# Responsive wed Design (Diseño web adaptable).

En esta Certificación de diseño web receptivo, aprenderá los lenguajes que los desarrolladores utilizan para crear páginas web: HTML (lenguaje de marcado de hipertexto) para contenido y CSS (hojas de estilo en cascada) para diseño.

Primero, creará una aplicación de fotos de gatos para aprender los conceptos básicos de HTML y CSS. Más adelante, aprenderá técnicas modernas como las variables CSS mediante la creación de un pingüino y las mejores prácticas de accesibilidad mediante la creación de un formulario web.

Finalmente, aprenderá a crear páginas web que respondan a diferentes tamaños de pantalla mediante la creación de una tarjeta de Twitter con Flexbox y un diseño de blog complejo con CSS Grid

# HTML y HTML5 basicos#

HTML es un lenguaje de marcado que usa una sintaxis o notación especial para describir la estructura de una página web al navegador. Los elementos HTML suelen tener etiquetas de apertura y cierre que rodean y dan significado al contenido. Por ejemplo, diferentes elementos pueden describir el texto como un encabezado, párrafo o elemento de lista.

En este curso, creará una aplicación de fotografía de gatos para aprender algunos de los elementos HTML más comunes: los componentes básicos de cualquier página web.

## Say Hello to HTML Elements (saluda a los elementos HTML).

La mayoría de los elementos HTML tienen etiquetas de apertura de y de cierre (apertura **<**h1**>**) y (cierre**</**h1**>**). La diferencia entre apertura y cierre es barra diagonal.

## titles with h elements (Títulos con los elementos “H”)

Los elementos **h1** se utilizan para títulos principales, mientras que los elementos **h2** generalmente se utilizan para subtítulos. Los elementos **h3**, **h4**, **h5** y **h6** se utilizan para indicar distintos niveles de subtítulos.

## Inform with the Informar con el Paragraph Element (elemento de párrafo).

Este elemento está representado por una **“p”** en minúscula porque todas las etiquetas en HTML se escriben en minúscula y este elemento significa **párrafo** y se utiliza para texto de párrafos ejemplo. <p> </p>.

## Fill in the Blank with Placeholder Text **(Rellene el espacio en** blanco con texto de marcador de posición).

El lorem ipsum es una forma de llenar de contenido en los códigos con elementos “p" que son los párrafos de la página. El lorem ipsum son una serie de línea que se usas para tener ubicado los párrafo de la página wed que vamos a utilizar.

## Uncomment HTML (Descomentar HTML).

Los comentarios son una forma de desactivar códigos sin tener que eliminarlos y también funciona para que otros desarrolladores comenten sin dañar el código. Los comentarios comienzan con <!-- y terminan -->.

Comment out HTML (comentar en HTML).

Para comentar en html debes inciar usando <!--y cerrar usando --> .

## Delete HTML Elements **Eliminar elementos en HTML.**

Solo borre el elemento h1 para simplificar la vista de la página, dejando el h2 como titulo principal.

## Introduction to HTML5 Elements (Introduccion a los elementos HTML 5).

HTML tiene etiquetas más descriptivas que HTML y esta incluye **main, header, footer, nav, video, article, section** y otros.

El main funciona para colocar el contenido principal de una página web de manera que los motores de búsqueda y otros desarrolladores encuentren fácil el contenido principal. Ejemplo

**<main>**

**<h1>**Hello World**</h1>**

**<p>**Hello Paragraph**</p>**

**</main>**

## Add Images to Your Website (Agregar imagen a su sitio web).

Para agregar imágenes al sitio web se usar el elemento **img** y agregar la URL de la imagen que se desea subir, usando el atributo **src.**

**<img src=**<https://www.freecatphotoapp.com/your-image.jpg>**>**

Recordar que los elementos ***img*** deben tener un atributo ***alt*** que es un texto que se utiliza para dar explicación de lo que contiene la imagen. Ejemplo

<img src="https://www.freecatphotoapp.com/your-image.jpg" alt="A business cat wearing a necktie.">

## Link to External Pages with Anchor Elements (Enlaces a paginas externas con elementos de anclaje).

Para vincular un contenido fuera de la página web necesitaremos el elemento “a” de (ancla) y estos elementos necesitan un dirección web de destino y usaremos el atributo href. Que a su vez se agregara un texto de anclaje (sobre el enlace). Ejemplo.

