 Inicio del desarrollo web

el internet es el medio por el cual se realiza la comunicación entre el servidor y el cliente  
  
el cliente realiza peticiones y el servidor responde  
  
Especialidades:  
frontend: trabaja del lado del cliente, todas las funcionalidades, apariencia y realizar eventos para el usuario, además de la comunicación  
HTML  
CSS  
JavaScript  
  
backend: almacenar datos y seguridad de la aplicación.   
Java  
JavaScript  
PHP  
PYHTON

Domino: el nombre de mi página o de una pagina   
  
Hosting: el lugar donde está alojado las páginas.

En esta sesión de estudios hablamos de las tecnologías principales el desarrollo frontend y backend, lado del cliente lado del servidor.

 DOM

La visión del frontend es ver como esta la separación de cada elemento dentro de la página.

Una pagina no es un elemento único, es un elemento que está conformado por más elementos internos, y a su vez estos están formados por otros.  
  
es importante que un elemento HTML que este ubicado al mismo nivel que otros se denominaran Son elementos o nodos hermanos, cuando estos nodos tienen elementos internos, son llamados nodos hijos, y de igual si hay mas de un elemento hijo dentro del padre estos serán hermanos

Siempre use el DOCTYPE correcto  
  
siempre se debe validar la anidación de los nodos en cada documento HTML, esto para no generar ningún problema al momento de que el navegador intérprete nuestro documento.   
  
es posible obtener, crear y modificar un elemento a través de JavaScript.

Dom: modelo de objetos del documento.

Tener un buen manejo del DOM una buena anidación.  
además de dividir la página HTML por bloques

Forms y label, navegación ...  

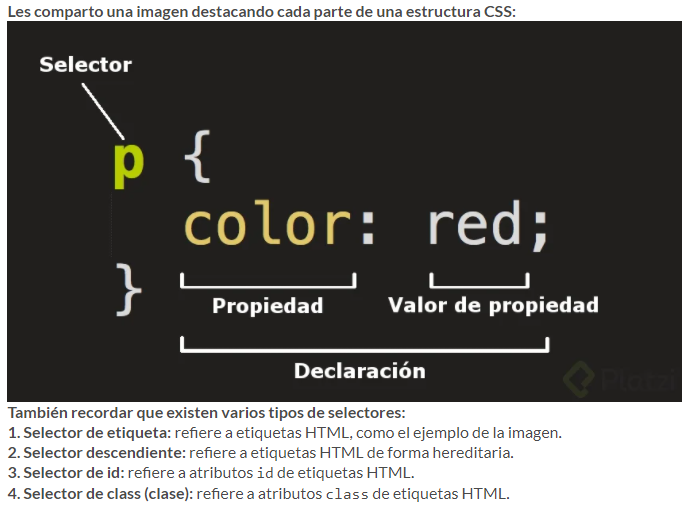

El atributo for tendrá como valor el id de alguna etiqueta cercana esto encadenará a las dos etiquetas.  
  
cuando le de clic al label hará focus al tag dueño del id.   
  
Los formularios se crean con la etiqueta form. El atributo principal de un formulario es action, ya que contiene la ruta a la que serán enviados los datos recolectados.  
  
# es un enlace interno en HTML, también indica que sea hace referencia a un id

Viewpord: área visible del navegador

Valores de targer:  
\_selft: abre el enlace en la misma página.

\_blank: abre el enlace en una nueva pestaña.  
\_parent: Carga la URL en el contexto de navegación padre (parent) del actual. Si no existe el padre, este se comporta del mismo modo que self.  
\_top: Carga la URL en el contexto más alto de navegación (el cual es un ancestro del actual, y no tiene padre (parent)). Si no hay padre (parent), este se comporta del mismo modo que \_self.

Selectores css , unidad PX y fuentes   

Impiertan impone un nuevo estilo sobre cualquier otro  
una regla de CSS se compone de : selectores, declaraciones, propiedades y valores   
**font-family**: define el tipo de fuente aplicado al texto.

**color**: define el colore del texto.

**line-height**: define la altura desde la base del texto hasta la base de la siguiente línea de texto.

**font-size**: define el tamaño del texto, admite cualquiera de las unidades de medida disponibles.

**letter-spacing**: define el espaciado entre las letras del texto.

**font-weight**: define el ““peso”” de la letra, negrita, normal, light y normalmente se indica en múltiplos de 100 o usando keywords.

**text-decoration**: define el decorado del texto como subrayado, tachado, con subrayado superior, etc.

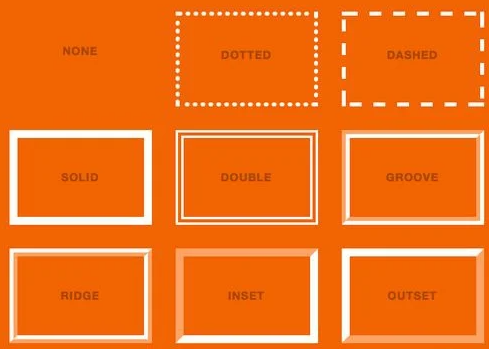
**text-transform**: permite transformar el estado de mayúsculas / minúsculas en el texto, usando uppercase para mayúsculas sostenidas, lowercase para minúsculas sostenidas, etc.

regla  
selector  
declaraciones  
propiedades

valores px:  
es la unidad de medida más pequeña en una imagen digital.

