**Diseño y Desarrollo Web**

**Internet**

Internet (el internet o, también, la internet)​ es un conjunto descentralizado de redes de comunicación interconectadas que utilizan la familia de protocolos TCP/IP. Sus orígenes se remontan a 1969, cuando se estableció la primera conexión de computadoras, conocida como ARPANET, entre tres universidades en California.

La WWW (World wide web) es un conjunto de protocolos que permite, de forma sencilla, la consulta remota de archivos de hipertexto. Esta fue un desarrollo posterior (1990) y utiliza internet como medio de transmisión. Hay que aclarar que **NO ES INTERNET.** Es un sistema de informacion en línea que opera sobre internet. Tim Berners-Lee propuso la arquitectura de lo que se conoció como la World Wide Web. **Creó el primer servidor web, navegador web y página web** en su computadora en el laboratorio de investigación de física del CERN en 1990, donde el sistema consiste en una vasta colección de páginas web interconectadas.

Utiliza hipervínculos para conectar paginas web entre si, permitiendo a los usuarios navegar entre diferentes páginas.

Una forma fácil de verla es una red de computadoras interconectadas que se encargan de alojar servicios (servidores) y responder a las solicitudes de sus usuarios. Donde cada servicio tiene una dirección IP y un dominio asociado a esa dirección IP permitiendo que los usuarios puedan acceder a dicho servicio. Los navegadores son capaces de Interpretar los datos de estros servicios y reenderizarlos a los usuarios finales

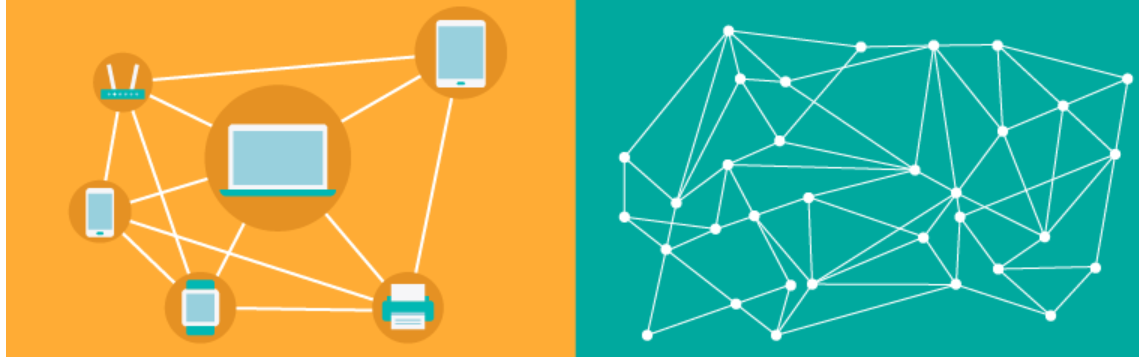
la Web es una de las muchas aplicaciones creadas sobre Internet. Existen otros servicios y protocolos en internet, aparte de la Web:

* el envío de correo electrónico (SMTP)
* la transmisión de archivos (FTP y P2P)
* las conversaciones en línea (IRC)
* la mensajería instantánea y presencia, la transmisión de contenido y comunicación multimedia —telefonía (VoIP), televisión (IPTV)
* los boletines electrónicos (NNTP)
* el acceso remoto a otros dispositivos (SSH y Telnet) o los juegos en línea.​

**Como Funciona Internet**

Internet se define como una gran “**red de redes”,** es decir, una red conectada a otra de manera continua y simultánea, pero para entenderlo mejor veamos primero qué es una red:

Cada uno de estos gráficos representa una especie de red: de computadoras, de puntos o de pesca. Lo que define a estas redes como tales es que existe “interconexión” entre sus componentes. Desde cualquier punto de la red podremos llegar a otro. Internet es justamente eso: una gran red donde todos los dispositivos están conectados entre sí.



**¿Como se comunican las redes de Internet?**

Para que una interconexión sea realmente global, todas las redes que se conectan a la gran red de redes deben hacerlo a través de un mismo protocolo o “lenguaje en común”. Es decir, e**l protocolo de comunicación de Internet debe ser una implementación estándar** que garantice la conexión desde cualquier origen hasta cualquier destino. Durante la década del 70’ en USA se creo el Protocolo de comunicación TCP/IP.

**Interconexión**

Probablemente la computadora que utilizamos a diario en la oficina esté conectada a una red de computadoras. En nuestra casa, la computadora portátil, tablet y celulares los vinculamos a un dispositivo inalámbrico que luego se conecta a la red de fibra óptica de la empresa que nos brinda servicio de Internet y de televisión por cable o telefonía. Éste se comunica a la red de otro Proveedor de Servicios de Internet o ISP (por sus siglas en inglés) más grande, que llega a un número mayor de hogares y empresas, y así sucesivamente. A su vez, estos proveedores de acceso internacional se interconectan con otros más grandes, denominados Carriers, a través de fibras ópticas transcontinentales y satélites, entre otros.

**Qué es un protocolo TCP/IP y cómo funciona?**

Es importante que los equipos tengan un modo común de comunicarse entre ellos. Para la mayoría de los equipos actuales, este modo es TCP/IP. TCP/IP suele venir integrado en los equipos y está automatizado en buena medida, aunque puede ser útil comprender el modelo TCP/IP, en especial si va a configurar un equipo para conectarlo a otro sistema.

**Qué significa TCP/IP?**

TCP/IP son las siglas de **Transmission Control Protocol/Internet Protocol** (Protocolo de control de transmisión/Protocolo de Internet). **TCP/IP es un conjunto de reglas estandarizadas que permiten a los equipos comunicarse en una red como Internet.**

**Qué hace exactamente TCP/IP? ¿Y cómo funciona?**

Por sí mismo, un equipo puede realizar determinadas tareas. Pero su potencia se multiplica cuando es capaz de comunicarse con otros. Una interacción determinada puede darse entre dos sistemas informáticos o involucrar cientos de sistemas. Sin embargo, **cada transacción se produce entre solo dos equipos cada vez. Para que esto suceda, los dos equipos deben saber cómo se espera que se comuniquen.**

**¿Cómo inician la conversación?**

**¿A quién le toca comunicarse?**

**¿Cómo sabe un equipo si su mensaje se ha transmitido correctamente?**

**¿Cómo terminan la conversación?**

Los equipos lo resuelven mediante protocolos. **Un protocolo es un conjunto de reglas convenido**. Cuando todos los equipos emplean el mismo protocolo, es posible transferir información. Cuando no es así, cunde el caos. **Cada fabricante tenía un sistema de comunicación propio entre sus máquinas**, pero dichos sistemas **no permitían la comunicación con los equipos de los demás fabricantes**. Pronto quedó claro que **era necesario un estándar convenido que permitiera a los equipos de todos los fabricantes comunicarse** entre ellos. Ese estándar es **TCP/IP.**

**¿En qué se diferencian TCP e IP?**

* IP es la parte que obtiene la dirección a la que se envían los datos.
* TCP se encarga de la entrega de los datos una vez hallada dicha dirección IP

**Analogía**: ‘La dirección IP es como el número de teléfono que se asigna a su smartphone. TCP es toda la tecnología que hace que el teléfono emita un timbre al recibir una llamada y que le permite hablar con alguien al otro lado de la línea’.

TCP/IP fue **desarrollado para especificar el modo en que los equipos transfieren** datos de un dispositivo a otro. TCP/IP **incide mucho en la precisión y dispone de varios pasos** para garantizar la correcta transmisión de los datos.

Lo que hace TCP/IP **es descomponer cada mensaje en paquetes** que se vuelven a ensamblar en el otro extremo. De hecho, **cada paquete podría tomar una ruta distinta** hasta el equipo de destino si la ruta deja de estar disponible o está muy congestionada.

TCP/IP **divide las distintas tareas de comunicación en capas**. Cada capa tiene una función distinta. Los **datos pasan por cuatro capas independientes** antes de recibirse en el otro extremo. A continuación, TCP/IP **recorre estas capas en orden inverso para reensamblar los datos y presentárselos al destinatario**.

**El propósito de las capas es crear un sistema estandarizado**, sin que los distintos fabricantes de hardware y software tengan que gestionar la comunicación por su cuenta.

**Las cuatro capas del modelo TCP/IP**

**Capa de enlace de datos**

La capa de enlace de datos (también denominada capa de enlace, capa de interfaz de red o capa física) es la que maneja las partes físicas del envío y recepción de datos mediante el cable Ethernet, la red inalámbrica, la tarjeta de interfaz de red, el controlador del dispositivo en el equipo, etcétera.

**Capa de Internet**

La capa de Internet (también denominada capa de red) controla el movimiento de los paquetes alrededor de la red.