<a href="https://freecodecamp.org">this links to freecodecamp.org</a>.

## Link to Internal Sections of a Page with Anchor Elements (Enlaces a secciones internas de una pagina con elementos de ancla).

Los elementos de “a”(ancla) también se pueden vincular para crear enlaces dentro de la misma página web.

Para crear el enlace interno se sigue usando el atributo **href** y en vez de poner un enlace colocamos el símbolo de almohadilla (#) que servirá para definir el enlace interno que puede ser una palabra, para completar el enlace interno debemos agregar un **id** que generalmente en la parte de debajo de la página y ambos estará vinculado. Tanto la almohadilla (#) como el id ambos tendrán la misma palabra. Ejemplo

<a href="#footer">Jump to Bottom</a> ***parte incial***.

<footer id=footer>Copyright Cat Photo App</footer> **parte *final.*** La palabra footer esta enlazada con la almohadilla **(#)** y el **id** que es un identificador.

## Nest an Anchor Element within a Paragraph(Anidar un elemento de anclaje dentro de un párrafo).

Para agregar enlaces dentro de un párrafo utilizamos el elemento de anclaje <a>……..</a> dentro del párrafo <p>……… </p>. Dentro del anclaje utilizaremos un atributo llamado **target** que específica donde abrir el enlace, más el valor de **\_blank** que especifica abrir el enlace en una nueva pestaña. Ejemplo **target=”\_blank”**  más **href=”**aquí va el enlace**” +** lo que continua del párrafo, ojo debe tener coherencia el párrafo con el enlace.ejemplo:

<p>vermas <a href="https://freecatphotoapp.com" target="\_blank">fotos de gatos</a> </p>

Quedaría así: ver más fotos de gatos.

## Make Dead Links Using the Hash Symbol (crear enlaces muertos usando el simbolo hash).

Para crear un enlace muerto utilizaremos la almohadilla **(#)** esto porque a veces no tenemos definido dónde se vincularan. El enlace que trae el atributo **href=”**[**www.enlaces.com**](http://www.enlaces.com)” lo suplantamos por una almohadilla para crear un enlace muerto **href=”#”.**

## **Turn an Image into a Link (convertir una imagen en un enlace**).

Para convertir una imagen en enlace debemos incluirlo dentro del elemento de ancla <a>, utilizar la almohadilla en el anclaje para convertirlo en enlace muerto ejemplo:

<a href="#"><img src="https://bit.ly/fcc-running-cats" alt="Three kittens running towards the camera."></a>

Sabremos que el código es bueno cuando el puntero mormal del cursor se convierta en el puntero de cilc de enlace.

## Create a bulleted Unordered list (Crear una lista desordenada con viñetas).

Las listas desordenas comienza con el elemento <ul> de apertura seguido de un elemento <li>…</li> y cierran con un </ul> ejemplo.

<ul>

<li>milk</li>

<li>cheese</li>

</ul>.

## Create an Ordered List (Crear una lista ordenada)

Las listas ordenas comienza con el elemento <ol> de apertura seguido de un elemento <li>…</li> y cierran con un </ol> ejemplo.

<ol>

<li>Garfield</li>

<li>Sylvester</li>

</ol>

A diferencia que cuando ejecutamos el código la lista aparece enumerados en orden.

## Create a text field (crear un campo de texto)

La forma más conveniente de obtener información de usuario es usando el elemento <input type=”text”> esto creara una caja de texto así:

## Add placeholder text to a text Field (agregar texto de marcador de posicion a un campo de texto).

El texto de marcador de posición es lo que se muestra en el elemento <input> antes de que el usario haya escrito algo. Ejemplo:

Ingresar nombre

Al <input se le debe agregar  *placeholder=”ingresar nombre”>* ejemplo de código:

<input type="text" placeholder="this is placeholder text">.

## Create a form element (Crear un elemento de formulario).

Para crear un formulario debemos utilizar la etiqueta <form> que servirá para crear un conjunto de controles (botones, cajas de texto, casillas de verificación, botones radio, etc) que permiten al usuario introducir datos.



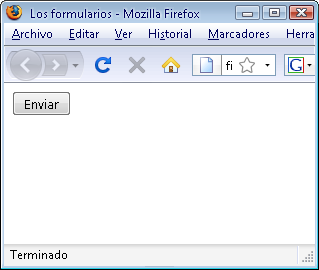
Al elemento form le agremamos el atributo action que indica la página a la **que** se envían los datos del **formulario. Ejemplo**

<form action="https://freecatphotoapp.com/submit-cat-photo">.

