habitual que los estilos que se aplican a un mismo selector se definan en diferentes reglas:

```
h1 { color: red; }
h1 { font-size: 2em; }
h1 { font-family: Verdana; }
```

se pueden agrupar las declaraciones de las reglas y quitar los espacios, para hacer las hojas de estilos más eficientes.

```
h1{color:red;font-size:2em;font-family:Verdana;}
```

## Herencia

Cuando se establece el valor de una propiedad CSS en un elemento, sus elementos descendientes heredan de forma automática el valor de esa propiedad.

establecer el color de la letra en el elemento <body> de la página implica cambiar el color de letra de todos los elementos de la página.

```
body { color: blue; }
```

## Colisiones de estilos

CSS se apliquen a un mismo elemento HTML. El problema de estas reglas múltiples es que se pueden dar colisiones.

El método seguido por CSS para resolver las colisiones

- Determinar todas las declaraciones que se aplican al elemento para el medio CSS seleccionado.
- Ordenar las declaraciones según su origen (CSS de navegador, de usuario o de diseñador) y su prioridad (palabra clave ! important).
- Ordenar las declaraciones según lo específico que sea el selector. Cuanto más genérico es un selector, menos importancia tienen sus declaraciones.
- Si después de aplicar las normas anteriores existen dos o más reglas con la misma prioridad, se aplica la que se indicó en último lugar.

conceptos de tipo de hoja de estilo y la prioridad, el mecanismo simplificado que se puede aplicar es el siguiente:

- Cuanto más específico sea un selector, más importancia tiene su regla asociada.
- A igual especificidad, se considera la última regla indicada.

# Unidades de medida

## Unidades absolutas

**in**, pulgadas.

**cm**, centímetros.

**mm**, milímetros.

**pt**, puntos. Un punto equivale a 1 pulgada/72, es decir, unos 0.35 milímetros.

**pc**, picas. Una pica equivale a 12 puntos, es decir, unos 4.23 milímetros.

## Unidades relativas

**em**, (no confundir con la etiqueta  de HTML) relativa respecto del tamaño de letra del elemento.

**ex**, relativa respecto de la altura de la letra x ("equis minúscula") del tipo y tamaño de letra del elemento.

**px**, (píxel) relativa respecto de la resolución de la pantalla del dispositivo en el que se visualiza la página HTML.

## Porcentajes

El porcentaje también es una unidad de medida relativa, aunque por su importancia CSS la trata de forma separada a em, ex y px. Un porcentaje está formado por un valor numérico seguido del símbolo %

## Ejemplos

```
body {
  font-size: 12px;
  text-indent: 3em;
}
h1 { font-size: 15px }.
```

El elemento <body> define un valor para esta propiedad, pero el elemento <h1> no lo hace, por lo que heredará el valor de su elemento padre. Sin embargo, el valor heredado no es 3em, sino 36px.

# Colores

## Palabras clave

**aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, orange, purple, red, silver, teal, white, yellow.**

## RGB decimal

RGB un color se define indicando sus tres componentes R (rojo), G (verde) y B (azul).

**p { color: rgb(71, 98, 176); }**

## RGB porcentual

La diferencia es que en este caso el valor de las componentes RGB puede tomar valores entre 0% y 100%.

**p { color: rgb(27%, 38%, 69%); }**

## RGB hexadecimal

los valores hexadecimales de las componentes RGB en orden y se les añade el prefijo #.

**p { color: #4762B0; }**

## Colores del sistema

Los colores del sistema son similares a los colores indicados mediante su nombre, pero en este caso hacen referencia al color que muestran algunos elementos del sistema operativo del usuario.

## Colores del sistema

216 colores que formaron la paleta de colores "web safe". Esta paleta de colores son utilizados por los diseñadores con la seguridad de que se verán correctamente en cualquier navegador de cualquier sistema operativo de cualquier usuario.

## Modelo de cajas

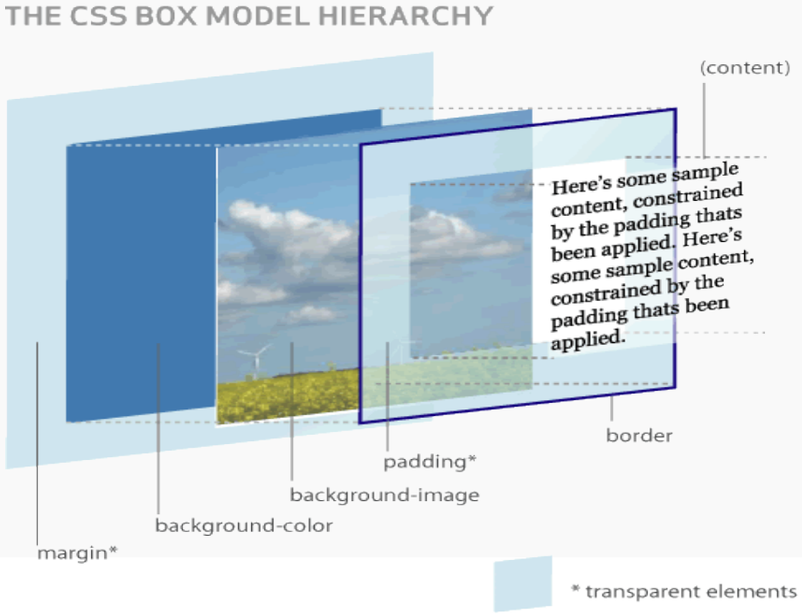
Cada vez que se inserta una etiqueta HTML, se crea una nueva caja rectangular que encierra los contenidos de ese elemento.

```
<p>Este es un párrafo y esto esta
<strong>resaltado</strong> y aquí sin
resaltar</p>
<h4>esto es un titulo</h4>
<p>otro párrafo</p>
```

Este es un párrafo y esto esta **resaltado** y aquí sin resaltar

esto es un titulo

otro párrafo




Las partes que componen cada caja y su orden de visualización desde el punto de vista del usuario :

- Contenido (content): se trata del contenido HTML del elemento (las palabras de un párrafo, una imagen, el texto de una lista de elementos, etc.)
- Relleno (padding): espacio libre opcional existente entre el contenido y el borde.
- Borde (border): línea que encierra completamente el contenido y su relleno.
- Imagen de fondo (background image): imagen que se muestra por detrás del contenido y el espacio de relleno.
- Color de fondo (background color): color que se muestra por detrás del contenido y el espacio de relleno.
- Margen (margin): separación opcional existente entre la caja y el resto de cajas adyacentes.

# Ancho y altura

## Ancho

**Width :**Establece la anchura de un elemento, Se aplica a Todos los elementos, salvo los elementos en línea que no sean imágenes, las filas de tabla y los grupos de filas de tabla el valor inicial es **auto**.

**Valores unidad de medida | porcentaje | auto | inherit**

## Altura

**Height :**Establece la altura de un elemento,Se aplica a Todos los elementos, salvo los elementos en línea que no sean imágenes, las columnas de tabla y los grupos de columnas de tabla.el valor inicial es **auto**.

**Valores unidad de medida | porcentaje | auto | inherit**

# Margen y relleno

## Margen

**margin-top, margin-right, margin-bottom, margin-left** Establece cada uno de los márgenes horizontales y verticales de un elemento. Se aplica a Todos los elementos, salvo margin-top y margin-bottom que sólo se aplican a los elementos de bloque y a las imágenes. El Valor inicial es 0

**Valores unidad de medida | porcentaje | auto | inherit**

**margin** permite definir de forma simultanea los cuatro márgenes se denomina margin. Se aplica a Todos los elementos salvo algunos casos especiales de elementos mostrados como tablas

**Valores ( unidad de medida | porcentaje | auto ) {1, 4} | inherit**

**Un valor**, todos los márgenes tienen ese valor. **Dos valores** 1ro superior y inferior, 2do izquierdo y derecho,**Tres valore**s 1ro superior, tres inferior, 2do izquierdo y derecho, **Cuatro valores** margen superior, margen derecho, margen inferior y margen izquierdo.

# Ancho y altura

## Relleno

**padding-top, padding-right, padding-bottom, padding-left :**Establece cada uno de los rellenos horizontales y vertiTodos los elementos excepto algunos elementos de tablas como grupos de cabeceras y grupos de pies de tabla el valor inicial es **auto**.

**Valores unidad de medida | porcentaje | auto | inherit**

**padding :**Establece de forma directa todos los rellenos de los elementos,Todos los elementos excepto algunos elementos de tablas como grupos de cabeceras y grupos de pies de tabla .el valor inicial es **-**.

**Valores ( unidad de medida | porcentaje ) {1, 4} | inherit**

**Un valor**, todos los márgenes tienen ese valor. **Dos valores** 1ro superior y inferior, 2do izquierdo y derecho,**Tres valore**s 1ro superior, tres inferior, 2do izquierdo y derecho, **Cuatro valores** margen superior, margen derecho, margen inferior y margen izquierdo.

# Bordes

modificar el aspecto de cada uno de los cuatro bordes de la caja de un elemento.

## Ancho

**border-top-width, border-right-width, border-bottom-width, border-left-width :**Establece la anchura de cada uno de los cuatro bordes de los elementos Se aplica a Todos los elementos el valor inicial es **Medium**.

