HTML Basico Cheat Sheet

by Aleordoh (Martin Ordonez) via cheatography.com/146230/cs/31992/

HTML: Estructura

```
<!..HyperText Markup Language-->
<!D OCTYPE html>
<html lang="e s">
<he ad>
        <meta name="v iew por t" conten t="w hid -
t=d evi ce- whidt, inicia l-s cal e=1 "> (permite
armar una pagina reposnive)
        <meta charse t="U TF- 8">
        <ti tle >Hello world< /ti tle>
</h ead>
          <bo dy>


          </b ody>
</h tml>
```

HTML: Etiquetas de Texto

<p>: Parrafo

<h1,h2,h3>: Cabeceras HTML

background= "URL de la imagen"

bgcolor="color de fondo del documento"

text="color para el texto documento."

<b,i,u>: negrita, itálica y subrayado

****añadir fuente y texto dentro de esta etiqueta (1)

(1) La sintaxis para agregar atributos a las etiquetas es
<etiqueta atributo1="valor"> *Texto* **</etiqueta>**.
P/Ej.:
**** *Texto* ****.

HTML: Etiquetas Semanticas

<Header> : Tradicionalmente, aquí van cosas como el logotipo de la web, la barra de navegación, los enlaces a las redes sociales e incluso un pequeño campo de búsquedas rápidas.

HTML: Etiquetas Semanticas (cont)

<nav>: Sirve para generar una barra de navegación, sea la navegación principal o una navegación alternativa. Dentro de esta etiqueta, el árbol de navegación se suele implementar a través de listas desordenadas, elementos de listas y enlaces o hipervínculos. Generalmente va dentro del <header> o del <footer>

<section>: Nos permite definir una sección de contenido. Si quisiéramos crear un breve apartado sobre un producto o servicio, esta etiqueta sería una excelente opción.

<article>: Nos permite definir una pieza de contenido independiente. Es decir, contenido que podría funcionar por sí solo sin necesidad de todo lo que lo rodea: un producto, un servicio, una noticia, etc.

<footer>: Nos sirve para generar el pie de página principal del documento, o el pie de página de una sección de contenido. Tradicionalmente, aquí van cosas como los derechos reservados y algunos enlaces adicionales de la web.

Ejemplo de Estructura

HTML: Rutas, Hipervinculos e Imagenes

Ruta Absoluta: https://www.google.com

Ruta Relativa: ../imagenes/perfil.jpg

Hipervínculo Externo A Google-

Hpv. Local: Inicio

Anclas: Biografía
Contacto
Nuestro equipo

Correo: Enviar mensaje

Telefono: Teléfono

Imágen: <img src="img/fotoPerfil.jpg" alt="Imagen de perfil">
<img width="320" src="img/fotoPerfil.jpg">

(1) Los tributos width y height permiten ajustar el ancho y alto de la imagen.-

HTML Basico Cheat Sheet

by Aleordoh (Martin Ordonez) via cheatography.com/146230/cs/31992/

HTML: Listas

<ol> Lista ordenada (nums y letras)
Atributo: *type*="Numerico, Alfabetico, Romano"/*start="1, 16, 2"
<ol type="A" start="10">....

<ul> Listas desordenadas (puntos)
Atributo: *type*="Disc (•), Circle (○), Square (■), None"
<ul type="disc">....

<dl> lista completa

<li> Elemento lista

<dt> delimitan terminos

<dd> definiciones

CSS: Vinculación con HTML

Vinculación con HTML: escribiendo la Etiqueta **<link>** en el campo **<HEAD>** de nuestro documento HTML.
P/ej.:
<link href="css/estilos.css" rel="stylesheet">
Esquema general de sintaxis de CSS::
selector #id .class :pseudoclase ::pseudoelemento [atributo] {
propiedad: valor;
propiedad: valor;
propiedad: valor;
}

CSS: Colores de Fondo e Imagenes de Fondo

Color de Fondo: Asigna un color de fondo a un elemento.
CSS: p { background-color: red}
CSS: p { background-color: #3459ff }
CSS: p { background-color: rgb(12, 34, 32) }

Imagen de Fondo: Asigna una imagen de fondo al elemento.
CSS: body { background-image: url('../img/bici.jpg') }

Repeticion de Imagen: Controla si se repite la imagen y de qué manera. Recibe los valores repeat, no repeat, repeat-x, repeat-y, round y space.
CSS: body { background-repeat: repeat-x }