**Capa de transporte**

La capa de transporte es la que proporciona una conexión de datos fiable entre dos dispositivos. Divide los datos en paquetes, hace acuse de recibo de los paquetes que recibe del otro dispositivo y se asegura de que el otro dispositivo haga acuse de recibo de los paquetes que recibe a su vez.

**Capa de aplicaciones**

La capa de aplicaciones es el grupo de aplicaciones que requiere comunicación de red. Es con lo que el usuario suele interactuar, como el correo electrónico y la mensajería. Como la capa inferior gestiona los detalles de la comunicación, las aplicaciones no tienen que preocuparse por ello.

**¿Mis paquetes de datos se mantienen privados?**

No. **Cuando los paquetes se transmiten entre equipos, son vulnerables y otros pueden verlos**. Esa es una de las razones por las que se aconseja evitar las redes Wi-Fi públicas para enviar datos que deban mantenerse privados. Para evitar esto puede cifrar los datos empleando una red privada virtual (VPN) o trabajando con páginas web con seguridad https.

**¿TCP/IP funciona con toda clase de direcciones IP?**

Hay varios tipos de direcciones IP, todas ellas utilizan TCP/IP.

* Las direcciones IP estáticas no cambian en ningún momento.
* Las direcciones IP dinámicas cambian, o al menos están diseñadas para cambiar

Con el número siempre creciente de dispositivos conectados, TCP/IP ha tenido un problema similar. Básicamente, **Internet se estaba quedando sin direcciones IP.** Por eso se **desarrolló una nueva versión de dirección IP denominada IPv6,** una alternativa a las direcciones IPv4 existentes.

**¿Cómo puedo saber cuál es mi dirección TCP/IP?**

Podés consultar tu IP pública desde una página web como <https://www.cual-es-mi-ip.net//>

**Arquitectura de una Página web**

La arquitectura cliente-servidor se basa en la idea de que las funcionalidades se dividen entre el cliente y el servidor. El cliente se encarga de la presentación y la interacción con el usuario, mientras que el servidor maneja la lógica de negocio, el almacenamiento de datos y otros procesos que no están directamente relacionados con la interfaz del usuario

**Server Side**

Python y Ruby son dos lenguajes de programación del lado del servidor que se utilizan principalmente para desarrollar páginas web server-side. Al igual que PHP, estos lenguajes permiten procesar datos y generar contenido dinámico en el servidor antes de ser enviado al client. Una página web server-side se genera dinámicamente en el servidor antes de ser enviada al cliente. Esto significa que el servidor procesa la lógica y los datos para generar la página final que se mostrará al usuario.

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

**Client Side**

Una página web cliente-side ejecuta la lógica y el procesamiento en el navegador web del cliente. Esto significa que todas las operaciones y procesamientos ocurren en el lado del cliente, utilizando principalmente tecnologías como HTML, CSS y JavaScrip

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

**Static Pages**

**Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente**

El server-side rendering permite que las páginas se carguen de forma excelente,  
aunque requiere una alta utilización en el servidor web.  
El client-side rendering funciona en sentido contrario y alivia al servidor, pues  
renderiza una gran parte de la página en el navegador (para ello, el usuario no  
debe haber bloqueado JavaScript).  
La generación de sitios estáticos reduce la carga tanto al servidor como al cliente  
y, gracias al enfoque de prerrenderización, permite mostrar rápidamente el  
contenido, siempre que no sea interactivo ni cambie constantemente

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

**Ingresando al mundo HTML**

**Hyper Text Markup Language**, más conocido mediante sus siglas HTML. Al final es de suma importancia el lenguaje porque es el medio con el cual se suministra el contenido a los navegadores. HTML es el lenguaje con el que se escribe el contenido de las páginas web. Las páginas web pueden ser vistas por el usuario mediante un tipo de aplicación llamada cliente web o más comúnmente "navegador".

**Un poco de historia.**

El lenguaje HTML se creó en 1991. Su padre es **Tim Berners-Lee** que lo diseñó con objetivos divulgativos. Inicialmente no se pensó que la web llegaría a ser un área de ocio con carácter multimedia.

Sin embargo, esas reglas de representación no son subjetivas de cada fabricante del navegador, sino que **existe una organización llamada W3C que se encarga de definir el estándar que todos deben seguir a la hora de escribir e interpretar el HTM**L. Estos estándares del HTML se conocen como "**Especificaciones**", las cuales han ido apareciendo en el tiempo**. El HTML5 es el último estándar en la actualidad.** Es la base fundamental de la WWW.

**Los navegadores y sus problemas**

**El conflicto generado por los navegadores es debido a su diversidad**. Existen multitud de navegadores o clientes web presentes en el mercado los cuales muchas veces no son capaces de interpretar un mismo código de una manera unificada.

Afortunadamente, **en la actualidad las diferencias de interpretación de los navegadores con respecto a un mismo código HTML son mínimos.**

Los navegadores son los **responsables de introducir nuevas etiquetas** en el uso común del día a día, que se han ido incorporando al estándar HTML en sucesivas versiones. Aunque antes de estandarizarse esas **etiquetas era común que cada navegador crease su etiqueta propietaria para resolver la misma necesidad.**

**Los lenguajes de la web**

* HTML
* CSS
* JAVA
* PYTHON
* PHP
* GO
* JS
* Etc.

**Como funciona un navegador web**

Primero y principal hace falta tener un navegador web

**Los editores de HTML**

Además del navegador necesario para ver los resultados de nuestro trabajo, necesitamos evidentemente otra herramienta capaz de crear la página en sí.

Un archivo HTML (una página) no es más que un texto plano (sin forma estética) al que le colocamos extensión ".html". **Es por ello que para programar en HTML necesitamos un editor de texto.**

No obstante, es aconsejable en un principio utilizar una herramienta lo más sencilla posible para poder prestar la máxima atención a nuestro código y familiarizarnos lo antes posible con él.

**Igualmente para quienes están comenzando nosotros recomendamos:**

* Notepad++
* Geany
* Brackets

En resumen, HTML sirve para decir qué contenido debe tener una página y CSS sirve para decir cómo se debe representar tal contenido. Como has visto, **una página es un archivo donde está contenido el código HTML en forma de texto**. Estos archivos tienen extensión .html o .htm (es indiferente cuál utilizar). De modo que cuando programemos en HTML lo haremos con un editor de textos y guardaremos nuestros trabajos con extensión .html, por ejemplo mipágina.html

**HTML y su significado**

HTML (HyperText Markup Language) is the markup language used to turn text documents into web pages and applications. The fundamental purpose of HTML as a markup language is to provide a semantic description (the meaning) of the content. Las paginas WEB se pueden escribir tanto en syntaxis HTML como XML. HTML5 documents can be written in XHTML syntax (formally known as the “XML Serialization of HTML5”). HTML5 documents written in XML syntax do not require a DOCTYPE but may include an XML declaration.

**HTML5 y su lanzamiento y sus nuevas adquisiciones**

the HTML5 Candidate Recommendation maintained by the World Wide Web Consortium (W3C) dated December 17, 2012, the HTML 5.1 Editor’s Draft dated June 15, 2013, and the “living” HTML specification maintained by the Web Hypertext Application Technology Working Group (WHATWG) as of June 15, 2013.

HTML5 offers new features (elements, attributes, event handlers, and APIs, new inputs types) for easier web application development and more sophisticated form handling. The HTML5 specification is based on HTML 4.01 Strict. Instead, it uses the Document Object Model (DOM, the “tree” formed by a document’s structure) as its basis.

**HTML Iniciando con la sintaxis**

El HTML es un "lenguaje de marcado". **Basa su sintaxis en un elemento base al que llamamos marca, tag o simplemente etiqueta**. A través de las etiquetas vamos definiendo los elementos del documento. **Un documento HTML estará constituido por texto y un conjunto de etiquetas** para definir la función que juega cada contenido dentro de la página. Todo eso servirá al navegador para saber cómo se tendrá que presentar el texto y otros elementos en la página.

**Anatomía de una etiqueta HTML**

Una etiqueta HTML es una instrucción que se utiliza para marcar o etiquetar partes especificas del contenido de una página web. La etiqueta presenta frecuentemente dos partes, su apertura y cierre, y se encierran ambas partes entre

símbolos "menor que" y "mayor que". Lo veremos a continuación.

**Apertura**

El inicio de una etiqueta se produce de la siguiente manera:

<etiqueta>

**Cierre**

El final de una etiqueta se produce de manera similar a su apertura, aunque agregando una barra:

</etiqueta>

Todos los atributos de una etiqueta, siempre se encuentran en la etiqueta de apertura.