**Valores ( unidad de medida | thin | medium | thick ) | inherit**

**Border-width:** Establece la anchura de todos los bordes del elemento, Se aplica a Todos los elementos el valor inicial es **Medium**.

**Valores ( unidad de medida | thin | medium | thick ) {1, 4} | inherit**

## Color

**border-top-color, border-right-color, border-bottom-color, border-left-color** Establece el color de cada uno de los cuatro bordes de los elementos Se aplica a Todos los elementos. El Valor inicial -.

**Valores color | transparent | inherit**

**border-color** Establece el color de todos los bordes del elemento. Todos los elementos. Valor inicial -.

**Valores ( color | transparent ) {1, 4} | inherit**

# Bordes

## Estilo

**border-top-style, border-right-style, border-bottom-style, border-left-style :**Establece el estilo de cada uno de los cuatro bordes de los elementos Todos los elementos el valor inicial es **none**.

**Valores none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset | inherit**

**Border-style:** Establece el estilo de todos los bordes del elemento se aplica a Todos los elementos el valor inicial -.

**Valores (none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset ) {1, 4} | inherit**

## Propiedades shorthand

**border-top, border-right, border-bottom, border-left** Establece el estilo completo de cada uno de los cuatro bordes de los elementos Se aplica a Todos los elementos. El Valor inicial -.

**Valores ( unidad de medida_borde || color_borde || estilo_borde ) | inherit**

**border** Establece el estilo completo de todos los bordes de los elementos se aplica a todos los elementos. Valor inicial -.

**Valores ( unidad de medida_borde || color_borde || estilo_borde ) | inherit**

# Fondos

El fondo puede ser un color simple o una imagen. El fondo solamente se visualiza en el área ocupada por el contenido y su relleno.

**background-color :** Establece un color de fondo para los elementos, Se aplica a Todos los elementos Valor inicial transparent.

**Valores color | transparent | inherit**

**background-image:** Establece una imagen como fondo para los elementos se aplica a Todos los elementos el valor inicial none.

**Valores url | none | inherit**

**background-repeat** Controla la forma en la que se repiten las imágenes de fondo Se aplica a Todos los elementos. El Valor inicial repeat.

**Valores repeat | repeat-x | repeat-y | no-repeat | inherit**

**background-position** Controla la posición en la que se muestra la imagen en el fondo del elemento se aplica a todos los elementos. Valor inicial 0% 0%.

**Valores ( ( porcentaje | unidad de medida | left | center | right ) ( porcentaje | unidad de medida | top | center | bottom )? ) | ( ( left | center | right ) || ( top | center | bottom ) ) | inherit**

**background-attachment** Controla la forma en la que se visualiza la imagen de fondo: permanece fija cuando se hace scroll en la ventana del navegador o se desplaza junto con la ventana se aplica a todos los elementos. Valor inicial scroll.

**Valores scroll | fixed | inherit**

**background** Establece todas las propiedades del fondo de un elemento Se aplica a Todos los elementos.

**Valores ( background-color || background-image || background-repeat || background-attachment || background-position ) | inherit**

# Posicionamiento y visualización

Para cumplir con el modelo de cajas los navegadores crean una caja para representar a cada elemento de la página HTML según los siguientes factores.

- Las propiedades width y height de la caja (si están establecidas).
- El tipo de cada elemento HTML (elemento de bloque o elemento en línea).
- Posicionamiento de la caja (normal, relativo, absoluto, fijo o flotante).
- Las relaciones entre elementos (dónde se encuentra cada elemento, elementos descendientes, etc.)
- Otro tipo de información, como por ejemplo el tamaño de las imágenes y el tamaño de la ventana del navegador.

**Tipos de elementos** Los elementos de bloque siempre empiezan en una nueva línea y ocupan todo el espacio disponible hasta el final de la línea. Por su parte, los elementos en línea no empiezan necesariamente en nueva línea y sólo ocupan el espacio necesario para mostrar sus contenidos.

**Posicionamiento** El estándar de **CSS** define cinco modelos diferentes para posicionar una caja.

**normal o estático:** posicionamiento si no se indica lo contrario.

**relativo**: posicionar una caja desplazada según el posicionamiento normal

**absoluto**: la posición de una caja se establece de forma absoluta.

**fijo**:caja en un elemento inamovible.

**flotante**: desplaza las cajas todo lo posible hacia la izquierda o hacia la derecha de la línea en la que se encuentran.

# Posicionamiento y visualización

El posicionamiento de una caja se establece mediante la propiedad position:

**Position** : Selecciona el posicionamiento con el que se mostrará el elemento. El valor inicial es static se aplica a todos los elementos

**Valores static | relative | absolute | fixed | inherit**

El significado de cada uno de los posibles valores:

**static**: se ignoran los valores de las propiedades top, right, bottom y left.

**relative**: la caja se controla con las propiedades top, right, bottom y left.

**absolute**: se controla con las propiedades top, right, bottom y left, pero las coordenadas del posicionamiento de su elemento contenedor.

**fixed**: el elemento permanece inamovible en la pantalla.

**CSS** define cuatro propiedades llamadas top, right, bottom y left para controlar el desplazamiento de las cajas posicionadas.

**top, right, bottom, left** Indican el desplazamiento horizontal y vertical del elemento respecto de su posición original, y se aplica a Todos los elementos posicionados el Valor inicial es auto.

**Valores unidad de medida | porcentaje | auto | inherit**

En el caso del posicionamiento relativo, cada propiedad indica el desplazamiento del elemento desde la posición original de su borde superior/derecho/inferior/izquierdo. Si el posicionamiento es absoluto, las propiedades están con respecto del borde superior/derecho/inferior/izquierdo de su primer elemento padre posicionado.

En cualquiera de los dos casos, si el desplazamiento se indica en forma de porcentaje, se refiere al porcentaje sobre la anchura (propiedades right y left) o altura (propiedades top y bottom) del elemento

# Posicionamiento y visualización

**Visualización** : Las propiedades display y visibility controlan la visualización de los elementos. La propiedad display permite ocultar completamente un elemento y Por otra parte, la propiedad visibility permite hacer invisible un elemento.

**Display:** Permite controlar la forma de visualizar un elemento e incluso ocultarlo. Se aplica a Todos los elementos y su Valor inicial inline.

**Valores inline | block | none | list-item | run-in | inline-block | table | inline-table | table-row-group | table-header-group | table-footer-group | table-row | table-column-group | table-column | table-cell | table-caption | inherit**

**visibility:** Permite hacer visibles e invisibles a los elementos, Se aplica a Todos los elementos su Valor inicial visible.

**Valores visible | hidden | collapse | inherit**

**Relación entre display, float y position:** Cuando se establecen las propiedades display, float y position sobre una misma caja:

Si display vale none, se ignoran las propiedades float y position y la caja no se muestra en la página.

Si position vale absolute o fixed, la caja se posiciona de forma absoluta, se considera que float vale none y la propiedad display vale block tanto para los elementos en línea como para los elementos de bloque. La posición de la caja se determina mediante el valor de las propiedades top, right, bottom y left.

En cualquier otro caso, si float tiene un valor distinto de none, la caja se posiciona de forma flotante y la propiedad display vale block tanto para los elementos en línea como para los elementos de bloque

# Posicionamiento y visualización

**overflow:** Permite controlar los contenidos sobrantes de un elemento. Se aplica a Elementos de bloque y celdas de tablas y su Valor inicial visible.

**Valores visible | hidden | scroll | auto | inherit**

**Los valores de la propiedad overflow tienen el siguiente significado:**

**visible:** el contenido no se corta y se muestra sobresaliendo la zona reservada para visualizar el elemento.

**hidden:** el contenido sobrante se oculta y sólo se visualiza.

**scroll:** solamente se visualiza el contenido que cabe dentro de la zona reservada para el elemento, pero también se muestran barras de scroll.

**auto:** el comportamiento depende del navegador, aunque normalmente es el mismo que la propiedad scroll.

**z-index** Establece el nivel tridimensional en el que se muestra el elemento. Se aplica a Elementos que han sido posicionados explícitamente. El Valor inicial auto.

**Valores auto | numero | inherit**

Cuanto más alto sea el valor numérico, más cerca del usuario se muestra la caja. Un elemento con z-index: 10 se muestra por encima de los elementos con z-index: 8 o z-index: 9, pero por debajo de elementos con z-index: 20 o z-index: 50.

# Texto

CSS define numerosas propiedades para modificar la apariencia del texto. A demas existe las de alineación

**color:** Establece el color de letra utilizado para el texto Se aplica a todos los Elementos y su Valor inicial depende del navegador.

**color | inherit**

**font-family:** Establece el tipo de letra utilizado para el texto.Se aplica a todos los Elementos y su Valor inicial depende del navegador.

**Valores (( nombre_familia | familia_generica ) (,nombre_familia | familia_generica)* ) | inherit**

**font-size:** Establece el tamaño de letra utilizado para el textol. Se aplica a todos los Elementos y su Valor inicial medium.

**Valores tamaño_absoluto | tamaño_relativo | unidad de medida | porcentaje | inherit**

**font-weight** Establece la anchura de la letra utilizada para el texto. Se aplica a todos los Elementos . El Valor inicial normal.