CSS: Colores de Fondo e Imagenes de Fondo (cont)

Posicion de Imagen: Mueve la imagen dentro del elemento. Recibe como valores tamaños en pixeles y porcentajes, así como también right, bottom, left, etcétera.
CSS: body { background-position: right top }

Movimiento de Imagen: Establece si la imagen de fondo se va a mover junto con la página al hacer scroll. Valores: fixed, scroll, inherit e initial.
CSS:body { background-attachment: fixed }

Tamaño de Imagen: Recibe como valor contain, cover, inherit; así como también tamaños en pixeles y porcentajes indicando con el primer valor el ancho y con el segundo el alto..
CSS: body { background-size: 130px }

CSS: Selectores

Regla CSS: selector {propiedad: valor;}
body {
background-color: purple;
font-family: sans-serif;
text-align: center;
}

Selectores de ID: se indica con #:
CSS: #saludo{color: blue}
HTML: <h3 id="saludo">¡Hola!</h3>

Selectores de clase: se indica con ".clase".
CSS: .noticia { font-size: 22px }
HTML: <h3 class="noticia">Una noticia</h3>

Selectores de Etiqueta: se aplicará sobre todas las etiquetas indicadas.
CSS: p { color: gray }
HTML: Ad lorem ipsum

Selectores Combinado: en este caso un selector de etiqueta con uno de clase.
CSS h2.subtitulo { color: yellow }
HTML: <h2 class="subtitulo">Un subtítulo</h2>

Selectores Descendentes: Son mas espesíficos.
En este caso se aplicara uncamente a las etiquetas <li> contenidas en la etiqueta <ul> con el ID #lista.
CSS: ul#lista li { text-align: center }
HTML: <ul id="lista">
li>Primer ítem

CSS: Selectores (cont)

Selectores Universales: Afecta a todo el documento
*{font-family: arial}

Selectores de Atributo

Pseudo Selectores

pseudo clases: *permiten aplicar estilos en función de:*

- *Los estados de los elementos.*
- *La ubicación dentro de la estructura de HTML.*
- *La presencia de ciertos atributos de HTML.*

:link: Se utiliza para aplicar estilo a los enlaces **<a>** que tengan la propiedad *href*
a:link {
background-color: rgb(234, 0, 255);{
border-color: rgb(161, 17, 89);{
color: red;{
}

:visited: Se utiliza para aplicar estilo a los enlaces **<a>** que han sido visitados al menos una vez por parte del usuario.
a:visited {
background-color: rgb(234, 0, 255);{
border-color: rgb(161, 17, 89);{
color: red;{
}

:hover: Si bien está relacionado con los **enlaces**, puede ser aplicado a cualquier otro elemento de HTML. Se utiliza para aplicar estilo a cualquier elemento sobre el cual el usuario posicione el cursor.
a:hover {
background-color: gold;
}

:active[(1)]: aplica estilo a los enlaces **<a>** que estén siendo clickeados por el usuario. Normalmente se utiliza para la animación del click.
a:active {
background-color: rgb(234, 0, 255);
border-color: rgb(161, 17, 89);
color: red;
}

:focus: Se aplica sobre el **<imput>** cuando el cursor se encuentra dentro del elemento. El caso más normal es cuando el usuario está completando un campo de un formulario.
input:focus {
color: orange;
font-weight: bold;
}

Pseudo Selectores (cont)

:disabled: Se aplica cuando un elemento está deshabilitado, Normalmente se utiliza para darle estilos a los campos que no se pueden completar en un formulario. O a aquellas opciones que están desactivadas.
input:disabled {
background-color: gray;
}

Pseudo elementos: nos permiten crear elementos desde CSS sin tener que modificar la estructura del HTML.
Para usarlos, escribimos el nombre del selector primero, seguido de doble dos puntos :: y el pseudo elemento que queramos utilizar.

::before Se utiliza junto con la propiedad content para introducir contenido en el documento antes del contenido interno del elemento.
div::before {
content: "Esto se renderiza antes del elemento";
color: red;
}

::after Se utiliza junto con la propiedad content para introducir contenido en el documento después del contenido interno del elemento.
div::after {
content: "Esto se renderiza después del elemento";
color: red;
}

(1) : :active serán anulados por cualquier pseudoclase posterior relacionada con el enlace (:link, :hover o :visited) que tenga al menos la misma especificidad. utilizar respetando el orden :link — :visited — :hover — :active.