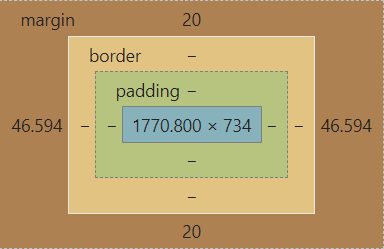
background y bordes   


* **background**: con la que se puede indicar un color, o usada de manera extendida, puede incluir color de fondo, url de la imagen, posición y modo de repetición de la imagen.
* **background-image**: contiene la url que se usará como fondo del elemento.
* **background-color**: indica el color de fondo, se puede usar en combinación con la imagen.
* **background-size**: se puede indicar en valores de alto y ancho o en alguna de las palabras claves permitidas: cover o contain.
* **background-position**: indica la posición de la imagen dentro del elemento, puede indicarse en unidades o en palabras claves como center, left, top y right.
* **background-repeat**: indica el método de repetición de la imagen de fondo, puede ser: repeat, repeat-x, repeat-y o no-repeat.



Modelo de caja   


Es muy importante tener la consideración el modelo de caja para el diseño responsivo, también que el tamaño de los elementos sea fijo y relativo el tamaño de los bloques que contengan estos elementos, esto último queda a resolver con mas experiencia.

  
  
El grafico de la parte de arriba explica el modelo de caja

Margin : indica el espacio que hay entre un elemento y otro.  
border: indica el groso o tipo de borde que tendrá cada elemento .  
Padding: indica el margen interno que hay en el elemento y la separación interna que hay con los elementos hijos.

El cuadro azul indica el tamaño del elemento.

Los valores caja serán sumados tener en cuenta esto, pero el modelo de caja se puede modificar con la propiedad box-sizing

Con la propiedad **box-sizing**, y en particular con el valor **border-box** de esta propiedad, podemos modificar el comportamiento del modelo de caja para que nuestro elemento nunca supere el tamaño máximo que le hayamos definido en **width y height**. Esta es la opción recomendada para trabajar

Tipos de display más utilizados   
  
  
  
Tema?  


Todos los elementos tienen un tipo de display, el display agrega ciertas propiedades a tu elemento, a continuación, algunos de los más utilizados:  
  
**block**: el elemento intenta abarcar todo el ancho posible.

**inline**: reduce su tamaño exclusivamente hasta lo que abarca su contenido, descartando las propiedades width y height.

**inline-block**: combina lo mejor de block e inline, ya que respeta las dimensiones indicadas en las propiedades width y height, pero coloca el elemento en línea (al costado) de elementos hermanos que también tengan display: inline o inline-block.

**flex**: asume algunas propiedades por defecto que favorecen la alineación de los elementos internos.

**grid**: similar a flex, asume algunas propiedades por defecto organizando los contenidos en filas y columnas.

**none**: oculta el elemento.

Flex box: justificación y alineación de elementos  


**Justify-content:** aliena los elementos de forma horizontal.

* **flex-start:** Alinea elementos al lado izquierdo del contenedor.
* **flex-end:** Alinea elementos al lado derecho del contenedor.
* center: Alinea elementos en el centro del contenedor.
* **space-between**: Muestra elementos con la misma distancia entre ellos.
* **space-around**: Muestra elementos con la misma separación alrededor de ellos

**Align-items**: alinea sus elementos verticalmente, alguno de sus valores son los siguientes:

* **flex-start:** Alinea elementos a la parte superior del contenedor.
* **flex-end:** Alinea elementos a la parte inferior del contenedor.
* **center:** Alinea elementos en el centro (verticalmente hablando) del contenedor.
* **baseline:** Muestra elementos en la línea base del contenedor
* **stretch:** Elementos se estiran para ajustarse al contenedor.

he tenido bastantes problemas con estas dos propiedades de Flex-box, pero en pocas palabras estados propiedades posicionan los elementos dentro del padre

Flex-box: Dirección , Order y  


**Flex-direction:** Esta propiedad CSS define la dirección de los elementos en el contenedor, y acepta los siguientes valores:

* row: Elementos son colocados en la misma dirección del texto.
* row-reverse: Elementos son colocados en la dirección opuesta al texto.
* column: Elementos se colocan de arriba hacia abajo.
* column-reverse: Elementos se colocan de abajo hacia arriba.

Cuando le damos una dirección o fila invertida se ven afectadas por la inversión las propiedades **Justify-content y** **Align-items**

Cuando la dirección se convierte en columna, Justify-content pasa ser modificar en vertical y Align-items en horizontal

**Order:** con esta propiedad nos permite modificar el orden de los elementos, por defecto el valor será cero( 0 ) para cada elemento, esta propiedad acepta valores positivos y negativos;  
**Align-Selft:** esta propiedad afecta la posición de un elemento individual y acepta los valores de Align-items por lo tanto modifica los elementos de manera vertical.

Flex-box: Wrap, Flex-Flow y align-conent  


La propiedad flex-wrap permite que los elemento se ubiquen en líneas posteriores o no de acuerdo con el valor de propiedad que se le da

* **nowrap**: Cada elemento se ajusta en una sola línea.
* **wrap**: los elementos se envuelven alrededor de líneas adicionales.
* **wrap-reverse**: Los elementos se envuelven alrededor de líneas adicionales en reversa.

**Flex-Flow:** agrupa los valores de las propiedades **Flex-Direction** y **Flex-wrap.**

Flex-box: Align-conent  


**Align-content**

Esto puede ser confuso, pero align-content determina el espacio entre las líneas, mientras que align-items determina como los elementos en su conjunto están alineados dentro del contenedor. Cuando hay solo una línea, align-content no tiene efecto.

* flex-start: Las líneas se posicionan en la parte superior del contenedor.
* flex-end: Las líneas se posicionan en la parte inferior del contenedor.
* center: Las líneas se posicionan en el centro (verticalmente hablando) del contenedor.
* space-between: Las líneas se muestran con la misma distancia entre ellas.
* space-around: Las líneas se muestran con la misma separación alrededor de ellas.
* stretch: Las líneas se estiran para ajustarse al contenedor.