**normal | bold | bolder | lighter | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900 | inherit**

**font-style** Establece el estilo de la letra utilizada para el texto. Se aplica a todos los Elementos . El Valor inicial normal.

**Valores normal | italic | oblique | inherit**

**font-variant** Establece el estilo alternativo de la letra utilizada para el texto. Se aplica a todos los Elementos . El Valor inicial normal.

**Valores normal | small-caps | inherit**

**font** Permite indicar de forma directa todas las propiedades de la tipografía de un texto. Se aplica a todos los Elementos . El Valor inicial -.

**( ( font-style || font-variant || font-weight )? font-size ( / line-height )? font-family ) | caption | icon | menu | message-box | small-caption | status-bar | inheritt**

# Adicional estan

## Enlaces

**Estilos básicos** : Tamaño, color y decoración. Pseudo-clases.
**Estilos avanzados**: Decoración personalizada. Imágenes según el tipo de enlace. Mostrar los enlaces como si fueran botones.

## Imágenes

**Estilos básicos:** Establecer la anchura y altura de las imágenes. Eliminar el borde de las imágenes con enlaces.

**Estilos avanzados**: Sombra (drop shadow).

## Tablas:

**Estilos básicos:** border-collapse, border-spacing

**Estilos avanzados:** empty-cells, caption-side

## Formularios

**Estilos básicos** : Mostrar un botón como un enlace, Mejoras en los campos de texto, Labels alineadas y formateadas.

**Estilos avanzados**: Formulario en varias columnas, Resaltar el campo seleccionado

## Listas

**Estilos básicos** : Viñetas personalizadas, Menú vertical, Menú vertical avanzado.

**Estilos avanzados**: Menú horizontal básico, Menús avanzados,

# Adicional están

## Layout

Las principales ventajas de diseñar la estructura de las páginas web con CSS en vez de con tablas HTML son las siguientes: Mantenimiento, Accesibilidad, Velocidad de carga, Semántica.

Centrar una página horizontalmente: la propiedad margin de CSS, centrar una página web horizontalmente.

Centrar una página verticalmente:

```
#contenedor {
  width: 500px;
  height: 500px;

  position: absolute;
  top: 50%;
  left: 50%;

  margin-top: -250px;   /* height/2 = 500px / 2 */
  margin-left: -250px;  /* width/2 = 500px / 2 */
}
```

## Diseño a 2 columnas con cabecera y pie de página

```
#contenedor {
}
#cabecera {
}
#menu {
  float: left;
  width: 15%;
}
#contenido {
  float: left;
  width: 85%;
}
#pie {
  clear: both;
}
```

## Diseño a 3 columnas con cabecera y pie de página Se modifica y se adiciona

```
#contenido #principal {
  float: left;
  width: 80%;
}
#contenido #secundario {
  float: left;
  width: 20%;
}
```

## animaciones CSS

CSS3 permiten animar la transición entre un estilo CSS y otro. Las animaciones constan de dos componentes: un estilo que describe la animación CSS y un conjunto de fotogramas que indican su estado inicial y final, así como posibles puntos intermedios en la misma.

- Las animaciones CSS tienen tres ventajas principales sobre las técnicas tradicionales de animación basada en scripts:
- fáciles de usar para animaciones sencillas
- La animación se muestra correctamente, incluso en equipos poco potentes.
- Al ser el navegador quien controle la secuencia de la animación, permitimos que optimice el rendimiento y eficiencia de la misma

Con ellas no configuramos la apariencia actual de la animación, para ello disponemos de @keyframes como describiremos más adelante .

Las subpropiedades de animation son:

Animation-delay:Tiempo de retardo entre el momento en que el elemento se carga y el comienzo de la secuencia de la animación.

Animation-direction: Indica si la animación debe retroceder hasta el fotograma de inicio.

Animation-duration: Indica el tiempo que la animación consume en completar su ciclo.

Animation-iteration-count El número de veces que se repite. infinite para repetir la animación indefinidamente.

Animation-name: Especifica el nombre de la regla @keyframes que describe los fotogramas de la animación.

Animation-play-state: Permite pausar y reanudar la secuencia de la animación

Animation-timing-function: Indica el ritmo de la animación.

Animation-fill-mode: Especifica qué valores tendrán las propiedades después de finalizar la animación.

Universidad Industrial de Santander

# Maquetación

# Maquetación Digital

FrameWork, es un sistema preprogramado desde el que podemos empezar un proyecto, siguiendo unas reglas o instrucciones predefinidas. Los FrameWorks de maquetación son herramientas para maquetar webs de forma sencilla y rápida. Es muy útil, sobretodo, para crear estructuras de columnas. Además tienen compatibilidad con diseño responsive, por lo que nos permiten crear una web autoajustable para smartphones y tablets.

La mayoría se basan en un sistema de doce columnas que se conoce como Grid. Establece un estándar de 960 píxeles de ancho en el que podemos ver la Web centrada en pantallas de más de 1024 píxeles de ancho. Recomiendo visitar la Web de 960.gs y descargar las plantillas, sobre todo para diseño Web, si no estáis acostumbrados a la maquetación por columnas en Web

**herramientas para maquetar webs**

**Bootstrap:** Este es el FrameWork que siempre uso. Desarrollado por los creadores de Twitter. Viene con una biblioteca de funcionalidades maquetadas muy amplia. También con muchas funcionalidades Javascript uno de los más completos, versión 3.0.

**Foundation**: Bastante similar a Bootstrap. Está más orientado a páginas web, opción es más sencilla que Bootstrap, por lo que te puede ser más fácil de interpretar. actualización constantemente.

**Ink**: FrameWork muy completo, con diseño muy moderno y actual. Incluye un gran número de funcionalidades, tanto CSS3 como Javascript. Compatible con tipografías WebFont y con integración de HTML5.

**KickStart**: Librería muy completa, pero con las opciones mínimas, tanto para CSS como de JS. Contiene los elementos más importantes.

# Maquetación Digital

## herramientas para maquetar webs

**Ivory**: Sistema muy minimalista. Se centra en los CSS básicos para maquetar columnas. No tiene funcionalidades Javascript, por lo que reduce todavía más su peso.

**Tuktuk**: Con diseño moderno, destaca por su optimización (la librería solo pesa 9kb). Da más importancia a los CSS que a las funcionalidades Javascript.

**Skeleton**: Uno de los FrameWorks más sencillos y minimalistas. Lleva las opciones justas y necesarias para maquetar columnas, pesa poco dentro del proyecto y optimizamosel tiempo de descarga de la Web.

**Neat**: Para programadores avanzados. Usa directamente el lenguaje SASS y Bourbon para CSS. Implica una gran optimización de los estilos CSS y permite programar a un nivel superior. Pero el código es más complejo y requiere conocimientos extensos en el sublenguaje SASS de CSS.

**MaterializeCSS** permite implementar de una manera muy sencilla las guías de diseño de **Material Design**. Aprende a aplicarlo en tu sitio web y Materialize CSS no requiere jQuery para funcionar.

Universidad Industrial de Santander

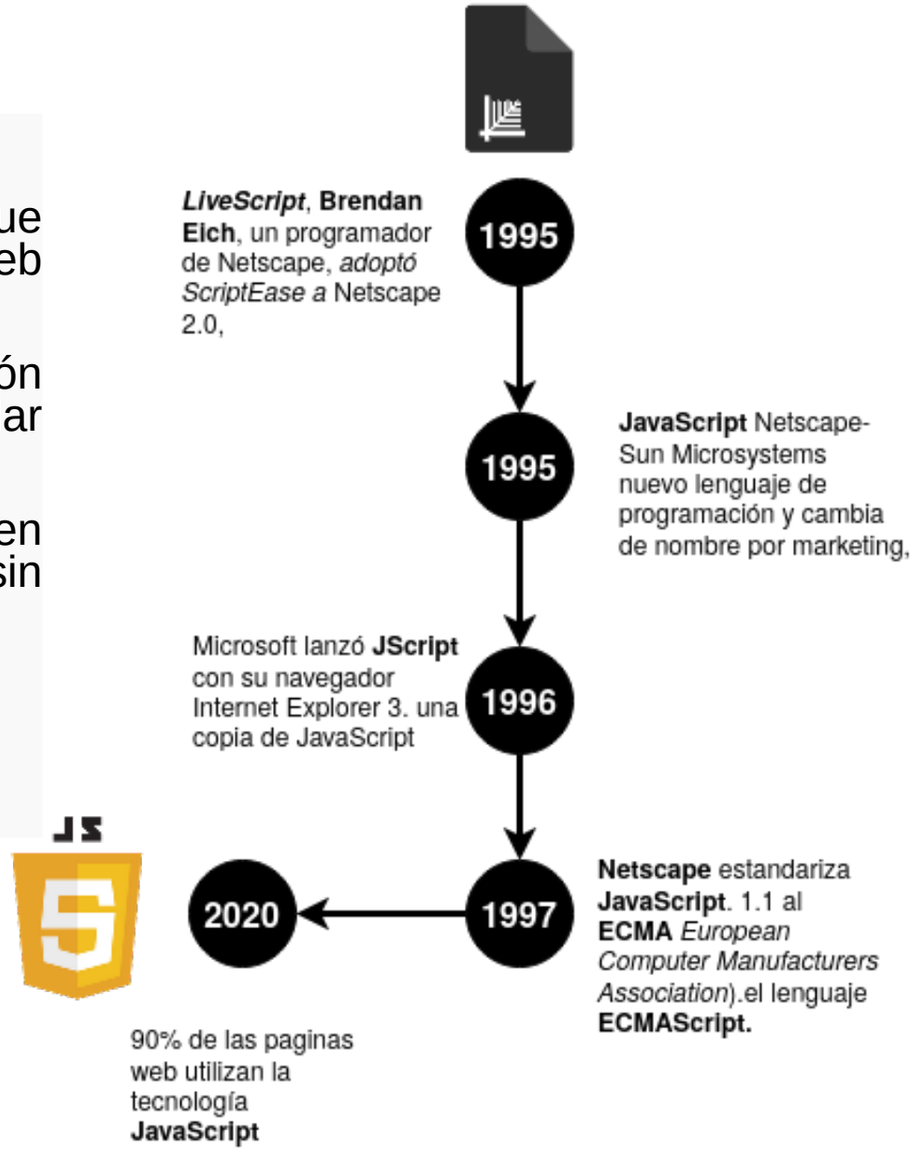
# JavaScript

## Qué es

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas.