CSS: Media queries

HTML: <meta name="viewport" content="width=device-width, initial-scale=1">

min-width: “Si como mínimo el viewport tiene N píxeles de ancho, apliquemos estas reglas”.
@media (min-width: 460px)
{ body { background: red;}
}

max-width: “Si como máximo viewport tiene N píxeles de ancho, apliquemos estas reglas”.
@media (max-width: 768px)
{ body { background: yellow;}
}

Cheatography

HTML Basico Cheat Sheet

by Aleordoh (Martin Ordonez) via cheatography.com/146230/cs/31992/

CSS: Media queries (cont)

Orientación: “Si como máximo el viewport tiene N píxeles de ancho y además el dispositivo está en posición vertical/horizontal, apliquemos estas reglas”.
@media (max-width: 460px) and (orientation: landscape)
{ body { background: blue; }
}

Estrategia de diseño: Mobile Firts: Si utilizamos mobile first como estrategia de diseño, la idea es determinar de manera general las reglas CSS para pequeñas pantallas para luego, a través de media queries, ir aclarando el comportamiento en viewports más grandes.
CSS: body { background: red; }
@media (min-width: 460px){ / *Tablets* / }
@media (min-width: 768px){ / *Laptop* /}

breakpoints: 0-480 Smaller smartophones,
481-768 Tablets and larger smartphones,
769-1279 Laptosp, 1280+ Larger desktops

Reglas de CSS que nos permiten cambiar los estilos de los elementos en función de las Características del dispositivo que esté visualizando nuestro sitio.
Por lo general, se escriben al final de nuestra hoja de CSS.

HTML Formularios-1-

HTML Formularios-1- (cont)

<select> (Lista desplegable) nos permite agregar un componente que muestra opciones.
<option> Son las opciones de un select.
<form action="/colors" method="GET">
<label>Seleccione un color:</label>
<select name="color">
<option value="#ff0000">Rojo</option>
<option value="#00ff00">Verde</option>
<option value="#0000ff">Azul</option>
<option value="#770077" selected>Morado</option>
</select>
</form>
El atributo **selected** es opcional, se encarga de preseleccionar la opción

<button> Nos permite generar un botón.
Con la propiedad type definimos el tipo.
Un botón de tipo reset reinicia el formulario a su estado inicial.
Un botón de tipo submit se encarga de enviar el formulario.
Un botón de tipo button no realizará ninguna acción por defecto. Por lo general, programaremos una con ayuda de JavaScrip.
<form action="/login" method="POST">
<label>Nombre:</label>
<input type="text" name="usuario">
<label>Email:</label>
<input type="email" name="email">
<button type="submit">Enviar</button>
</form>

HTML Formularios -2-

Radio buttons: R ☐, al hacer click en él queda seleccionado. Para crear un elemento de tipo radio usamos un input cuyo atributo type tiene el valor radio. Podemos agrupar elementos de tipo radio y, en ese caso, solo se podrá elegir una opción del conjunto.{nl}}Para agruparlos usamos para todos los elementos el mismo valor en el atributo name.
La propiedad **checked** preselecciona la opcion
<input type="radio" name="medio-de-pago" value="Efectivo">
<input type="radio" name="medio-de-pago" value="Débito">
<input type="radio" name="medio-de-pago" value="Tarjeta" checked>

Checkbox: Representa una opción ☐, al hacer click en él queda seleccionado y con otro click quitamos la selección.
Para crear un elemento de tipo checkbox usamos un input cuyo atributo type tiene el valor checkbox.
Podemos agrupar elementos de tipo checkbox y, en ese caso, se podrán elegir múltiples opciones del conjunto. Para agruparlos, todos los elementos deben tener el mismo valor en el atributo name.
<input type="checkbox" name="hobbies" value="Música">
<input type="checkbox" name="hobbies" value="Pintura">
<input type="checkbox" name="hobbies" value="Juegos" checked>

<form> indica que vamos a crear un formulario
form action="/registro" method="POST">
<!-- Aquí van los campos -->
</form>
action define la ruta en donde se va a procesar la información capturada. **method** define cómo se enviará la información hay solo dos valores posibles: GET y POST.