## Add a submit botton to a form (Agregar un botón enviar formulario).

A continuación se creara un botón (enviar) al formulario en uso, al hacer clic en ese botón se enviaran los datos del formulario al URL que agregamos en el atributo *action* del formulario. Ejemplo de código:

<button type="submit"> enviar </button>  Este es el resultado un botón así:



## Use HTML5 to Require a field (Use HTML5 para requerir un campo).

Esta sección consiste en que el campo de texto que se utilizó para almacenar datos, se le agregara un nuevo atributo que es el *required* dentro del input. Ejemplo:

<input type=”text” required> esto hará que no se pueda enviar el formulario sin completar ese campo. Para confirmarlo se intentara envía el formulario sin ingresar ningún texto y el formulario te notificara que el campo es obligatorio.

## Créate a set of radio buttons (Crear un conjunto de botones de opción).

Los botones de opción sirven para que los usuarios den una respuesta entre múltiples opciones.

A continuación se utilizara un elemento ***label*** que sirve como etiqueta y sirve principalmente, para mostrar información, como por ejemplo, exhibir un título, el nombre de una casilla, mostrar  el tiempo y muchas otras cosas más.

For es el único atributo que suele utilizarse en la etiqueta <label> y sirve para indicar el identificador (atributo id) del campo de formulario para el que esta etiqueta hace de título.

Es importante resaltar que todos los botones deben estar relacionado con el mismo atributo ***name*** ya que esto los encierra dentro de un circulo y te permitirá solo seleccionar un botón. #pregunta la relación entre for e id ¿Qué pasaría si dejo de usar el for o el id?

<label for="indoor">

<input id="indoor" type="radio" name="indoor-outdoor">Indoor

</label>.

## Create a set of checkboxes (Crear un conjunto de casillas de verificación).

En este sección también se crearan otros botones pero puedes seleccionar varios porque son type=” checkbox” es decir casillas de verificación, ósea varias casillas, también son de tipo input y se anida al elemento label, debes relacionar las casillas con el mismo atributo name

Y utilizar el for dentro del label que coincida con el id ejemplo.

#Pregunta: que pasaría si dejo de usar el for o el id?

<label for="loving"><input id="loving" type="checkbox" name="personality"> Loving</label>.

## Use the value attribute with radio buttons and checkboxes (Utilice el atributo de valor con botones de opción y casillas de verificación).

En esta sección solo le agregaremos un nuevo atributo a nuestro input y es ***Value*** y sirve para definir el nombre personal de cada botón. A continuación agregaremos un value a los botones tipo radio y tipo checkbox. Ejemplo.

# ¿Qué pasaría si no uso el atributo de value para los botones?

<label for="indoor">

<input id="indoor" value="indoor" type="radio" name="indoor-outdoor">Indoor

</label>

## Check radio buttons and checkboxes by Default (Marque los botones de opción y las casillas de verificación de forma predeterminada)

Para marcar de forma predeterminada unas de las casillas puede ser de tipo radio o tipo checkboxes solo debemos agregar la palabra checked dentro del input ejemplo:

<label for="indoor"><input id="indoor" type="radio" name="indoor-outdoor" value="indoor" **checked**> Indoor</label>.

Esto hara que los venga marcado pero se pueden cambiar según la opción que desees ingresar en el formulario.

## Nest many elements within a single div Element (Anidar muchos elementos dentro de un solo elemento div).

Definición. **div de** "**division**" -división . **Sirve para** crear secciones o agrupar contenidos. Sus etiquetas son: <**div**> y </**div**> (ambas obligatorias).

A continuacion colocamos un <div> al comienzo de las listas de gatos y uno al final </div>.

## Declare the doctype of an HTML (Declarar el doctype de un documento 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, un DOCTYPE distinto podría implicar hasta una visualización diferente del sitio web dentro del mismo navegador

El DOCTYPE va en la parte superior del documento, que servirá para decirle al navegador que versión de HTML está usando la página. Ya que el html es u lenguaje en evolución. Ejemplo

<!DOCTYPE html>.

## Define the head and body of an HTML Document(Definir el encabezado y el cuerpo de un documento HTML).

El **elemento <body> de HTML** representa el contenido de un documento HTML. Solo puede haber un elemento <body> en un documento. Cierre </body>

El **elemento HTML <head>** provee información general (metadatos) acerca del documento, incluyendo su título y enlaces a scripts y hojas de estilos. Cierre </head>.