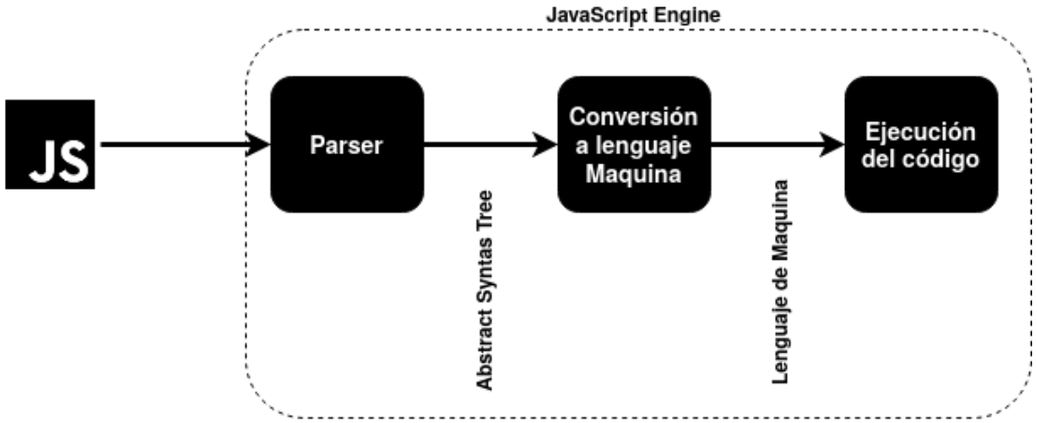
**JavaScript** es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos.

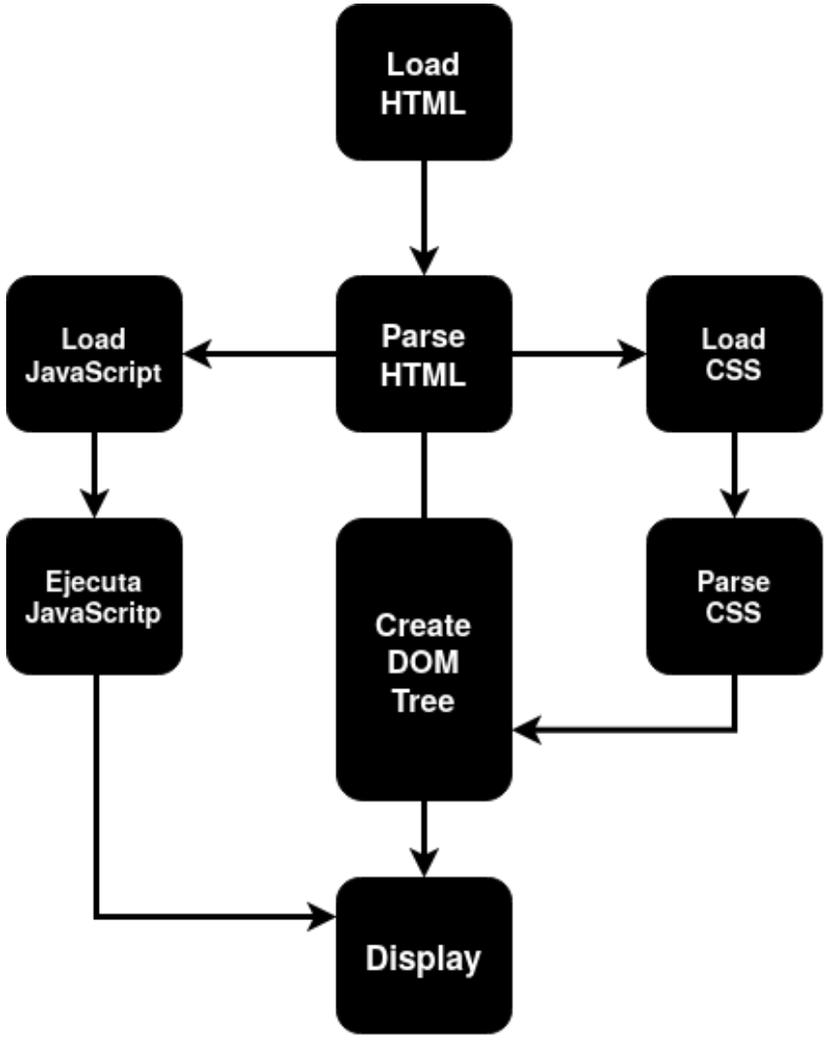
Los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios.



## Como funciona

El documento HTML donde pueden existir referencias o relaciones a otros documentos, como archivos CSS o archivos Javascript el navegador lo descarga y lo aplica al documento HTML, cambiando su apariencia visual. De la misma forma, si encuentra una referencia a un archivo Javascript, el navegador lo descarga y ejecuta las órdenes o acciones que allí se indican, mediante El Javascript Engine que interpreta el código JavaScript y ejecuta un script acorde a las instrucciones dadas.



# Cómo incluir JavaScript

Hay tres formas para incluir código JavaScript en las páginas web.

**en el mismo documento**

El código JavaScript se encierra entre etiquetas <script> y se incluye en cualquier parte del documento. Pero se recomienda definir el código JavaScript dentro de la cabecera del documento (dentro de la etiqueta <head>)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Ejemplo de código JavaScript en el propio documento</title>
<script type="text/javascript">
  alert("Un mensaje de prueba");
</script>
</head>
<body>
```

**en un archivo externo**

Las instrucciones JavaScript pueden estar en un archivo externo de tipo JavaScript enlazada mediante la etiqueta <script>. Se pueden crear todos los archivos JavaScript que sean necesarios como necesite

Los archivos de tipo JavaScript son documentos normales de texto con la extensión .js

```
<!DOCTYPE html >
<html >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Ejemplo de código JavaScript en el propio documento</title>
<script type="text/javascript" src="/js/codigo.js"></script>
</head>
<body>
```

# Cómo incluir JavaScript

## en los elementos

Método menos utilizado, ya que consiste en incluir trozos de JavaScript dentro del código HTML de la página:

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Ejemplo de código JavaScript en el propio documento</title>
</head>

<body>
<p onclick="alert('Un mensaje de prueba')">Un párrafo de texto.</p>
</body>
</html>
```

## Etiqueta <noscript>

la etiqueta <noscript> para mostrar un mensaje al usuario cuando su navegador no puede ejecutar JavaScript

```
<head> ... </head>
<body>
<noscript>
  <p>Bienvenido a Mi Sitio</p>
  <p>La página que estás viendo requiere para su funcionamiento el uso de JavaScript.
Si lo has deshabilitado intencionadamente, por favor vuelve a activarlo.</p>
</noscript>
</body>
```

# Palabras Basicas de JavaScript

**Script**

cada uno de los programas, aplicaciones o trozos de código creados con el lenguaje de programación JavaScript.

**Sentencia**

cada una de las instrucciones que forman un script.

**Palabras reservadas**

**break, case, catch, continue, default, delete, do, else, finally, for, function, if, in, instanceof, new, return, switch, this, throw, try, typeof, var, void, while, with.**

## Sintaxis

- **No se tienen en cuenta los espacios en blanco y las nuevas líneas.**
- **Se distinguen las mayúsculas y minúsculas.**
- **No se define el tipo de las variables.**
- **No es necesario terminar cada sentencia con el carácter de punto y coma (;).**
- **Se pueden incluir comentarios.**

## Posibilidades y limitaciones

- JavaScript fue diseñado de forma que se ejecutara en un entorno muy limitado.
- scripts de JavaScript no pueden comunicarse con recursos que no pertenezcan al mismo dominio desde el que se descargó el script
- Los scripts tampoco pueden cerrar ventanas que no hayan abierto esos mismos scripts.
- Los scripts no pueden acceder a los archivos del ordenador del usuario (ni en modo lectura ni en modo escritura) y tampoco pueden leer o modificar las preferencias del navegador.
- Si la ejecución de un script dura demasiado tiempo (ejem un error de programación) el navegador informa al usuario de que un script está consumiendo demasiados recursos y le da la posibilidad de detener su ejecución.