<imput> puede recibir varios atributos
type permite obtener distintos tipos de campos
require hace que el campo sea obligatorio.
placeholder Da una pista al usuario sobre lo que puede introducir.

```
<form action="/registro" method="POST">
<input type="text" name="usuario" required>
<input type="password" name="clave" required>
<input type="email" name="email">
<input type="tel" name="telefono">
<input type="number" name="edad">
</form>
```

<label>Sirve para asociar un texto descriptivo a un campo determinado.

```
<form action="/registro" method="POST">
<label for="nombre">Nombre:</label>
<input type="text" name="nombre" id="nombre">
<label for="email">Email:</label>
<input type="email" name="email" id="email">
<label for="telefono">Edad:</label>
<input type="tel" name="edad" id="telefono">
</form>
```

<textarea> nos permite texto de gran tamaño y con múltiples líneas.

```
<form action="/contact" method="POST">
<label for="comentario">Deje su Comentario:</label>
<textarea name="comentario"
id="comentario">Contenido</textarea>
</form>
```

HTML Formularios -2- (cont)

El **checkbox** puede ser útil para recolectar información que responde a preguntas simples del estilo sí/no o verdadero/falso.
En ese caso no es necesario agregar un valor, ya que, en caso de seleccionarse la opción, llegará con el valor “on”.
<label for="terminos">Acepto los términos.</label>
<input type="checkbox" name="terminos">

HTML Formularios Avanzados -3-

El input de fecha: usamos un input a cuyo atributo type le damos el valor date.
<input type="date" name="fecha" value="2021-07-22" min="2018--01-01" max="2022-12-31">
min y **max** son opcionales

El input de archivos Nos permite cargar uno o más archivos desde la computadora.
<input type="file" name="avatar" accept=".jpg" multiple>
multiple es opcional y permite subir mas de un archivo.

CSS: Propiedades tipográficas

Familia Tipografica: tipo de letra
CSS: p { font-family: Arial, sans-serif; }

Tamaño Tipográfico: tamaño de la letra indicada en pixeles
CSS: p { font-size: 23px; }

Estilo Tipográfico: (italic, normal, oblique)
CSS: p { font-style: normal; }

Peso Tipográfico: (bold, lighter,normal, o numeros de 100 en 100)
CSS: p { font-weight: 500; }

Alineacion de Texto: center, left, right, inherit y justify.
CSS: p { text-align: justify; }

Decoración de Texto: line-through, underline, overline y none.
CSS: p { text-decoration: underline; }

Interlineado del Texto: Se estipula que sea 8 unidades mayor al font-size.
CSS: p { line-height: 20px; }

CSS: Propiedades tipográficas (cont)

Color de Texto: Se pueden indicar de las siguientes formas: Nombre , Hexadecimal (#f05331), RGB (rgb(255, 100, 50)) y RGBA (rgba(122, 50, 125, 0.5)). En el caso de RGBA la ultima cifra va del 0 al 1 e indica la opacidad.
CSS: p { background-color: rojo}
CSS: p { background-color: #3459ff }
CSS: p { background-color: rgb(12, 34, 32) }

Opacidad: le otorga transparencia a un elemento
CSS: p { opacity: 0.5 }

CSS: Medidas Relativas

Porcentajes:Cualquier medida expresada en porcentaje siempre estará relacionada con la medida (en ese mismo eje) del elemento padre que la contiene.
CSS: .elementoContenedor { width: 300px }
.elementoInterior { width: 50% } // Será 150px

em: Si utilizamos ems en una propiedad que no sea font-size, se tomará para el cálculo el font-size que tenga el elemento que estemos modificando.
CSS:p { font-size: 20px;
line-height: 2em; // 20px *2 = 40px{{n}} padding: 1.5em; // 20px* 1.5 = 30px }

rem: Los rems funcionan muy parecido a los ems, con la diferencia de que siempre tomarán de base el tamaño de font size del elemento <html>.
CSS:css p { font-size: 1.5rem }

vw y vh: *vw o viewport width es relativo al ancho total del viewport.* vh o viewport height* es relativo al alto total del viewport.
CSS: div { width: 25vw; // 25% del ancho disponible
height: 50vh; // 50% del alto disponible }

Las **medidas relativas** son aquellas que tienen en cuenta el contexto donde se encuentran. Si el contexto cambia, estas medidas cambiarán con él.