**Atributos**

Los atributos son propiedades que se utilizan para proporcionar informacion adicional sobre una etqueta HTML. Se especifica dentro de la etiqueta de apertura y pueden influir en como se muestra o comporta el elemento. Están compuestos por un nombre y un valor -> nombre = “valor”

**Browser support**

Most of the new HTML5 semantic elements and attributes are supported by contemporary desktop and mobile browsers (Chrome, Safari, Opera, Firefox, and Internet Explorer 9+). For browsers that do not support new elements and APIs, there is usually a JavaScript polyfill (patch) that simulates support.

**Validating HTML5 Documents**

You can check to see if your HTML5 document is valid using the online validator at **validator.w3.org** (HTML5 support is in beta as of this writing) or **html5.validator.nu**.

**Partes de un documento HTML**

Un documento HTML ha de estar delimitado por la etiqueta HTML.

Podemos asimismo distinguir dos partes principales:

* **La cabecera, delimitada por la etiqueta HEAD**, donde colocaremos etiquetas de índole informativo. El contenido de la cabecera no suele aparecer en el cuerpo de la página, pero sirve a los navegadores y otros sistemas para encontrar información útil para entender y procesar el documento. Contains information about the document, a collection of metadata. Every head element must include a title element that provides a description of the document. The head element may also include any of these elements in any order: base, link, meta, noscript, script, and style. Todas las etiquetas dentro del head habla del contetxo de la pagina web
* **El cuerpo, flanqueado por la etiqueta BODY**, que será donde colocaremos nuestro texto e imágenes delimitados a su vez por otras etiquetas como las que hemos visto. There may only be one body element in a document. In HTML documents, it is optional; in XHTML, it is required.

El resultado de un documento básico tiene la siguiente estructura:

<html>

<head>

……………….

</head>

<body>

…………………

</body>

</html>

**Las mayúsculas o minúsculas son indiferentes al escribir etiquetas.** Quiere decir que las etiquetas pueden ser escritas con cualquier tipo de combinación de mayúsculas y minúsculas.

**Saltos de Línea en HTML 🡪** <br>

BR no muestra algo en pantalla, sino que directamente muestra el texto en el siguiente renglón.

**Tu Primera Pagina HTML**

**Doctype**

Reconozco que el "doctype" no es la etiqueta más intuitiva, pero debemos mencionarla ahora porque es el inicio de cualquier archivo HTML. Indica que el type del documento es HTML

<!DOCTYPE html>

El Doctype o “Declaración del tipo de documento” es una instrucción especial que va

al inicio de nuestro documento HTML y que **permite al navegador entender qué**

**versión de HTML estamos utilizando.** Esta información determinará la manera en la que el navegador procesará el documento.

**Juego de caracteres**

Mientras que unos sistemas como Linux o Mac usan por defecto un juego de caracteres llamado UTF-8, en Windows se usa de manera predeterminada otro juego de caracteres llamado ISO-8859-1. **En HTML5 el juego de caracteres a usar es siempre UTF-8**. Por lo que tendremos que tener especial atención si somos usuarios de Windows, para asegurarnos que usamos la codificación correcta.

**Para definir qué juego de caracteres** se tiene que escribir una etiqueta en la cabecera de la página, en el HEAD, llamada **META**. Realmente las etiquetas META.

<meta charset="UTF-8">

**Tipos de Etiquetas - Bloque vs Linea**

Existen dos tpos de etiquetas que tienen diferente comportamiento de presentación en la pagina y estas se categorizan en dos grupos:

* Bloque (block)
* En línea (inline)

Los elementos de bloque ocupan todo el ancho de la pagina y se colocan uno debajo del otro, creando un nuevo bloque en la pagina. Algunos ejemplos son: <div> <p> <h1> … <h6>.

Los elementos en línea ocupan solo el espacio necesario y no crean un nuevo bloque. Se colocan uno al lado del otro siempre que haya suficientemente espacio horizontal. Como por ejemplo: <span> <a> <strong>

**Etiquetas para la base**

Definen la estructura fundamental de una pagina web. Establece metadatos, vínculos a hojas de estilo y recursos. Configuran detalles como el titulo y la codificación de caracteres. Todas estas etiquetas van dentro de <head>.

Tabla

Descripción generada automáticamente

**Etiquetas para texto**

Permiten dar formato al texto, crear vínculos y agregar elementos de efansis. Incluyen encabezados, negritas, cursivas y otros estilos de texto. Facilitan la creación de citas y saltos de línea en el contenido.

Tabla

Descripción generada automáticamente

**Etiquetas mutlimedia**

Hacen posible la inclusión de imágenes, videos, audio y contenido embebido.

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

**Etiquetas para estructurar**

Son para organizar y facilitar la estructura de la página en secciones lógicas. Ayudan a definir áreas como encabezado, pie de página, contenido principal y secciones independientes. Permiten agrupar elementos

Tabla

Descripción generada automáticamente

Tabla

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Formatos de párrafos en HTML**

Dar formato a un texto pasa por tareas tan evidentes como definir los párrafos, justificarlos, introducir viñetas, numeraciones o bien poner en negrita, itálica etc.

**Hemos visto que para definir los párrafos nos servimos de la etiqueta P** que introduce un salto y deja una línea en blanco antes de continuar con el resto del documento.

**Los párrafos delimitados por etiquetas P pueden ser fácilmente alineados** a la izquierda, centro, derecha o **justificados** especificando dicha alineación en el interior de la etiqueta **por medio de un atributo "align**".

**Un atributo, es entonces, un parámetro incluido en el interior de la etiqueta** que ayuda a definir el funcionamiento de la etiqueta de una forma más personalizada.

Así, si deseásemos introducir un texto **alineado a la izquierda escribiríamos**:

<p align="left">Texto alineado a la izquierda</p>

**Para una justificación al centro**:

<p align="center">Texto alineado al centro</p>

**Para alinear a la derecha:**

<p align="right">Texto alineado a la derecha</p>

Una forma de simplificar nuestro código y de evitar introducir continuamente el atributo align sobre cada una de nuestras etiquetas es utilizando la etiqueta DIV. Como hemos visto, **la etiqueta DIV marca divisiones en las que definimos un bloque de contenido.**

Así, el código:

<p align="left">Parrafo1</p>

<p align="left"> Parrafo3</p>

<p align="left"> Parrafo2</p>

es equivalente a:

<div align="left">

<p>Parrafo1</p>

<p>Parrafo2</p>

<p>Parrafo3</p>

</div>

**Encabezados/Títulos**

**Hay varios tipos de encabezados, que se diferencian visualmente en el tamaño de la letra que utilizan**. La etiqueta en concreto es la H1, para los encabezados más grandes, H2 para los de segundo nivel y así hasta H6 que es el encabezado más pequeño. Pero lo importante, insistimos es la estructura que denotan. **Una página tendrá generalmente un encabezado de nivel 1** y dentro varios de nivel 2. Luego, dentro de los H2 encontraremos si acaso H3, etc.

HTML syntax requires that headings appear in order (for example, an h2 should not precede an h1) for proper document structure. Doing so not only improves accessibility, but aids in search engine optimization. In HTML5, heading order can be repeated within sections of the same document, allowing greater flexibility with heading levels

<h1>Encabezado de nivel 1</h1>

<h2>Encabezado de nivel 2</h2>

<h3>Encabezado de nivel 3</h3>

<h4>Encabezado de nivel 4</h4>

<h5>Encabezado de nivel 5</h5>

<h6>Encabezado de nivel 6</h6>

1. **Unidad 2**

**Contenidos**

Hipervínculos - Listas - Tablas - Iframe - Listas (ul – lo) - Tablas (table - tr - td - th) - Iframe (incrustar Youtube o Google Maps) - Imágenes y Formularios Imágenes - image src. Formularios: Form - input (text - email - password - submit - check - radio - textarea - button - label - Select / option )



**Organización del Sitio Web**

Desarrollo Local 🡪 Informacion dividida en carpetas

* Index.html 🡪 Pagina Prinicpal Ya que el navegador Busca un index.html
* Carpeta CSS
* Carpeta JS
* Carpeta Img
* Archivos.html

**Formateando el texto en HTML**

Pero antes de comenzar cabe hacer una reflexión sobre por qué son interesantes estas etiquetas y se siguen usando, ya que en la práctica están directamente formateando el aspecto de las fuentes. **Son importantes porque las etiquetas** **en si no están para definir un estilo en concreto, sino una función de ciertas palabras dentro de un contenido**

* **Negrita**

Podemos escribir texto en negrita incluyéndolo dentro de las etiquetas B y su cierre (bold). Esta misma tarea es desempeñada por STRONG y su cierre, siendo ambas equivalentes.