# Programación básica

## Variables

Las variables en JavaScript se crean mediante la palabra reservada **var.** En JavaScript no es obligatorio inicializar las variables, ya que se pueden declarar por una parte y asignarles un valor posteriormente.

El nombre de una variable también se conoce como identificador y debe cumplir las siguientes normas:

- Sólo puede estar formado por letras, números y los símbolos $ (dólar) y _ (guión bajo).
- El primer carácter no puede ser un número.

```
var numero_1 = 3;
var numero_2 = 1;
resultado = numero_1 + numero_2;
var $_a__$4;
```

## Tipos de variables

depende del tipo de valor que se quiere almacena

**Numéricas:** el valor se asigna indicando directamente el número entero o decimal.

```
var iva = 16;        // variable tipo entero
var total = 23.67;  // variable tipo decimal
```

**Cadenas de texto: S**e encierra el valor entre comillas dobles o simples, para delimitar su comienzo y su final.

```
var tex1 = "frase con 'comillas simples' dentro";
var tex2 ='Una frase con "comillas dobles" dentro';
```

**Cadenas de texto:** Para definir un array, se utilizan los caracteres [ y ] para delimitar su comienzo y su final y se utiliza el carácter , (coma) para separar sus elementos.

```
var nombre_array = [valor1, valor2, ..., valorN];
```

**Booleanos:** Una variable que solamente puede tomar dos valores: true (verdadero) o false (falso).

```
var clienteRegistrado = false;
```

# Programación básica

## Operadores

**Asignación:**El símbolo utilizado es =;

```
var numero1 = 3;
```

**Incremento y decremento**: incremento el prefijo ++, decremento –.

Lógicos:

- Negacion : !;
- AND : &&;
- OR : ||

**Matemáticos**:suma (+), resta (-), multiplicación (*), división (/) y modulo (%).

**Relacionales**: mayor que (>), menor que (<), mayor o igual (>=), menor o igual (<=), igual que (==) y distinto de (!=).

# Programación básica

## Estructuras de control

### if...else

```
if(condicion) {
  ...
}
else {
  ...
}
```

### for

```
for(inicializacion; condicion; actualizacion) {
  ...
}
```

### for...in:

```
for(indice in array) {
  ...
}
```

## Funciones para cadenas de texto

**Length:** calcula la longitud de una cadena de texto.

**+:** se emplea para concatenar, también se puede utilizar la función **concat()**.

**toUpperCase();** transforma en mayúsculas, **toLowerCase()** minúsculas.

**charAt(posicion):** obtiene el carácter que se encuentra en la posición indicada.

**indexOf(caracter):** obtiene la primer posición del carácter indicado, Si la cadena no contiene el carácter, la función devuelve el valor -1 su función análoga es lastIndexOf(caracter) obtiene la última posición del carácter.

**substring(inicio, final),** extrae una porción de una cadena de texto.

**split(separador):** convierte una cadena de texto en un array de cadenas de texto

# Programación básica

## Funciones para arrays

**Length:** calcula el número de elementos de un array.

**concat():** se emplea para concatenar los elementos de varios arrays,

**join(separador**): es la función contraria a split().

**pop()**:elimina el último elemento del array y lo devuelve.

**push()**: añade un elemento al final del array.

**shift():** elimina el primer elemento del array y lo devuelve.

**unshift():** añade un elemento al principio del array.

**reverse()**: modifica un array colocando sus elementos en el orden inverso a su posición original.

## Funciones para números

**NaN**: (del inglés, "Not a Number") JavaScript emplea el valor NaN para indicar un valor numérico no definido.

**isNaN():** permite proteger a la aplicación de posibles valores numéricos no definidos

**Infinity**, hace referencia a un valor numérico infinito y positivo (también existe el valor –Infinity para los infinitos negativos).

**toFixed(digitos):** devuelve el número original con tantos decimales como los indicados por el parámetro digitos y realiza los redondeos necesarios.

## Acceso a los nodos de DOM

**getElementsByTagName():** obtiene todos los elementos de la página XHTML cuya etiqueta sea igual que el parámetro que se le pasa a la función.

```
var parrafos = document.getElementsByTagName("p");

for(var i=0; i<parrafos.length; i++) {
  var parrafo = parrafos[i];
}
```

**getElementsByName():** La función es similar a la anterior, pero en este caso se buscan los elementos cuyo atributo name sea igual al parámetro proporcionado.

**getElementById():** es la más utilizada cuando se desarrollan aplicaciones web dinámicas. Se trata de la función preferida para acceder directamente a un nodo y poder leer o modificar sus propiedades.

## Creación y eliminación de nodos

**Creación de elementos:** ya que implica la utilización de tres funciones DOM:

**createElement(etiqueta)**: crea un nodo de tipo Element que representa al elemento XHTML cuya etiqueta se pasa como parámetro.

**createTextNode(contenido)**: crea un nodo de tipo Text que almacena el contenido textual de los elementos XHTML.

**nodoPadre.appendChild(nodoHijo)**: añade un nodo como hijo de otro nodo. Se debe utilizar al menos dos veces con los nodos habituales: en primer lugar se añade el nodo Text como hijo del nodo Element y a continuación se añade el nodo Element como hijo de algún nodo de la página.

**Eliminación de nodo**s: solamente es necesario utilizar la función removeChild():

# Modelo básico de eventos

| Evento | Descripción | Elementos definido |
|---|---|---|
| onblur | Deseleccionar el elemento | <button>, <input>, <label>, <select>, <textarea>, <body> |
| onchange | Deseleccionar un elemento que se ha modificado | <input>, <select>, <textarea> |
| onclick | Pinchar y soltar el ratón | Todos los elementos |
| ondblclick | Pinchar dos veces seguidas con el ratón | Todos los elementos |
| onfocus | Seleccionar un elemento | <button>, <input>, <label>, <select>, <textarea>, <body> |
| onkeydown | Pulsar una tecla (sin soltar) | Elementos de formulario y <body> |
| onkeypress | Pulsar una tecla | Elementos de formulario y <body> |
| onkeyup | Soltar una tecla pulsada | Elementos de formulario y <body> |
| onload | La página se ha cargado completamente | <body> |
| onmousedown | Pulsar (sin soltar) un botón del ratón | Todos los elementos |
| onmousemove | Mover el ratón | Todos los elementos |
| onmouseout | El ratón "sale" del elemento (pasa por encima de otro elemento) | Todos los elementos |
| onmouseover | El ratón "entra" en el elemento (pasa por encima del elemento) | Todos los elementos |
| onmouseup | Soltar el botón que estaba pulsado en el ratón | Todos los elementos |
| onreset | Inicializar el formulario (borrar todos sus datos) | <form> |
| onresize | Se ha modificado el tamaño de la ventana del navegador | <body> |
| onselect | Seleccionar un texto | <input>, <textarea> |
| onsubmit | Enviar el formulario | <form> |
| onunload | Se abandona la página (por ejemplo al cerrar el navegador) | <body> |

**Manejadores de eventos:** Para que los eventos resulten útiles, se deben asociar funciones o código JavaScript a cada evento:

**Manejadores de eventos como atributos:** el código se incluye en un atributo del propio elemento

```
<input type="button" value="Pinchame y verás" onclick="alert('Gracias por pinchar');" />
```

**Manejadores de eventos y variable this)**: En los eventos, se puede utilizar la variable this para referirse al elemento que ha provocado el evento

```
<div id="contenidos" style="width:150px; height:60px; border:thin solid silver" onmouseover="this.style.borderColor='black';" onmouseout="this.style.borderColor='silver';">
  Sección de contenidos...
</div>
```

# Modelo básico de eventos

**Manejadores de eventos como funciones externas:**es el método menos aconsejable de tratar con los eventos en JavaScript es recomendable para los casos más sencillos.

```
function muestraMensaje() {
  alert('Gracias por pinchar');
}

<input type="button" value="Pinchame y verás" onclick="muestraMensaje()" />
```

**Manejadores de eventos semánticos: los manejadores semánticos consiste en**

1) Asignar un identificador único al elemento mediante el atributo id.
2) Crear una función de JavaScript encargada de manejar el evento.
3) Asignar la función externa al evento correspondiente en el elemento deseado.

1) document.getElementById("pinchable");

2)document.getElementById("pinchable").onclick = ..

3)document.getElementById("pinchable").onclick = muestraMensaje;

```
<input id="pinchable" type="button" value="oprimetk" onclick="alert('Gracias');" />

// Función externa
function muestraMensaje() {
  alert('Gracias ');
}

// Asignar la función externa al elemento
document.getElementById("pinchable").onclick = muestraMensaje;

// Elemento XHTML
<input id="pinchable" type="button" value="oprimetk" />
```

Universidad Industrial de Santander

# JQUERY

## Historia

En 2006, **John Resig** era un desarrollador web trabajando en sus propios proyectos. Estaba frustrado con lo difícil que era escribir JavaScript que funcionara en distintos navegadores y decidió escribir su propia biblioteca de JS para arreglar su problema: **jQuery**.

A la comunidad de desarrollo le gusto la sencillez y el poder de jQuery, y la biblioteca se vuelve popular.