# CSS básico

CSS, o hojas de estilo en cascada, le dice al navegador cómo mostrar el texto y otro contenido que escribe en HTML. Con CSS, puede controlar el color, la fuente, el tamaño, el espaciado y muchos otros aspectos de los elementos HTML.

Ahora que has descrito la estructura de tu aplicación de fotos de gatos, dale un poco de estilo con CSS.

## Cambiar el color del texto (change the color of text).

**style** - estilo. Es el elemento encargado de indicar la información de estilo. En este caso solo queremos cambar de color. Es importante resaltar que para terminar las declaraciones de **style** se debe usar **;** . ejemplo.

<h2 style="color: blue;">CatPhotoApp</h2>.

## Utilice selectors css para diseñar elementos (Use CSS selectors to style Elements).

Esta seccion explicara como establecer automáticamente que el h2 este de color rojo ejemplo

Se crea un código style en la parte superior ejemplo:

<style>

</style>

Y dentro de ese bloque creamos un selector de CSS para todos los elementos h2 esto para que todos los h2 tengan el mismo estilo en este caso color rojo. Ejemplo:

<style>

h2 {

color: red;

}

</style>

Hay que tener en cuenta que es importante tener llaves de apertura y cierre ( { y }) alrededor de las reglas de estilo de cada elemento. Y que estas aperturas estén entre las etiquetas de estilo de apertura y cierre.

## Usar una clase CSS para diseñar un elemento (Use a CSS Class to Style an Element).

Las clases son estilos reutilizables que se pueden agregar a elementos HTML.

Para agragar un esitlo a cualquier elemento HTML debemos declararlo de esta manera.

<style>

.blue-text {

color: blue;

}

</style>

Haciendo esta declaración podemos poner de color azul varias palabra de los elementos HTML de la siguiente manera:

<h2 class="blue-text">CatPhotoApp</h2> siempre usando el class=”blue-text”.

## Aplicar estilos a varios elementos con una clase CSS(Style Multiple Elements with a CSS class):

Esta sección aplicaremos de estilo (rojo) a elemento <p> de html con la siguiente declaración.

<style>

  .red-text {

    color: red;

  }

</style>

Y en el elemento <p> solo agregamos la clase ejemplo:<p class=”red-text”>.

## Cambiar el tamaño de fuente de un elemento (Change the Font Size of an Element).

A continuación estableceremos los tamaños. Solo agregamos el siguiente código al style que está en uso:

p {

  font-size: 16px;

}

Esto hara que todos los párrafos tengan el mismo tamaño, mientras más alto sea el px más grande serán las letras.

## Establecer una familia de fuentes de un elemento (Set the Font family of an Element).

, es el tipo de letra o familia de tipos de letra es el tipo de letra que se aplicará al cuerpo de texto en un navegador. Ejemplo. <p style="font-family: Arial;"> es decir con el Font-family será la herramienta para usar varios tipos de letras.

## Importar una fuenta de google (import a google font).

En este desafío usaremos letras para nuestro sitio wed que no son comunes de los navegadores, si no que utilizaremos letra de google (fuente o letra lobster) y para eso lo usaremos este link de referencia. Y lo pegaremos antes del elemento style. <link href="https://fonts.googleapis.com/css?family=Lobster" rel="stylesheet" type="text/css">. Este link lo buscaremos en GoogleFonts.

Después agregaremos la fuente al elemento que vayamos a utilizar en este caso se lo importaremos al h2 pero lo declararemos dentro del style de la siguiente manera

H2 {

font-family:lobster;

}.

## Especificar como deben degradarse las fuentes (Specify how fonts should degrade).

Cuando una fuente o la letra que queremos usar no está disponible, se le puede indicar al navegador que use otra fuente o otra letra se explicara en el siguiente ejemplo:

H2{Font-family: lobster, monospace;}

Aquí estamos indicándole al navegador que en caso de que no agarre lobster, use monospace.

Las familias de fuentes genéricas incluyen monospace, serif y sans-serif los nombres de estas familias no necesitan comillas y puedes usar mayúsculas y minúsculas porque son palabras claves.