<b> Texto en Negrita <b>

**¿Qué diferencia hay entre B y STRONG?**

Aunque las dos etiquetas hacen el mismo efecto. La etiqueta B indica negrita, mientras que la etiqueta STRONG indica que se debe escribir con fuerza (tiene mayor peso semanticamente)

**Mientras que B significa simplemente negrita** y todos los navegadores la interpretarán como negrita, **STRONG es una etiqueta que significa que se tiene que resaltar fuertemente el texto** y cada navegador es el responsable de resaltarlo como desee.

* **Em**

Etiqueta en línea que indica que su contenido debe mostrarse en cursiva

* **Itálica**

También en este caso existen dos posibilidades, una corta: I y su cierre (italic) y otra un poco más larga: EM y su cierre.

<i>Texto en itálica</i>

* **Subrayado**

El HTML nos propone también para el subrayado el par de etiquetas: U (underlined). Además, **cabe decir que la etiqueta U se ha quedado obsoleta**, debido a que es algo que realmente se debe hacer del lado del CSS, al ser básicamente un estilo.

* **Subíndices y supraíndices**

Este tipo de formato resulta de **extremada utilidad para textos científicos**. Las etiquetas empleadas

son:

<sup> y </sup> para los supraíndices

<sub> y </sub> para los subíndices

**Anidar etiquetas**

**Todas estas etiquetas y por supuesto el resto** **pueden ser anidadas unas dentro de otras** de manera a conseguir resultados diferentes. Así, podemos sin ningúnproblema crear texto en negrita e itálica embebiendo una etiqueta dentro de la otra:

<b>Esto sólo está en negrita <i>y esto en negrita e itálica</i></b>

**TIP:** Antes de cerrar la etiqueta principal cierres las etiquetas que hayas abierto dentro de ella

**Los colores y HTML**

Ahora aprenderás a crear colores en notación RGB con valores en hexadecimal en base 16. **En HTML se usa una notación específica** de especificar un color, **compuesta por tres valores "RGB"**: Red, Green, Blue. Es decir, que para conseguir un color cualquiera mezclaremos cantidades de cada uno de esos colores.

**Para conseguir un color**, mezclaremos valores asignando **dos dígitos a cada valor RBG**. De esta manera: **"#RRGGBB”.**

**Actualmente hay otras formas de escribir colores en las que profundizaremos en CSS**. Muchas veces se utilizan diferentes estilos de escritura para definir colores, porque alguna notación es más versátil.

**Atributos de color en etiquetas HTML**

En HTML existen numerosas etiquetas que soportan atributos de color.

así se cambiaría la fuente para escribir en rojo:

<font color="#FF0000">Rojo</font>

**De nuevo tenemos que advertir sobre la necesidad de expresar todo lo que son estilos mediante CSS. En HTML nos debemos centrar en lo que es escribir el contenido y en CSS en aplicar el estilo.**

Para obtener una rueda y los códigos de colores te recomendamos que ingreses en el sitio: <https://htmlcolorcodes.com/es/>

**Colores Seguros 🡪** corresponden a colores compatibles con todos los sistemas. **Hoy la necesidad de usar colores seguros no es tan grande** como hace años porque es raro encontrar un monitor que solo soporte 256 colores.

Utilizar estos valores: **00-33-66-99-AA-CC-FF**

**Atributos para páginas**

Explicamos una serie de atributos que se aplican de manera global a toda la página. En este artículo nos metemos de nuevo en el terreno del CSS. Las páginas HTML pueden construirse con variedad de atributos que le pueden dar un aspecto a la página muy personalizado. Podemos definir atributos como el color de fondo, el color del texto o de los enlaces. **Estos atributos se definen en la etiqueta BODY y, como decíamos son generales a toda la página.**

* + - * **Atributos para fondos**
* **gcolor: especificamos un color de fondo para la página**. El color de fondo que podemos asignar con bgcolor es un color plano, es decir el mismo para toda la superficie del navegador.
* **background: sirve para indicar la colocación de una imagen como fondo de la página.**

<body background="fondo.jpg">

**Recomendación:** Siempre que coloques una imagen de fondo, debemos poner también un color de

fondo cercano al color de la imagen.

* **Color del texto**
* **text: este atributo sirve para asignar el color del texto de la página**. Por defecto es el negro.

Además del color del texto, tenemos tres atributos para asignar el color de los enlaces de la página.

Para ello suelen aparecer subrayados y con un color más

vivo que el texto. Los tres atributos son los siguientes:

* **link: el color de los enlaces que no han sido visitados**. (por defecto es azul clarito)
* **vlink: el color de los enlaces visitandos. La "v" viene justamente de la palabra visitado**. Es el color que tendrán los enlaces que ya hemos visitado.
* **alink: es el color de los enlaces activos. Un enlace está activo en el preciso instante que se pulsa**.
* **Márgenes**

Con otros atributos de la etiqueta BODY se pueden asignan espacios de margen en las páginas, lo

que es muy útil para eliminar los márgenes en blanco que aparecen a los lados, arriba y debajo de la

página. **Estos atributos son distintos para Internet Explorer y para otros navegadores, por lo que**

**debemos utilizarlos todos si queremos que todos los clientes web los interpreten perfectamente.**

* **leftmargin:** para indicar el margen a los lados de la página. Válido para iexplorer.
* **topmargin**: para indicar el margen arriba y debajo de la página. Para iexplorer.
* **marginwidth:** la contrapartida de leftmargin para Firefox. (Margen a los lados)
* **marginheight:** igual que topmargin, pero para Firefox. (Margen arriba y abajo)

**Por qué todos estos estilos deberían definirse en CSS ?**

Como hemos dicho, todos estos estilos deberían indicarse en el CSS. Además es un tipo de codificación que entre otras cosas, permite tener los estilos definidos en un único lugar, un archivo con código en texto plano, y todas las páginas de tu sitio web usarían ese mismo archivo para definir su presentación.

**Enlaces en HTML y sus distintos tipos**

**Un sitio web podría ser considerado como un conjunto de archivos**, basado en páginas HTML e imágenes, que constituyen el contenido al que el usuario tiene acceso.

No podríamos hablar de navegación si estos archivos HTML no estuviesen debidamente conectados entre ellos y el exterior de nuestro sitio por medio de enlaces url´s.

**Sintaxis de un enlace**

Para colocar un enlace, **nos serviremos de las etiquetas A y su cierre**. Dentro de la etiqueta de

apertura deberemos especificar asimismo el destino del enlace. Este destino será introducido bajo

forma de atributo, el cual lleva por nombre "href".

Existen 3 tipos de enlaces”

* **Absoluto 🡪** es un enlace que incluye todas las partes de una URL.
* **Relativo** 🡪 hace referencia a un recurso que se encuentra en una posición relativa a nuestra URL.
* **Ancla (o anchor):** Se utiliza para indicar un elemento dentro de la misma página que

estamos viendo. Permite ir a un sector específico dentro de la misma página. Para utilizarlo necesitamos el enlace propiamente dicho y el sector al cual debemos dirigirlo (ancla).

**<a href="destino">hacé click aquí</a>**

Siendo el "hacé click aquí" un texto o una imagen.

**Si queremos que el contenido del enlace sea una imagen**, podremos colocar la

etiqueta IMG dentro de la etiqueta A.

**<a href="https://www.google.com/"><img src="logo.jpg"></a>**

**Tipos de enlaces**

En función del destino los enlaces pueden ser agrupados en:

* **Enlaces internos**: los que se dirigen a otras partes dentro de la misma página.
* **Enlaces locales**: los que se dirigen a otras páginas del mismo sitio web.
* **Enlaces remotos**: los dirigidos hacia páginas de otros sitios web.
* **Enlaces emails**: para crear un mensaje de correo dirigido a una dirección.
* **Enlaces con archivos**: para que los usuarios puedan hacer download de ficheros.
* **Sintaxis de los enlaces en la misma página (ANCLAS)**

Crear un enlace que apunte al final de la página. Lo primero será colocar nuestro enlace origen.

**<**a href="**#abajo**">Ir abajo</a>

**#abajo**, es un punto de la misma página que todavía no hemos definido. **Ojo al símbolo "#":** **es él quien especifica al navegador que el enlace apunta a una sección en particular de la misma página.**

En segundo lugar, **hay que generar un enlace en el destino, al que hemos llamado "ancla".** Este enlace no llevará contenido ya que nos sirva de ancla. Tampoco llevará el atributo "href", porque le apuntarán a él. **Para poder distinguirlo a cada ancla se le asigna un nombre por medio del atributo "name".**

<a **name**="abajo"></a>

* **Cómo construir enlaces en HTML cuyo destino sean otras páginas dentro del mismo sitio web**

Un sitio web está constituido de páginas interconexas, que se relacionan mediante enlaces de hipertexto.

**Enlaces locales**

Estos enlaces locales **nos permiten relacionar distintos documentos HTML que componen un sitio web**. Gracias a los enlaces locales podremos convertir varias páginas sueltas en un sitio web completo, compuesto de varios documentos.