CSS: Box Model

width y height: Ancho y alto
div {width: 130px; height: 120px;}

padding: Espacio de relleno que podemos agregar entre el contenido del elemento y su borde.
div { padding: 12px; } (se aplica a los cuatros lados
div { padding: 12px 15px 18px 21px; } (arriba, derecha, abajo, izquierda)

border: Podemos asignarle un valor a esta propiedad definiendo el estilo de línea (solid, dotted, dashed o double, su espesor y su color.
div { border: 3px solid yellow } (solid: estilo de linea, 3px grosor, yellow color).

margin: Hace referencia al margen exterior. Sirve para separar una caja de la otra.
div { margin: 15px }

border-radius: Redondea los bordes.
div{border-radius: 5px;}

box-sizing: content-box: Aplica el ancho y alto que definamos al contenido del elemento.
box-sizing: border-box: El ancho y alto que indiquemos tomará en cuenta no solo el contenido del elemento, sino también el padding y el borde, dejando solo el margen por fuera.
Es una práctica muy común asignarle box-sizing: border-box a todos los elementos del sitio con la siguiente línea de código: ***{box-sizing: border-box}**

Cada elemento en HTML es una caja, y esas cajas se componen de márgenes (margin), bordes (border), relleno (padding) y finalmente el contenido (content).
Estas propiedades pueden aplicarse de manera diferente a los cuatro lados de cada caja (top, right, bottom, left). Ver Imagen

CSS: Estructura básica de Flexbox

Flexbox propone una estructura basada en el uso de un contenedor padre **(Flex-container)** y sus elementos hijos **(Flex-items)**.
Por defecto, los elementos hijos de un contenedor flex van a tratar de entrar todos en una misma línea. Para aclararle al contenedor que debe respetar el ancho definido de sus hijos usamos la propiedad **flex-wrap** con el valor wrap:
.contenedor-padre {
display: flex;
flex-wrap: wrap;
}
flex-wrap también puede recibir los valores nowrap y wrap-reverse.
Un **flex-item**, a su vez, puede convertirse en un **flex-container**. Para eso, solo hace falta asignarle la regla **display:flex**, para que así sus elementos hijos pasen a ser **flex-items**.

CSS: Flexbox

Para utilizarlo se debe asignar la regla **display: flex** a un selector CSS.

flex-direction: Con esta propiedad definimos el main axis (eje principal) del contenedor, que puede ser tanto horizontal como vertical.

flex-direction: row: Los ítems se disponen en el eje x, de izquierda a derecha.

flex-direction: row-reverse: Los ítems se disponen en el eje x, de derecha a izquierda.

flex-direction: column: Los ítems se disponen en el eje y, de arriba hacia abajo.

flex-direction: column-reverse: Los ítems se disponen en el eje y, de abajo hacia arriba.

justify-content: Con esta propiedad alineamos los ítems a lo largo del main axis.

justify-content: flex-start: Los ítems se alinean respecto del inicio del main axis que hayamos definido.

CSS: Flexbox (cont)

justify-content: flex-end: Los ítems se alinean respecto del final del main axis que hayamos definido.

justify-content: center: Los ítems se alinean en el centro del main axis.

justify-content: space-around: Los ítems se distribuyen de manera uniforme.-

align-items: Con esta propiedad alineamos los ítems a lo largo del cross axis.

align-items: stretch: Los ítems se ajustan para abarcar todo el contenedor.

align-items: flex-start: Los ítems se alinean al inicio del cross-axis.

align-items: flex-end: Los ítems se alinean al final del eje transversal.

align-items: center: Los ítems se alinean al centro del eje transversal.

Flexbox trabaja con dos ejes para desarrollar todo su flujo interno: *el **eje X*** y *el **eje Y***.
Según cómo decidamos ordenar los elementos, llamaremos **main axis** a uno y **cross axis** al otro.

CSS: Flex Item

order: Con esta propiedad controlamos el orden de cada ítem, sin importar el orden original que tengan en la estructura HTML. Esta propiedad recibe un número entero, positivo o negativo, como valor.
.caja {
order: 1;}

flex-grow: Con esta propiedad definimos cuánto puede llegar a crecer un ítem en caso de disponer de espacio libre en el contenedor.
.caja-b{
flex-grow: 1;}
1 equivale al 100% del espacio disponible, y 0 al 0%. Podemos usar cualquier valor en el medio, como 0.25 para el 25%.

CSS: Flex Item (cont)

align-self: Nos permite alinear, sobre el cross axis, a cada ítem. Puede tomar los valores: end, center, start, stretch:.
.contenedor-padre {
align-items: flex-start;}
.caja-dos{
align-self: flex-end;}