## Tamaño de sus imágenes (Size your Images).

Para modificar el tamaño de una imagen en css debemos usar una propiedad llamada ***width***. Y usaremos px para especificar en ancho de la imagen. Lo declaramos en el< style> y después lo agregamos en el elemento <img> creando una clase (class) ejemplo:

<style>.smaller-image {Width:100px } <style/>

Y el elemento <img class=”smaller-image”> estos procedimiento modificaran el tamaño de la imagen.

## Agrege bordes alrededor de sus elementos (Add Borders around Your Elements).

Los bordes de css tienen propiedades como ***style, color*** y ***width***

Style: solido

Color: verde

Width (tamaño): 10px

Se crea la siguiente clase thick-green-border. Ejemplo:

<style>

.thin-red-border {

border-color: green;

border-width: 10px;

border-style: solid;

}

</style>

Después lo agregamos a elemento html en ensta caso al elemento <img> dentro del class que ya se encuentra en el elemento ejemplo <img class:=”smaller-image thick-green-border”> esto creara un cuadro verde alrededor de la foto del gato.

## Agregar esquinas redondeadas con radio de borde (add rounded corners with border-radius).

El objetivo es redondear las esquinas del marco que tiene el gato a continuancion agregaremos una propiedad del css llamada border-radius en el cual su valor será de 10 pixeles (px). Para que esto de resultado agragaremos border-radius a la clase .thick-green-border.

## Hacer imágenes circulares con un radio de borde (Make circular images with a border-radius).

Además de usar pixeles pixeles(px) también puede se puede especificar en el border-radius el uso de un porcentaje, es decir, se usaría % en el valor ejemplo. Border-radius: 50%.

## Dar un color de fondo al elemento div (give a background to a div element).

Se puede establecer el color de fondo de un elemento con la propiedad background-color. Ejemplo:

.green-background {

background-color: green;

}

Con esta declaración en el style podemos colocar de fondo verde el elemento div

Ejemplo <div class=”Green-backgruond”>.

## Establecer la identificación de un elemento (Set the id of an Element).

En esta sección solo agregaremos un identificador (id) al form (formulario), este atributo tiene varios beneficios porque se puede usar para diseñar o darle estilo a un solo elemento.

Los nombres de los identificadores (id) deben ser único, por lo cual es una buena práctica no signa el mismo id a más de un elemento ejemplo de la clase:

<form Id=”cat-photo-app” >.

## Usar el atributo id para diseñar un elemento (Use an id attribute to Style an element).

Los id también se pueden diseñar usando css y no son reutilizables es decir solo se utilizara en un elemento. Estos atributos poseen una mayor importancia que una elemento class, por lo que si ambos se aplican dentro del mismo elemento se aplicara el atributo del (id). Practica de clase: darle un fondo verde al formulario ejemplo:

#cat-photo-form{

    background-color: green; (en CSS)

<form id=”cat-photo-form”> (en HTML).

## Ajustar el relleno de un elemento (adjust the padding of an element).

A continuación 3 propiedades que controlan el espacio que rodea a cada elemento HTML: *padding, border, y maargin.*

**padding=relleno** controla la cantidad de espacio entre el contenido de elemento y su borde. Es decir controla la distancia o espacio que hay entre texto que está en el cuadro y los bordes del cuadro. Depende de cómo lo quieras el cuadro puede estar más pegado a la palabra que se encuentra dentro o puede estar más lejos.

Padding:20px;

## Ajustar el margen de un elemento (adjust the margin of an element).

La propiedad CSS **margin**establece el margen para los cuatro lados. Es una abreviación para evitar tener que establecer cada lado por separado con las otras propiedades de margen:  [margin-top (en-US)](https://developer.mozilla.org/en-US/docs/Web/CSS/margin-top), [margin-right](https://developer.mozilla.org/es/docs/Web/CSS/margin-right), [margin-bottom](https://developer.mozilla.org/es/docs/Web/CSS/margin-bottom) y [margin-left](https://developer.mozilla.org/en-US/docs/Web/CSS/margin-left)

El elemento margin controla la cantidad de espacio entre border y los elementos circundantes es decir los elementos que los rodea.

Padding:20px;

Margin:20px;

## Agregar un margen negativo a un elemento (add a negative margin to an element).

Si queremos que nuestro margen sea negativo solo le agraremos es signo de menos (-) a nuestro controlador de margen ejemplo:

Margin:-15px

De esta manera el margen ocupara todo el ancho horizontal de cuadro que lo rodea

## Agregar relleno diferente a cada lado de un elemento (add different to each side an elememt).

Cuando tenemos diferentes padding o rellenos en un elemento a cada uno podemos darle su lado.