**Para crear este tipo de enlaces,** hemos de usar la misma etiqueta A que ya conocemos, de la

siguiente forma:

<a href="**archivo.html**">contenido</a>

**Rutas de los enlaces**

Como rutas nos referimos al destino del enlace, o sea, lo que ponemos en el atributo "href".

Por regla general, para una mejor organización, **los sitios suelen estar ordenados por directorios**.

Estos directorios suelen contener diferentes secciones de la página:

* Imágenes
* Scripts
* Estilos.
* Etc.

Tendremos que especificar además el directorio en el que nuestro archivo.html está alojado.

**Si la página de destino está en una carpeta o subdirectorio interior** al directorio donde está el archivo de origen 🡪 Carpeta/archivo\_destino.extension

**Si la página destino se encuentra en un directorio padre** (superior al de la página del enlace), hemos de escribir dos puntos y una barra "../" tantas veces como niveles subamos de carpetas hasta dar con el directorio donde esta emplazado el archivo destino.

* **Enlazar con una página diferente, pero en una sección interna**

Los enlaces locales pueden, a su vez, apuntar a la página y a una sección concreta. Este tipo de

enlaces resultan ser un híbrido de interno y local. La sintaxis es de este tipo:

<a href="archivo.html#seccion">contenido</a>

* **Enlaces remotos**

Son los enlaces que se dirigen hacia páginas que se encuentran fuera de nuestro sitio web. Los enlaces remotos son los que van a páginas que estarían en otro dominio diferente**. Simplemente colocamos en**

**el atributo HREF de nuestra etiqueta A la URL o dirección de la otra página**

<a href="http://www.google.com">ir a google.com</a>

* **Enlaces a direcciones de correo**

Los enlaces a direcciones de correo son aquellos que al seleccionarlos nos abre un nuevo mensaje

de correo electrónico dirigido a una dirección de mail determinada.

Para colocar un enlace dirigido hacia una dirección de correo **colocamos mailto: en el atributo href**

del enlace, seguido de la dirección de correo a la que se debe dirigir el enlace.

<a href="**mailto**:juan@gmail.com">juan@gmail.com</a>

**Importante: Si un usuario no tiene configurado un programa de correo por defecto en su ordenador no podrá enviar mensajes con esta metodología.**

**También podemos colocar en el enlace el asunto del mensaje**. Esto se consigue **colocando** después de la dirección de correo un **signo de interrogante** **(?),** la palabra **subject**, un signo igual (=) y el asunto en concreto.

<a href="malito:juan@gmail.com**?subject=contacto por email**">[juan@gmail.com</a](mailto:juan@gmail.com%3c/a)>

**En este caso indicamos también que el correo debe ir con copia** a [roberto@gmail.com](mailto:roberto@gmail.com). (&cc=mail)

<a href="mailto:juan@gmail.com?subject=contacto por

[mail**&cc=**roberto@gmail.com">juan@gmail.com</a](mailto:mail&cc=roberto@gmail.com%22%3ejuan@gmail.com%3c/a)>

**Formatos gráficos para páginas web**

**Tipos de archivos**

En Internet se **utilizan principalmente los tipos de archivos gráficos PNG, GIF y JPG**, para optimizar el tamaño que ocupan en disco, ya que se transmiten más rápidamente por la Red. El formato de archivo **GIF se usa para las imágenes que tengan dibujos**, mientras que el formato **JPG se usa para las fotografías.** Actualmente el uso de formatos SVG esta aumentando.

* **Formato GIF**

**Compresión**: Es muy buena para dibujo. Incluso puede ser interesante si laimagen es muy pequeña, aunque sea una foto.

**Transparencia**: es una utilidad para definir ciertas partes del dibujo como transparentes. De este modo podemos colocar las imágenes sobre distintos fondos sin que se vea el cuadrado donde está inscrito el dibujo.

**Colores**: Con este formato gráfico podemos utilizar paletas, conjuntos, de 256 colores o menos. Cuantos menos colores, menos espacio ocupa.

* **Formato JPG**

**Compresión**: Cuenta con un algoritmo de compresión casi ideal para fotografías. Además, con JPG podemos definir la calidad de la imagen, con calidad baja el fichero ocupará menos, y viceversa.

**Colores**: JPG trabaja siempre con 16 millones de colores, ideal para fotografías.

**Optimizar ficheros**:

Para que las imágenes ocupen lo menos posible y se transfieran rápidamente por la Red debemos

aprender a optimizar los ficheros gráficos. Para ello debemos hacer lo siguiente:

* **Para los archivos GIF y PNG:**

Reduciremos el número de colores de nuestra paleta. Esto se hace con nuestro editor gráfico, en

muchos casos podremos hacerlo al guardar el archivo.

* **Para los archivos JPG:**

Ajustaremos la calidad del archivo cuando lo estemos guardando. Este formato nos permite bajar

mucho la calidad de la imagen sin que esta pierda mucho en su aspecto visual.

* **Formato PNG**

(Portable Network Graphics) es un **formato de archivos de gráficos de mapa de bits** (una trama). Fue desarrollado en 1995 como una alternativa gratuita al formato GIF.

**Características del formato PNG**

El formato PNG **permite almacenar imágenes en blanco y negro** (una profundidad de color de 16 bits por píxel**) y en color real** (una profundidad de color de 48 bits por píxel), así como también imágenes indexadas, utilizando una paleta de 256 colores y soporta la posibilidad de definir 256 niveles de transparencia (Canal Alfa).

* **Formato SVG**

Scalable Vector Graphics (Gráficos Vectoriales Escalables). Es un n**uevo estándar usado para la creación y representación de gráficos e imágenes vectoriales** en las páginas web y aplicaciones de internet.

Las imágenes en **este formato pueden editar usando editores de texto plano** como NotePad+ o hasta con el Bloc de Notas de Windows. El formato SVG es recomendado por el W3C y **es compatible por la mayoría de los navegadores web modernos**. Las imágenes SVG se caracterizan por su pequeño tamaño.

**Atributos básicos para imágenes en HTML**

La etiqueta que utilizaremos para insertar una imagen es IMG (image). Esta etiqueta no posee su

Cierre y en ella hemos de especificar obligatoriamente el paradero de nuestro archivo gráfico mediante el atributo src (source).

<img src="logo.png">

* **Atributo alt**

Dentro de las comillas de este atributo colocaremos una brevísima descripción de la imagen.

<img src="logo.png" **alt="Descripción del logo o imagen**">

Se utiliza par:

* + posicionamiento en buscadores
  + Personas con dificultades visuales
  + Cuando la imagen no se encuentra disponible
* **Atributos height y width**

Estos atributos definen la altura y anchura respectivamente de la imagen en píxeles. Aunque estas

dimensiones forman parte del estilo de la imagen, y por tanto **podrían ir en el CSS.**

<img src="logo.gif" **width="200" height="300**">

* **Atributo border**

Podemos indicar que una imagen tenga borde. Mediante el atributo "border" se define el tamaño en píxeles del cuadro que rodea la imagen. De esta forma podemos recuadrar nuestra imagen si lo deseamos.

* **Atributos vspace y hspace**

Sirven para **indicar el espacio libre, en pixeles, que tiene que colocarse entre la imagen y los otros**

**elementos** que la rodean, como texto, otras imágenes, etc. Estos atributos forman parte también de

la responsabilidad de las CSS, así que **no sería recomendable usarlos**.

**FavIcon**

Un favicon es la **pequeña imagen que se muestra en la pestaña del navegador** o en la lista de marcadores (favoritos). **El tamañ**o en la barra de direcciones es de 16x16 píxeles.

Va siempre dentro **dentro del Head.**

**Conversor a .ico 🡪** https://convertico.com/ y

<link rel=’icon’ type=’extension’ href=’icon\_source >

**En caso de que la imagen no sea .ico,** se puede aclarar el tipo de formato de la imagen en el atributo ‘type’.

<link rel="icon" type="image/png" href="" />

**Listas en HTML**

Podemos distinguir tres tipos de listas HTML:

* Listas desordenadas
* Listas ordenadas
* Listas de definición

Ejemplo 🡪 https://jsbin.com/birojac/edit?html,output.

**Listas desordenadas**

Son delimitadas por las etiquetas UL y su cierre (unordered list). Cada uno de los elementos de la

lista es citado por medio de una etiqueta LI.

<p>Lenguajes de programación</p>

<ul>

<li>PHP</li>

<li>JAVA</li>

<li>Python</li>

</ul>

**Podemos definir el tipo de viñeta** empleada para cada elemento por medio del atributo type incluido dentro de la etiqueta de apertura UL aunque podemos definirlo por CSS.