Hoy, es la biblioteca más popular de JS en la web, y es mantenida por la jQuery Foundation.

**«write less, do more».**

## Que es jQuery?

jQuery es una librería de Javascript, que sirven para disponer de paquetes de código reutilizables. El objetivo de la librerías es simplificar tareas complejas tales como:

- garantiza la compatibilidad entre los principales navegadores.
- Simplifica **document.querySelectorAll(````).** Ahora esta expresión se reduce a **$(````)**.
- Javascript es necesario modificar una propiedad nodo por nodo. Con jQuery no es necesario ningún tipo de bucle (for) para afectar todos los nodos del documento a la vez.
- jQuery reduce considerablemente la interacción con el DOM. Modificar, crear, eliminar y interactuar con el DOM
- Facilita enórmemente AJAX (Asynchronous JavaScript And XML)
- Es compatible con los potentes y modernos selectores de CSS3.

## Características

- Selección de elementos DOM.
- Interactividad y modificaciones del árbol DOM, incluyendo soporte para CSS 1-3 y un plugin básico de XPath.
- Eventos.
- Manipulación de la hoja de estilos CSS.
- Efectos y animaciones.
- Animaciones personalizadas.
- AJAX.
- Soporta extensiones.
- Utilidades varias como obtener información del navegador, operar con objetos y vectores, funciones para rutinas comunes, etc.
- Compatible con los navegadores Mozilla Firefox 2.0+, Internet Explorer 6+, Safari 3+, Opera 10.6+ y Google Chrome 8+**.**

## CVentajas

Es mucho más fácil que sus competidores tiene un ahorro substancial de tiempo y esfuerzo. La licencia open source de jQuery permite que la librería siempre cuente con soporte constante y rápido, publicándose actualizaciones de manera constante. La comunidad jQuery es activa y sumamente trabajadora.

Otra ventaja de jQuery sobre sus competidores como Flash y puro CSS es su excelente integración con AJAX.

En resumen:

- jQuery es flexible y rápido para el desarrollo web
- Viene con licencia MIT y es Open Source
- Tiene una excelente comunidad de soporte
- Tiene Plugins
- Bugs son resueltos rápidamente
- Excelente integración con AJAX

## Desventajas

Una de las principales desventajas de jQuery es la gran cantidad de versiones publicadas en el corto tiempo. No importa si usted está corriendo la última versión de jQuery, usted tendrá que hostear la librería usted mismo (y actualizarla constantemente), o descargar la librería desde Google (atractivo, pero puede traer problemas de incompatibilidad con el código).

Además del problema de las versiones, otras desventajas que podemos mencionar:

jQuery es fácil de instalar y aprender, inicialmente. Pero no es tan fácil si lo comparamos con CSS

Si jQuery es implementado inapropiadamente como un Framework, el entorno de desarrollo se puede salir de control.

## Como Incluirlo

**Hay 2 formas de hacerlo.**

1) 1 Generando una hoja externa que vamos a guardar en una carpeta dentro del directorio de carpetas de nuestra página web.
   - Vamos a jQuery.com descargamos la versión
   - Copiamos el código y lo pegamos.

1) 2 Importando el código desde un CDN (Content Delivery Network).

**<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"> </script>**

## Ejecución en cargado del documento

la velocidad de carga percibida por el usuario sea mucho mayor en jQuery disponemos de 4 formas de hacerlo.

- Dos de ellas están directamente extraídas de Javascript.
- Las otras dos son íntegramente pertenecientes a jQuery.

  **$(document).ready(function(){})** es el mas usado.

  **$(function(){})** el menos intuitivo

  **window.addEventListener("load",function(){})** Cualquier sistema válido en javascript

  **window.onload = function(){};** el evento onload como un atributo del nodo window.

## página ha cargado jQuery

**La instrucción typeof():** Dado un contenido o variable typeof(…) devuelve el tipo de variable.

Cuando escribimos $ nos referimos a la variable jQuery.

Cuando acompañamos $() estamos solicitando al navegador que ejecute de forma inmediata la función $.

**typeof($) debe devolver «function»**
**typeof($()) debe devolver «object».**

## Como llamar los nodos

Equivalencia entre jQuery y Javascript clásico

Dos expresiones equivalentes para llamar a los nodos de un documento web mediante selector CSS

### En Javascript

```
document.querySelectorAll('//selector CSS');
```

### En jQuery

```
$('//selector CSS')
```

Obtener un nodo cualquiera

```
document.querySelectorAll('negrilla')[0];
```

**$('.negrilla:eq(0)') o $('.negrilla').:eq(0)**

**Y podemos usar length, first(), last();**

## Sintaxis jQuery

La sintaxis de jQuery está hecha a la medida para seleccionar elementos HTML y algunas acciones en los elementos.

La sintaxis básica es: **$(selector).action()**

Un símbolo de $ para definir/acceder a jQuery

Un (**selector**) para "consultar o encontrar" elementos HTML

Un jQuery **action**() para ser realizado en el elemento seleccionado

Ejemplos:

**$(this).hide()** - Oculta el elemento actual.
**$("p").hide()** - Oculta todos los elementos <p>.
**$(".test").hide()** - Oculta todos los elementos con class="test".
**$("#test").hide()** - Oculta el elemento con id="test".

## Cuales son los selectores

De etiquetas,Por identificador, Por clase, Por varias clases, Asterisco “*”

Concatenar varios selectores distintos

**$('div')** – Devuelve todos los elementos DIV de una página web.

**$('#divDatosUsu')** – Devuelve el elemento que contenga el ID divDatosUsu.

**$('.boton')** – Devuelve todos los elementos que contengan la CLASS boton.

**$('*'**) – Devuelve todos los elementos existentes en una página web. Selectores multiples:

**$('#miDiv, .boton')** – Devuelve el elemento que contenga el ID miDiv y todos los elementos que contengan la CLASS boton.

**$('input[type=radio][name=empresa]')** – Devuelve todos los elementos del tipo input(radio) y que tengan el nombre empresa**.**

## Selectores Basicos

**Selector por elemento**
$(this)
Hace referencia a él mismo

**Selector por ID**
$('#Resultados')
Retorna el elemento que contenga el id con valor Resultados

**Selector por tag**
$('div')
Busca todos los div que hay en la pagina

**Selector por CSS**
$('.negrita')
Busca todos los elementos que contengan la clase CSS negrita

## Selectores Complejos

| | | |
|---|---|---|
| :first | :last | :not() |
| :eq(i) | :gt(i) | :lt(i) |
| :selected | :disabled | :has(sel) |
| :visible | :first-child | :last-child |
| :radio | :checkbox | :image |
| :even | :odd | :input |
| :checked | :button | :submit |
| :parent | :hidden | :text |
| :file | :empty | :enabled |
| :animated | | |

## Manipulación del DOM

jQuery nos permite manipular cualquier elemento de un documento HTML.

Algunas Funciones:

-append() - Agrega un elemento al DOM
-remove() - Elimina un elemento al DOM
-empty() - Limpia el DOM

Ejemplo:
$('#divDatosUsu').empty().append(' Hola Clase ');

## Manejo de eventos

jQuery nos permite manejar los eventos de cualquier elemento de un documento HTML.

Algunas Funciones:

-click() - Eventos click
-change() - Eventos change (cambios en los estados de elementos HTML)
- keypress() - Eventos de teclado
- live() - Eventos para elementos no existentes (jQuery 1.3+)

Ejemplo:
$('#miBoton').click(function (event){ //evento capturado });

## Acceder a los hijos, padres, hermanos.

**La instrucción typeof():** Dado un contenido o variable typeof(…) devuelve el tipo de variable.

Cuando escribimos $ nos referimos a la variable jQuery.

Cuando acompañamos $() estamos solicitando al navegador que ejecute de forma inmediata la función $.

| | |
|---|---|
| **elemento.documentElement;** | **elemento.parect()** |
| **node.nextElementSibling;** | **elemento.next()** |
| **node.previoustElementSibling;** | **elemento.prev()** |
| **elemento.childNodes** | **elemento.children()** |

## Consultar, modificar, insertar, eliminar un nodo

```
<p class="parrafo-1">soy el párrafo 1</p>
<p class="parrafo-2">soy el párrafo 2</p>
<p class="parrafo-3">soy el párrafo 3</p>
<p class="parrafo-4">soy el párrafo 4</p>
<p class="parrafo-5">soy el párrafo 5</p>
<p class="parrafo-6">soy el párrafo 6</p>
```

**Consulta**

### Jquery

```
console.log($("p").html())
console.log($("p").eq(0).html())
```

### híbrido entre jQ y JS

```
console.log($("p")[0].innerHTML)
```

## Acceder a los hijos, padres, hermanos.

**La instrucción typeof():** Dado un contenido o variable typeof(…) devuelve el tipo de variable.

Cuando escribimos $ nos referimos a la variable jQuery.