<ul **type**="tipo de viñeta">

donde tipo de viñeta puede ser uno de los siguientes: **circle** – **disc** - **square**

**Listas ordenadas**

Para realizar las listas ordenadas usaremos las etiquetas OL (ordered list) y su cierre. Cada elemento

sera igualmente indicado por la etiqueta LI, que ya vimos en las listas desordenadas.

Pongamos un ejemplo:

<p>Reglas de convivencia</p>

<ol>

<li>No hacer ruidos molestos

<li>Ser amable con los vecinos

</ol>

En concreto nos es posible especificar el tipo de numeración empleado eligiendo entre números letras minúsculas y sus mayúsculas y números romanos en sus versiones mayúsculas (I, II, III,...) y minúsculas (i, ii, iii,...).

**El atributo type,** el cual será situado dentro de la etiqueta OL podemos **definir el tipo de numeración** y **con el atributo start desde qué valor comenzará.** Los valores que puede tomar el atributo en este caso son:

<ol> números (1, 2, 3...)

<ol type="A"> mayúsculas (A, B, C,...)

<ol type="a"> minúsculas (a, b, c...)

<ol type="I"> mayúsculas (I, II, III,...)

<ol type="i"> minúsculas (i, ii, iii,...)

**Listas de definición**

Las listas de definición sirven para hacer un conjunto de elementos con pares concepto-descripción. Para realizar una lista de definición, **la etiqueta principal es DL** y su cierre (definition list). Las etiquetas **del elemento y su definición son DT** (definition term) y **DD (definition definition) respectivamente**.

Alimentos Lácteos

<p>Alimentos Lácteos</p>

<dl>

<dt>Leche

<dd>descremada

<dd>entera

<dt>Queso

<dd>Fontina

<dd>Muzzarella

</dl>

Leche

descremada

entera

Queso

Fontina

Muzzarella

**Se Puede Anidar Listas**

**Tablas en HTML**

Una tabla es un conjunto de celdas dentro de las cuales podemos incorporar contenidos. En general, se utilizan para representar información tabulada, en filas y columnas. Esto es una realidad **en los últimos años, desde que las tablas se han descartado para fines relacionados con la maquetación.** Actualmente toda la maquetación de una página se organiza con CSS, lo que nos da un mayor control de todos los elementos de la página y la posibilidad de separar todos los estilos para definir el aspecto de una web en un fichero aparte del HTML.

Por ello, **en el momento actual las tablas se utilizan mucho menos** que en el pasado y realmente la recomendación es **usarlas solo en los casos en los que necesitemos incluir en una página información tabulada**. En HTML hay que definir una etiqueta para cada parte de la tabla.

**Etiquetas básicas para tablas en HTML**

Para empezar, nada más sencillo que por el principio: las tablas son definidas por las etiquetas

**TABLE** y su cierre.

Las tablas son descritas por líneas de arriba a abajo (y luego por columnas de izquierda a derecha**). Cada una de estas líneas**, llamada fila, es definida por otra etiqueta y su cierre: **TR.**

Dentro de cada línea, habrá diferentes celdas. **Cada una de estas celdas** será definida por otra etiqueta: **TD**. Dentro de ésta y su cierre será donde coloquemos nuestro contenido, el contenido de cada celda.

<table>

<tr>

<td>Celda 1, linea 1</td>

<td> Celda 2, linea 1</td>

</tr>

<tr>

<td> Celda 1, linea 2</td>

<td> Celda 2, linea 2</td>

</tr>

</table>

|  |  |
| --- | --- |
| Celda 1, linea 1 | Celda 2, linea 1 |
| Celda 1, linea 2 | Celda 2, linea 2 |

También es parte de una tabla **la etiqueta TH (Table Header),** que sirve para crear una celda cuyo

contenido posea un título o cabecera de la tabla.

Ejemplo:

<table>

<tbody>

<tr>

<th>Celda 1, linea 1</th>

<th> Celda 2, linea 1</th>

</tr>

<tr>

<td> Celda 1, linea 2</td>

<td> Celda 2, linea 2</td>

</tr>

</tbody>

</table>

Tambien podes dividir la Tabla en 3 partes 🡪 Thead – Tbody - Tfoot

<table>

<thead>

………………………

</tehad>

<tbody>

…………………………

</tbody>

<tfoot>

……………………………..

</tfoot>

</table>

**Atributos para tablas, filas y celdas**

* **cellspacing**: es el espacio entre celdas de la tabla.
* **cellpadding**: es el espacio entre el borde de la celda y su contenido.
* **border**: es el número de píxeles que tendrá el borde de la tabla.
* **Border-color**: es el color a asignar al borde de la tabla

Podemos usar prácticamente cualquier tipo de etiqueta dentro de la etiqueta TD para escribir su contenido.

**Podemos especificar el formato de nuestras celdas a partir de etiquetas introducidas en su interio**r o mediante atributos colocados **dentro de la etiqueta de celda TD** o bien, **dentro de la etiqueta TR**, si deseamos que el atributo sea valido para toda la línea.

**CSS ES LA FORMA IDEAL**

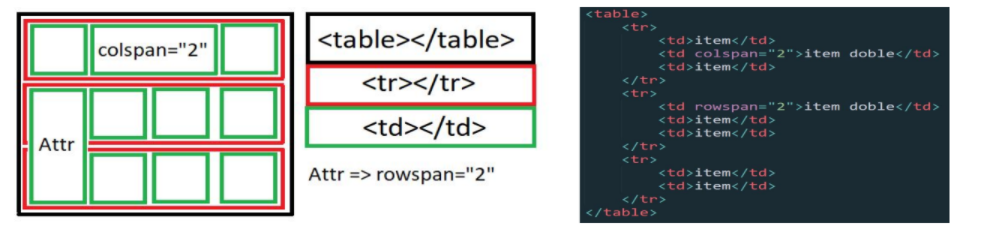
A continuación, algunos atributos útiles para la construcción de nuestras tablas**:**

* **align:** Justifica el texto de la celda del mismo modo que si fuese el de un párrafo**.**
* **valign:** Podemos elegir si queremos que el texto aparezca arriba (top), en el centro (middle) o abajo (bottom) de la celda.
* **bgcolor:** Da color a la celda o línea elegida.
* **Border-color:** Define el color del borde**.**
* **background:** Nos permite colocar un fondo para la celda a partir de un enlace a una imagen.
* **height**: Define la altura de la celda en pixels o porcentaje**.**
* **width:** Define la anchura de la celda en pixels o porcentaje**.**
* **Border-Style**

Si queremos agrupar celdas de una misma celda o columna hay que agregar los siguientes atributos:

colspan o rowspan:

* **colspan:** Expande una celda horizontalmente. Número de celdas a abarcar
* **rowspan:** Expande una celda verticalmente



**Etiqueta Iframe**

En concreto iframe sirve para crear un espacio dentro de la página donde se puede incrustar otra web. Es un cuadrado cuyas dimensiones debe especificar el desarrollador en la propia página, incluidas por los atributos width y height en la propia etiqueta IFRAME.

Podemos agregar: contenidos de terceros, interfaces de usuario, videos de YouTube, mapas de Google Maps y banners de publicidad desde otro sitio.

**El iframe tiene asociada una página web**, que se carga en el espacio y operará de manera totalmente independiente**. Esa página web tendrá sus propios contenidos y estilos**.

**Iframe se utiliza en muchos contextos.** Dentro de un iframe podemos mostrar contenidos de otras

páginas, como si estuvieran en la nuestra, por lo que sirven para ejemplos como**:**

• Visualizar contenidos de terceros, como bloques de noticias o novedades que ofrecen en

otras webs.

• Interfaces de usuario, en el que ciertas actividades se realizan de forma autónoma y el

procesamiento está en otra página web.

• Incrustar videos desde YouTube

• Incrustar mapas de Google Maps

• banners de publicidad desde otro sitio

**Construcción de la etiqueta iframe**

Como decimos, el iframe se coloca directamente en el código HTML, en el lugar donde queremos

que aparezca.

Se coloca con un código como este:

<iframe src="pagina\_fuente.html" width=290 height=250>Texto para cuando el navegador no

conoce la etiqueta iframe</iframe>

Los atributos principales de iframe son la página web que se va a mostrar en el espacio y el ancho y

alto del recuadro que reservemos para el frame flotante. **Es importante destacar que hay que mantener las proporciones** brindadas por la pagina de donde sacamos para colocar el iframe.

**Atributos de iframe**

Atributos disponibles para la etiqueta iframe pero sería más correcto, especificar dentro de las CSS.

* **src**: Para indicar la página web que se mostrará en el espacio del frame flotante.
* **width**: Para definir la anchura del recuadro del iframe
* **height**: Para definir la altura del iframe
* **name**: Para especificar el nombre del frame, que podemos utilizar luego para referirnos a él con el target de los links, o mediante javascript.
* **id**: Para indicar el identificador del iframe, y poder referirnos a él desde javascript.
* **frameborder**: para definir si queremos o no que haya un borde en el frame. Los valores posibles son 0 | 1. frameborder=0 indicaría que no queremos borde y frameborder=1 que sí.
* **scrolling**: indica si se quiere que aparezcan barras de desplazamiento para ver los contenidos del iframe completo, en el caso que no aparezcan en el espacio reservado para el iframe. Los valores posibles son: yes | no | auto.
* **marginwidth**: Para definir el margen a izquierda y derechar que debe tener la página que va dentro del iframe, con respecto al borde.
* **marginheight**: lo mismo que marginwidth, pero en este caso para el tamaño del margen por la parte de arriba y abajo.
* **margin**: para especificar alineación del frame, igual que se especifica para las imágenes.
* **style y class**: los atributos para definir el aspecto del iframe por medio de hojas de estilo css.

**Formularios HTML**

Los formularios son cajas de texto y botones que podemos encontrar en muchas páginas web y se utilizan para realizar búsquedas o bien para introducir datos personales o claves de acceso. Los datos que el usuario introduce en estos campos son enviados al correo electrónico del administrador del formulario o bien a un programa que se encarga de procesarlo automáticamente.

**Cómo hacer un formulario en HTML**

Los formularios son definidos por medio de las etiquetas FORM y su cierre. Entre estas dos

etiquetas colocaremos todos los campos y botones que componen el formulario. Dentro de esta

etiqueta FORM debemos especificar algunos atributos:

* **action**: define el tipo de acción a llevar a cabo con el formulario. Existen dos formas de enviar la

información a que sea procesada:

• El formulario es enviado a una dirección de correo electrónico

• El formulario es enviado a un programa o script que procesa su contenido

En el primer caso, el contenido del formulario es enviado a la dirección de correo electrónico

especificada por medio de una sintaxis de este tipo:

<form action="mailto:direccion@correo.com" ...>Si lo que queremos es que **el formulario sea procesado por un program**a, hemos de especificar la dirección del archivo que contiene dicho programa:

<form action="dirección del archivo" ...>

* **Action:** donde se envia la información ingresada en el formulario
* **method**: Este atributo se encarga de especificar la forma en la que el formulario es enviado. Los dos valores posibles que puede tomar este atributo son **post** y **get**.
* **enctype**: Se utiliza para indicar la forma en la que viajará la información que se mande por el

formulario. En el caso más corriente, enviar el formulario por correo electrónico, el valor de este

atributo debe de ser "text/plain".

* **Target:** Como se abre el archivo

**Elementos de Formularios. Campos de texto**

* **Etiqueta INPUT** para texto corto

Las cajas de texto son colocadas por medio de la etiqueta INPUT. Dentro de esta etiqueta hemos de especificar el **valor de dos atributos: type y name:**

<input type="text" name="nombre">

**El nombre** del elemento del formulario es de gran **importancia para poder identificarlo** en nuestro programa de procesamiento (De lado del Servidor)

Podes agregar que el elemento sea Requerido agregando **required** dentro del tag INPUT

Podes agregar la etiqueta **ID que va a ser útil para identificar al elemento/etiqueta**

* **Se utilizas el <label>** atrás de un Input, hay que utilizar el atributo for=’elemento\_target’

Donde elemento\_target debería ser el ID del input relacionado. Los une

**Existen otra serie de atributos** que pueden resultar de utilidad pero que **no son obligatorios**:

* **size**: define el tamaño de la caja de texto, en número de caracteres visibles. S
* **maxlength**: indica el tamaño máximo del texto, en número de caracteres, que puede ser escrito en el campo. En caso que el campo de texto tenga definido el atributo maxlength, el navegador no permitirá escribir más caracteres en ese campo que los que hayamos indicado.

Nota: Es importante no confundir el atributo maxlength con el atributo size. Mientras **size define el tamaño visible de la caja de texto, maxlength indica el tamaño máximo real del texto que se puede escribir**.

* **value**: en algunos casos puede resultarnos interesante asignar un valor definido al campo en

cuestión. Esto puede ayudar al usuario a rellenar más rápidamente el formulario o darle alguna idea

sobre la naturaleza de datos que se requieren. Este valor inicial del campo puede ser expresado

mediante el atributo value.

Veamos su efecto con un ejemplo sencillo:

<form>

<input type="text" name="nombre" value="Juan Perez">

</form>

Top of Form



Bottom of Form

**Etiqueta TEXTAREA**

Si deseamos poner a la disposición de usuario un campo de texto donde pueda escribir

cómodamente **sobre un espacio compuesto de varias líneas**, hemos de invocar una nueva etiqueta:

TEXTAREA y su cierre correspondiente. Este tipo de campos son prácticos cuando el contenido a enviar existe la posibilidad que el visitante desee rellenar varias líneas.

Dentro de la etiqueta textarea deberemos indicar el atributo name para **asociar el contenido a un nombre**. Además, **podemos definir las dimensiones del campo** a partir de los atributos siguientes:

* **rows**: define el número de líneas del campo de texto.
* **cols**: define el número de columnas del campo de texto

<textarea name="comentario" rows="10" cols="40"></textarea>

**Otros elementos de formulario**

**Listas de opciones**

Las listas de opciones son ese **tipo de menús desplegables** que nos permiten elegir una (o varias) de

las múltiples opciones que nos proponen. Para construirlas emplearemos una **etiqueta SELECT**, con

su respectivo cierre:

<select name="estacion">

<option>Primavera</option>

<option>Verano</option>

<option>Otoño</option>

<option>Invierno</option>

</select>

Esta estructura puede verse modificada a partir de otros dos atributos:

* **size**: Indica el número de valores mostrados a la vez en la lista. **Lo típico es que no se incluya ningún valor en el atributo size**.
* **multiple**: Permite la selección de más varios elementos de la lista. La elección de más de un elemento se hace como con el explorador de Windows, a partir de las teclas ctrl o mayúsculas.

La etiqueta OPTION puede asimismo ser matizada por medio de otros atributos

* **selected**: este atributo no toma ningún valor sino que simplemente indica que la opción que lo presenta esta elegida por defecto.
* **value**: Define el valor de la opción que será enviado al programa si el usuario elige esa opción

**Botones de radio**

La etiqueta empleada en este caso es INPUT en la cual tendremos el atributo type ha de tomar el

valor radio.

<input **type=’radio’** name=’estacion’ value=’1’> Primeravera

<br>

<input **type=’radio’** name=’estacion’ value=’2’> Verano

<br>

<input **type=’radio’** name=’estacion’ value=’3’> Otono

<br>

A cada una de las opciones se le atribuye una etiqueta input dentro de la cual asignamos el mismo nombre (name) y un valor (value) distinto para todas las opciones Si el usuario elige supuestamente Otoño, recibiremos: estacion=3

**Cabe señalar que es posible preseleccionar por defecto una de las opciones**. Esto puede ser

conseguido por medio del atributo checked:

<input type="radio" name="estacion" value="2" checked>Verano

**Cajas de Validacion**

Este tipo de elementos pueden ser activados o desactivados por el visitante por un simple clic sobre

la caja en cuestión

<input type="**checkbox**" name="jamon">Me gusta el jamón.

**La diferencia fundamental es el valor adoptado por el atributo type**.

**Envío, borrado y demás en formularios HTML**

Siguiendo con la explicación de todo lo relativo a formularios que estamos ofreciendo en el Manual

de HTML, ha llegado el momento de explicar **cómo podemos hacer un botón para provocar el envío**

**del formulario, entre otras cosas.**

**Botón de envío de formulario (botón de submit)**

La construcción un botón se realiza con las etiquetas INPUT ya vistas:

<input type="submit" value="Enviar">

**Botón de borrado (botón de reset)**

Este botón nos permitirá borrar el formulario por completo

<input type="**reset**" value="Borrar">

Hay que tener cuidado de no ponerlo muy cerca del botón de envío y de distinguir claramente el uno del otro.

**Datos ocultos (campos hidden)**

puede resultar práctico enviar datos definidos por nosotros mismos que ayuden al programa en su procesamiento del formulario. Este tipo de datos, que no se muestran en la página pero si pueden ser detectados solicitando el código fuente.

<input type=**hidden** name="clave" value="1234">

**Botones normales**

Dentro de los formularios también podemos colocar botones normales, Igual que ocurre con los campos hidden, estos botones por si solos no tienen mucha utilidad, pero podremos necesitarlos para realizar acciones en el futuro

<input type=**button** value="Texto escrito en el botón">

**HTML Define todos sus elementos en dos grupos**

* **Inline** (Comienzan en cualquier parte de la línea, Siempre sobre la misma línea, se van acumulando, de derecha hacia izquierda)
* **Block** ( Ocupan todo una Línea y siempre comienzan una nueva línea)

**Inline 🡪** br, a img, spna, b, strong, mark, sub etc.

**Block** 🡪 div, p, h1..h6, ul, ol, li, table, form, br

**Span Y Div**

* span ( abarcar).