Cuando acompañamos $() estamos solicitando al navegador que ejecute de forma inmediata la función $.

| | |
|---|---|
| **elemento.documentElement;** | **elemento.parect()** |
| **node.nextElementSibling;** | **elemento.next()** |
| **node.previoustElementSibling;** | **elemento.prev()** |
| **elemento.childNodes** | **elemento.children()** |

## Consultar, modificar, insertar, eliminar un nodo

```
<p class="parrafo-1">soy el párrafo 1</p>
<p class="parrafo-2">soy el párrafo 2</p>
<p class="parrafo-3">soy el párrafo 3</p>
<p class="parrafo-4">soy el párrafo 4</p>
<p class="parrafo-5">soy el párrafo 5</p>
<p class="parrafo-6">soy el párrafo 6</p>
```

**Consulta**

### Jquery

```
console.log($("p").html())
console.log($("p").eq(0).html())
```

### híbrido entre jQ y JS

```
console.log($("p")[0].innerHTML)
```

## Animaciones

Nos permite agregar efectos visuales, animaciones y efectos a la mayoría de los elementos de un documento HTML.

Ejemplos:

**show() y hide()**: Muestra u oculta elementos.

**show(speed, callback)**: Especifica el tiempo de duración y el otro se llama cuando se termina.

**toggle()**: Si está oculto se muestra o viceversa.

**fade()**: Desvanece y aparece objetos.

## Consultar, modificar, insertar, eliminar un nodo

```
<p class="parrafo-1">soy el párrafo 1</p>
<p class="parrafo-2">soy el párrafo 2</p>
<p class="parrafo-3">soy el párrafo 3</p>
<p class="parrafo-4">soy el párrafo 4</p>
<p class="parrafo-5">soy el párrafo 5</p>
<p class="parrafo-6">soy el párrafo 6</p>
```

**Consulta**

### Jquery

```
console.log($("p").html())
console.log($("p").eq(0).html())
```

### híbrido entre jQ y JS

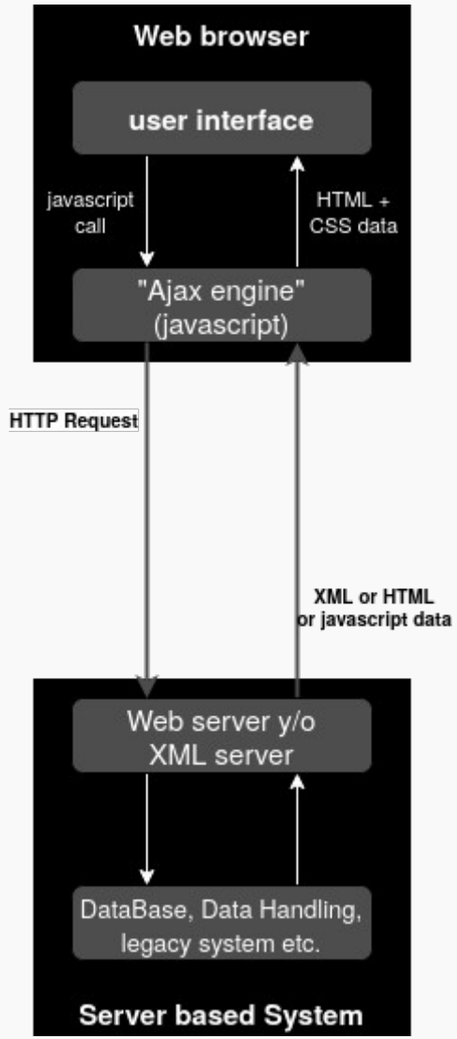
```
console.log($("p")[0].innerHTML)
```

# AJAX

Significa JavaScript asíncrono y XML (Asynchronous JavaScript and XML). Es un conjunto de técnicas de desarrollo web que permiten que las aplicaciones web funcionen de forma asíncrona, procesando cualquier solicitud al servidor en segundo plano.

AJAX es un conjunto de técnicas de desarrollo web.

- HTML/XHTML para el lenguaje principal y CSS para la presentación.
- Modelo de objetos del documento (DOM) para datos de visualización dinámicos y su interacción.
- XML para el intercambio de datos y XSLT para su manipulación. Muchos desarrolladores han comenzado a reemplazarlo por JSON porque es más similar a JavaScript en su forma.
- El objeto XMLHttpRequest para la comunicación asíncrona.
- Finalmente, el lenguaje de programación JavaScript para unir todas estas tecnologías.



## Tabla Comparativa

| Modelo convencional | Modelo AJAX |
|---|---|
| 1. Se envía una solicitud HTTP desde el navegador web al servidor. | 1. El navegador crea una llamada de JavaScript que luego activará XMLHttpRequest. |
| 2. El servidor recibe y, posteriormente, recupera los datos. | 2. En segundo plano, el navegador web crea una solicitud HTTP al servidor. |
| 3. El servidor envía los datos solicitados al navegador web. | 3. El servidor recibe, recupera y envía los datos al navegador web. |
| 4. El navegador web recibe los datos y vuelve a cargar la página para que aparezcan los datos. | 4. El navegador web recibe los datos solicitados que aparecerán directamente en la página. No se necesita recargar. |
| Durante este proceso, los usuarios no tienen más remedio que esperar hasta que se complete todo el proceso. No solo consume mucho tiempo, sino que también supone una carga innecesaria en el servidor. | |

## Metodo jQuery load()

El método load() jQuery es simple, pero carga métodos AJAX.

Sintaxis:

$(selector).load(URL,data,callback);

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("#div1").load("demo_test.txt");
  });
});
</script>
</head>
<body>
<div id="div1"><h2>Let jQuery AJAX Change This Text</h2></div>
<button>Get External Content</button>
</body>
</html>
```

## Solicitud HTTP: GET vs. POST

Dos métodos comúnmente utilizados para una solicitud-respuesta entre un cliente y un servidor son: GET y POST.

**GET**: solicita datos de un recurso especificado

**POST**: envía datos para su procesamiento a un recurso específico

## El metodo noConflict()

El uso de jQuery **noConflict()** se puede volver necesario cuando trabajamos con varias librerías de Java Script en la misma web

```
$.noConflict();
jQuery(document).ready(function(){
  jQuery("button").click(function(){
    jQuery("p").text("jQuery is still working!");
  });
});
```

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $.post("demo_test_post.asp",
    {
      name: "Donald Duck",
      city: "Duckburg"
    },
    function(data,status){
      alert("Data: " + data + "\nStatus: " + status);
    });
  });
});
</script>
</head>
<body>

<button>Send an HTTP POST request to a page and get the result back</button>

</body>
</html>
```

# jQuery Filters

Use jQuery to filter/search para elementos.

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("#myInput").on("keyup", function() {
    var value = $(this).val().toLowerCase();
    $("#myTable tr").filter(function() {
                                                $(this).toggle($(this).text().toLowerCase().indexOf(value) > -1)
    });
  });
});
</script>
```

```
<style>
table {
  font-family: arial, sans-serif;
  border-collapse: collapse;
  width: 100%;
}
td, th {
  border: 1px solid #dddddd;
  text-align: left;
  padding: 8px;
}
tr:nth-child(even) {
  background-color: #dddddd;
}
</style>
</head>
<body>
<h2>Filterable Table</h2>
<p>Type something in the input field to search the table for first names, last names or emails:</p>
<input id="myInput" type="text" placeholder="Search..">
<br><br>
```

# jQuery Filters

```
<table>
  <thead>
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
    <th>Email</th>
  </tr>
  </thead>
  <tbody id="myTable">
  <tr>
    <td>John</td>
    <td>Doe</td>
    <td>john@example.com</td>
  </tr>
  <tr>
    <td>Mary</td>
    <td>Moe</td>
    <td>mary@mail.com</td>
  </tr>
  <tr>
    <td>July</td>
    <td>Dooley</td>
    <td>july@greatstuff.com</td>
  </tr>
  <tr>
    <td>Anja</td>
    <td>Ravendale</td>
    <td>a_r@test.com</td>
  </tr>
  </tbody>
</table>
```

```
<p>Note that we start the search in tbody, to prevent filtering the table headers.</p>