Es un contenedor en línea. Sirve para aplicar estilo al texto, es una etiqueta en linea

agrupar elementos en línea.

**Sus etiquetas son**: <span> y </span> (ambas obligatorias).

**Está definido como**: Elemento especial, y por lo tanto en línea.

**Crea una caja**: En línea.

**Puede contener**: Texto, y/o Elementos en línea.

* div (division) -división.

Sirve para crear secciones o agrupar contenidos. Genera un salto de línea (etiqueta en bloque).

Denotes a generic “division” or container for content within the flow of the document. Elements contained within a div are treated as a semantic group. The div element is typically given meaning with the class, id, title, or lang attributes, which also allow it to be accessible to scripts and selected in stylesheets.

**Sus etiquetas son**: <div> y </div> (ambas obligatorias).

**Está definido como**: Elemento en bloque.

**Crea una caja**: En bloque.

**Puede contener**: Texto, y/o cero o más elementos en bloque o en línea.

**Atributos Globales HTML y CSS**

Los atributos globales lo puedes utilizar en todos los elementos HTML5. Los atributos globales pueden especificarse en todos los elementos HTML, e incluso en aquellos no especificados en el estándar. **Esto significa que cualquier elemento no estándar también debe permitir estos atributos**, aun cuando el uso de tales elementos significa que el documento ya no cumple con HTML5.

* **style**="estilo CSS" Especifica un estilo CSS conforme al elemento.
* **class**="texto" Especifica uno o más nombres de clases para un elemento (haciendo referencia a una clase en una hoja de estilo). **Las clases permiten a CSS y JavaScript seleccionar y acceder a elementos específicos a través de selectores de clase o funciones**, como el método Document.getElementsByClassName()
* **id**="texto" Especifica un id único por cada página.
* **title**="texto" Especifica información extra sobre un elemento (Tooltip Text)
* **hidden** (hidden="hidden" en XHTML) Evita que el elemento y sus descendientes se muestren en el navegador. Cualquier control de formulario o de script dentro de la sección hidden será ejecutado, aunque no se muestra al usuario.
* **tabindex**="número" Especifica la posición del elemento en el orden de tabulación del documento. Se usa para tabular a través de los links de la página (o campos de un formulario). También indica si el elemento puede obtener el foco del cursor y si debe participar de la navegación secuencial con el teclado. Si se cumplen estas condiciones, también determina en qué posición estará ordenado según la tabulación. Toma valores Diferentes:
  + **un valor negativo** significa que el elemento debe ser focuseable, pero no debe ser alcanzado vía la navegación secuencial del teclado.
  + **Un cero** significa que el elemento puede obtener el foco y alcanzable vía la navegación secuencial del teclado, pero el orden relativo es definido por la convención de la plataforma.
  + **un valor positivo** que significa que puede obtener el foco vía la navegación secuencial del teclado, el orden relativo es definido por el valor del atributo y sigue la secuencia en orden ascendente según el valor de tabindex.

<https://www.w3schools.com/tags/att_global_tabindex.asp>

* **translate**="yes|no" Indica si el texto del contenido del elemento y los valores del atributo debe ser traducidos o no al encontrar el documento. yes es por defecto; no deja el original

<https://www.w3schools.com/tags/att_global_translate.asp>

* **lang** Especifica el idioma del contenido del elemento.

<https://www.w3schools.com/tags/att_global_lang.asp>

* **spellcheck**="true|false" Especifica si se debe corregir o no la gramática y la ortografía del elemento. <https://www.w3schools.com/tags/att_global_spellcheck.asp>
* **draggable**="true|false" Indica si el elemento es arrastrable; se puede mover haciendo click sin soltar, moviéndolo a una nueva posición en la ventana. <https://www.w3schools.com/tags/att_global_draggable.asp>
* **dropzone**="copy|move|link|string:text string type| file:file type" Especifica si cuando los

datos arrastrados son soltados, son copiados, movidos, o linkeados. copy es cuando los datos son copiados; move mueve los datos a una nueva posición; y link crea un link a los datos originales. Si se incluye string:text/plain esto permite utilizar cualquier texto. El atributo file: indica que se puede utilizar un archivo (ej: file:image/png). Se pueden utilizar los dos, por ejemplo; dropzone="copy string:text/plain".

* **Accesskey:** Proporciona y genera un acceso de teclado para el elemento actual. Este atributo consiste de una lista de caracteres, separadas por espacios. El valor del atributo debe constar de un solo carácter imprimible.
* **Contenteditable:** Es un atributo enumerado que indica si el elemento puede ser modificable por el usuario. Si es así, el navegador modifica el elemento para permitir la edición. El atributo debe tener uno de los siguientes valores:
  + **true o un valor vacío**, el cual indica que el elemento debe ser editable.
  + **false**, el cual indica que el elemento no debe ser editable.
* **Data-\* :** Son atributos globales que forman una clase denominados atributos de datos personalizados , y que se caracterizan por permitir intercambiar información entre el HTML en el momento de la carga del archivo HTML.
* **Dir:** Es un atributo enumerado que indica la direccionalidad del texto del elemento. Puede tener los siguientes valor:
  + **Ltr** 🡪left to right y se utiliza para idiomas que se escriben de izquierda a derecha (como el español).
  + **Rtl** 🡪 right to left y se utiliza para idiomas que se escriben de derecha a izquierda (como el árabe);
  + **Auto** 🡪 permite que el navegador decida utilizando un algoritmo básico, que analiza los caracteres dentro del elemento hasta que encuentra un caracter con una direccionalidad fuerte, a continuación, aplica la direccionalidad a todo el elemento.

<https://www.w3schools.com/tags/att_dropzone.asp>

**Mas info** 🡪 <https://www.w3schools.com/tags/ref_standardattributes.asp>

**Etiquetas Semánticas**

Un elemento semántico describe claramente su significado tanto para el navegador como para el desarrollador.

• <**header>** Cabecera visual de la página o de una sección (logotipo, título, etc...). No

confundir con <head>.

• <**nav**> Apartado de navegación (enlaces de secciones, categorías, etc...)

• <**main**> Especifica el contenido principal de un documento

• <**footer**> Pie de página ( del documento completo) o de una sección.

• <**section**> Define una sección en un documento. Represents a section (a thematic grouping of content) of a document or application with its own internal outline and (optionally) a header and footer. The section element is not a generic container; it should be used only if the element’s contents should appear in the document’s outline.

• <**aside**> Agrupación de contenido no relacionado con el tema principal del documento.

• <**article**> Artículo. Parte principal de un escrito (posts, mensaje en foros, comentario...) . Represents a self-contained piece of content, such as a magazine article, blog post, reader comment, or newspaper article that is intended to be independently distributable, reusable, or used in syndication. article elements may be nested, such as for comments associated with a blog post.

• <**address**> Agrupación con la información de contacto del autor del artículo o documento.

• <**details**> Define un detalle adicional

• <**figcaption**> Define un título para un elemento <figure>

• <figure> Especifica contenido autónomo, como ilustraciones, diagramas, fotos, listas de códigos, etc.

• <**mark**> Define el texto marcado / resaltado

• <**summary**> Define un encabezado visible para un elemento <details>

• <**time**> Define una fecha / hora

**Hoisting Gratuito**

Permite probar rápidamente el sitio Web. Almacena la carpeta del sitio para hacer pruebas. Requiere registro.

Para agregar el sitio basta con arrastrar la carpeta que contiene el sitio.

https://app.netlify.com/drop

Al trabajar con clientes nos conviene pagar Hosting.

**Tutoriales sobre cómo subir a Netlify**:

https://youtu.be/-LRlQ\_jaLAU

https://www.youtube.com/watch?v=vywDFg2uIxY

**prbar**

https://wonderful-brown-da3263.netlify.app/

**Hay forma de sincronizar herramientas Git con servidores. Algunos videos útiles:**

* **Subir Página web a hosting gratuito con GitHub:**

<https://www.youtube.com/watch?v=SCxbN-UzKS0>

* **Netlify Tutorial – Deploying from Git**:

<https://www.youtube.com/watch?v=mN9oI98As_4>

* **Hosting Your Website With Github and Netlify:**

<https://www.youtube.com/watch?v=hBQlCtfRmqs>

* **Setup Custom Domain On Netlify**:

<https://www.youtube.com/watch?v=Q9giWrfIJKk>

**Árbol del DOM (Document Object Modelo)**