</body>
</html>
```

| Method | Description |
|---|---|
| $.ajax() | Performs an async AJAX request |
| $.ajaxPrefilter() | Handle custom Ajax options or modify existing options before each request is sent and before they are processed by $.ajax() |
| $.ajaxSetup() | Sets the default values for future AJAX requests |
| $.ajaxTransport() | Creates an object that handles the actual transmission of Ajax data |
| $.get() | Loads data from a server using an AJAX HTTP GET request |
| $.getJSON() | Loads JSON-encoded data from a server using a HTTP GET request |
| $.parseJSON() | Deprecated in version 3.0, use JSON.parse() instead. Takes a well-formed JSON string and returns the resulting JavaScript value |
| $.getScript() | Loads (and executes) a JavaScript from a server using an AJAX HTTP GET request |
| $.param() | Creates a serialized representation of an array or object (can be used as URL query string for AJAX requests) |
| $.post() | Loads data from a server using an AJAX HTTP POST request |
| ajaxComplete() | Specifies a function to run when the AJAX request completes |
| ajaxError() | Specifies a function to run when the AJAX request completes with an error |
| ajaxSend() | Specifies a function to run before the AJAX request is sent |
| ajaxStart() | Specifies a function to run when the first AJAX request begins |
| ajaxStop() | Specifies a function to run when all AJAX requests have completed |
| ajaxSuccess() | Specifies a function to run when an AJAX request completes successfully |
| load() | Loads data from a server and puts the returned data into the selected element |
| serialize() | Encodes a set of form elements as a string for submission |
| serializeArray() | Encodes a set of form elements as an array of names and values |

Universidad Industrial de Santander

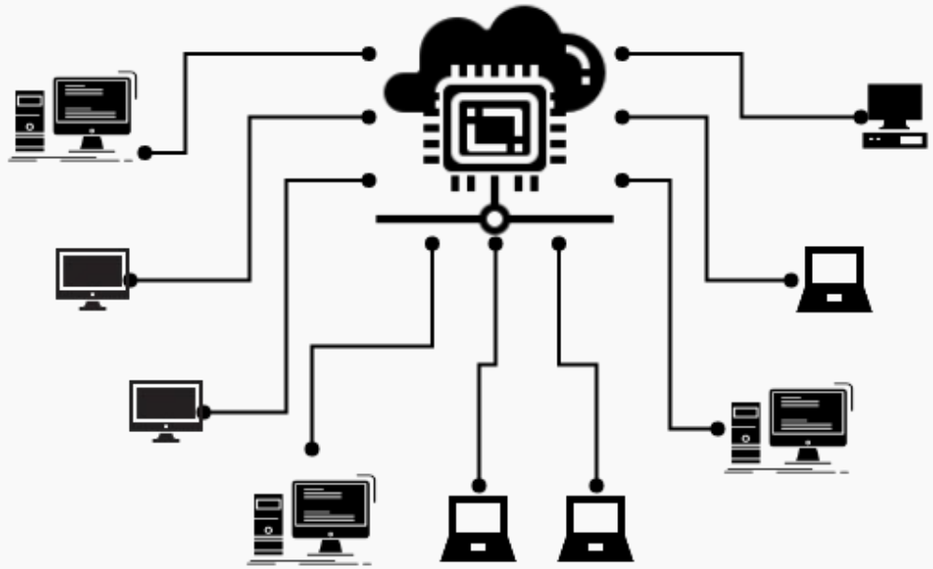
# Web Server

## Que es un Servidor Web

Un servidor web es un programa que implementa el protocolo HTTP (hypertext transfer protocol). Este protocolo pertenece a la capa de aplicación del modelo OSI y está diseñado para transferir lo que llamamos hipertextos, páginas web o páginas HTML (hypertext markup language): textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de música.

Es un programa que se ejecuta continuamente en un ordenador (también se emplea el término para referirse al ordenador que lo ejecuta), manteniéndose a la espera de peticiones por parte de un cliente (un navegador web) y que responde a estas peticiones adecuadamente, mediante una página web que se exhibirá en el navegador o mostrando el respectivo mensaje si se detectó algún error.

## Definición de Servidor



## Caracteristicas Servidor Web

almacena los archivos de una web y los proporciona a los clientes que los solicitan haciendo la transferencia de los archivos a través de la red mediante los navegadores. El cliente lo pide a través de su navegador y el servidor web lo envía al mismo navegador del cliente pare que este lo pueda visualizar.

Los archivos web incluyen texto, imágenes, videos, etc.. y que solo los navegadores pueden visualizar.

El servidor "sirve" (envía) el archivo web (por ejemplo una web en formato html) al navegador del cliente para que lo pueda visualizar. El servidor, el navegador y la comunicación a través de la red seguirán unas normas llamadas "protocolo HTTP".

## Hosting

El espacio que te dejan estos servidores para alojar tu web se llama Hosting. Hay dos tipos principales de hosting:

**Hosting Compartido**: en el servidor web hay varias páginas alojadas de distintos clientes.

**Hosting Dedicado**: tienes un servidor para ti solito donde puedes alojar tus webs. Lógicamente son más caros.

## Apache

Apache HTTP Server es un software de servidor web gratuito y de código abierto para plataformas Unix. Es mantenido y desarrollado por la Apache Software Foundation.

Le permite a los propietarios de sitios web servir contenido en la web, de ahí el nombre de . Es uno de los servidores web más antiguos y confiables, la primera versión lanzada en 1995.

Cuando alguien quiere visitar un sitio web, ingresa un nombre de dominio en la barra de direcciones de su navegador. Luego, el servidor web envía los archivos solicitados actuando como un repartidor virtual.

## caracteristicas

Hoy en día, el servidor Web Apache es el servidor más usado de Internet, con una utilización del 65% aproximadamente.

La primera aparición de Apache fue en Abril de 1995. Este servidor se sigue desarrollando “en Internet” como un proyecto de Software libre.

Las principales metas de su diseño son: velocidad, simplicidad, multiplataforma y facilidad del desarrollo distribuido.

Todo el código de fuente de Apache está escrito en C.

Es un servidor basados en procesos, utilizando la técnica pre-fork.

Ventajas

- Modular
- Open source
- Multi-plataforma
- Extensible
- Popular (facil conseguir ayuda/soporte)

## Apache

Apache HTTP Server es un software de servidor web gratuito y de código abierto para plataformas Unix. Es mantenido y desarrollado por la Apache Software Foundation.

Le permite a los propietarios de sitios web servir contenido en la web, de ahí el nombre de . Es uno de los servidores web más antiguos y confiables, la primera versión lanzada en 1995.

http://httpd.apache.org/docs/2.4/es/install.html

**Descarga Descarga la última versión desde http://httpd.apache.org/download.cgi**
**Extraer $ gzip -d httpd-NN.tar.gz**
**$ tar xvf httpd-NN.tar**
**$ cd httpd-NN**
**Configura $ ./configure --prefix=PREFIX**
**Compila $ make**
**Instala $ make install**
**Personalizalo $ vi PREFIX/conf/httpd.conf**
**Prueba $ PREFIX/bin/apachectl -k star**t

## caracteristicas

Hoy en día, el servidor Web Apache es el servidor más usado de Internet, con una utilización del 65% aproximadamente.

La primera aparición de Apache fue en Abril de 1995. Este servidor se sigue desarrollando “en Internet” como un proyecto de Software libre.

Las principales metas de su diseño son: velocidad, simplicidad, multiplataforma y facilidad del desarrollo distribuido.

Todo el código de fuente de Apache está escrito en C.

Es un servidor basados en procesos, utilizando la técnica pre-fork.

Ventajas

- Modular
- Open source
- Multi-plataforma
- Extensible
- Popular (facil conseguir ayuda/soporte)

## Tomcat

Estrictamente hablando, Tomcat no es un servidor web como Apache HTTPS Server o NGINX.

Comenzado en 1999 y desarrollado como un proyecto de código abierto por la Apache Software Foundation (ASF), Apache Tomcat es un contenedor Java Servlet, o contenedor web, que proporciona la funcionalidad extendida para interactuar con Java Servlets, al tiempo que implementa varias especificaciones técnicas de la plataforma Java: JavaServer Pages (JSP), Java Expression Language (Java EL) y WebSocket.

Este es un software que permite que un servidor web maneje contenido web dinámico basado en Java utilizando el protocolo HTTP. JSP es una tecnología similar que permite a los desarrolladores crear contenido dinámico utilizando documentos HTML o XML.

https://tomcat.apache.org/download-80.cgi

## Tomcat vs Apache

Si bien Apache es un servidor web HTTPS tradicional, optimizado para manejar contenido web estático y dinámico, carece de la capacidad de administrar Servlets Java y JSP. Tomcat, por otro lado, está casi totalmente orientado al contenido basado en Java.

Sin embargo, por sí solo, Tomcat no es particularmente eficiente como un servidor HTTP tradicional, por lo que Apache es una opción mucho mejor para sitios web dinámicos creados únicamente con un lenguaje como PHP.

# Nginx

NGINX, «engine-ex» servidor web de código abierto. En su versión inicial, funcionaba en servidores web HTTP. Hoy en día también sirve como proxy inverso, balanceador de carga HTTP y proxy de correo electrónico para IMAP, POP3 y SMTP.

NGINX fue lanzado oficialmente en octubre del 2004. El creador del software, Igor Sysoev, comenzó su proyecto en el 2002 como un intento de solucionar el problema C10k. Por esa razón, NGINX ofrece una arquitectura asíncrona y controlada por eventos, característica que hace de NGINX uno de los servidores más confiables para la velocidad y la escalabilidad.

Debido a su excelente capacidad para manejar muchas conexiones y a su velocidad, muchos sitios web de alto tráfico usan el servicio de NGINX. Google, Netflix, Adobe, Cloudflare, WordPress.com y muchos más.

NGINX trabaja con una arquitectura asíncrona y controlada por eventos. Esto significa que los hilos similares se administran bajo un proceso de trabajo, y cada proceso de trabajo contiene unidades más pequeñas llamadas conexiones de trabajo. Toda esta unidad es la responsable de manejar los hilos de las solicitudes. Las conexiones de trabajo entregan las solicitudes a un proceso de trabajo, que también lo enviará a su turno al proceso maestro. Finalmente, el proceso maestro proporciona el resultado de esas solicitudes.

NGINX puede procesar miles de solicitudes sin ninguna dificultad. Es una excelente opción para sitios web con mucho tráfico como comercio electrónico, motores de búsqueda y almacenamiento en la nube.

https://docs.nginx.com/nginx/admin-guide/installing-nginx/installing-nginx-open-source/