CSS le permite controlar el paddingde los cuatro lados individuales de un elemento con los padding-top, padding-right, padding-bottom, y padding-left propiedades

padding-top: 40px;

padding-right: 20px;

padding-bottom: 20px;

padding-left: 40px;.

## Agregar diferentes margenes a cada lado de un elemento (add Different Margins to Each Side of an Element).

Para agregar diferentes márgenes a cada uno de los lados de nuestro elemento por querer personalizar el elemento css permite controlar los 4 lados con estas propiedades.

Margin-top: ………………. Puede ser 60px

Margin-right:……………….. Puede ser 3px

Margin-bottom: …………… 40px

Margin-left:………………….20 px depende de la personalización que quieres dar.

## Utilice la notación en el sentido de las agujas del reloj para especificar el relleno de un elemento (Use Clockwise Notation to Specify the Padding of an Element).

En lugar de especificar de un elemento padding-top, padding-right, padding-bottom, y padding-leftpropiedades de forma individual, puede especificar todos ellos en una línea, como este:

padding: 10px 20px 10px 20px;

Estos cuatro valores funcionan como un reloj: arriba, derecha, abajo, izquierda y producirán exactamente el mismo resultado que usar las instrucciones de relleno específicas de los lados.

## Utilice la notación en el sentido de las agujas del reloj para especificar el margen de un elemento (Use Clockwise Notation to Specify the margin of an Element).

En lugar de especificar de un elemento margin-top, margin-right, margin-bottom, y margin-leftpropiedades de forma individual, puede especificar todos ellos en una línea, como este:

margin: 10px 20px 10px 20px;

Estos cuatro valores funcionan como un reloj: arriba, derecha, abajo, izquierda y producirán exactamente el mismo resultado que usar las instrucciones de margen específicas de cada lado.

## Utilice selectores de atributos para diseñar elementos (Use Attribute Selectors to Style Elements).

Aquí aprenderemos a utilizar un nuevo selector de CSS y se puede utilizar para seleccionar grupos personalizados de elementos para diseñar.

El selector es el siguiente [attr=value] para diseñar las casillas de verificación. También empareja y aplica estilo a elementos con un valor de tributo específico. Ejemplo

[type='checkbox']

{margin:10px 0px 15px 0px} aquí se le está aplicando un margen a las casillas de verificación (checkbox).

## Comprender las unidades absolutas frente a las relativas (Understand Absolute versus Relative Units).

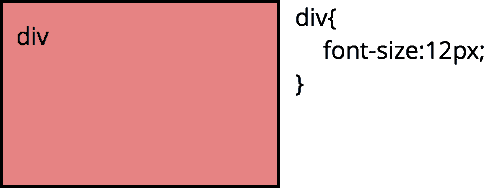
A continuación se utilizara otra unidad de longitud en css. Las unidades absolutas y las unidades relativas.

Las unidades absolutas se relacionan con las unidades físicas de longitud (in) y se refieren a pulgadas y milímetros (mm). Por ejemplo, (in)y se (mm) refieren a pulgadas y milímetros, respectivamente. Las unidades de longitud absoluta se aproximan a la medida real en una pantalla, pero existen algunas diferencias dependiendo de la resolución de la pantalla

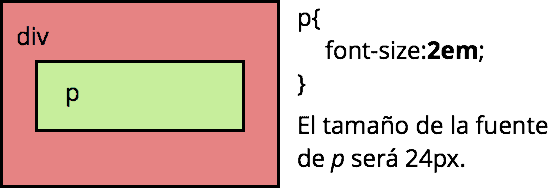
Las unidades relativas, como (em) o (rem) son relativa a otro valor de longitud. Por ejemplo el (em) se basa en el tamaño de la fuente de su elemento o elemento padre.

### Em:

Vamos a complicar un poquito las cosas, pero no demasiado. El em es otra medida relativa, pero esta vez al tamaño de la fuente de su elemento padre.



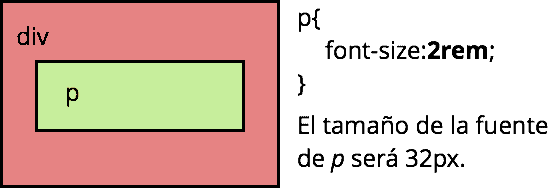
Imagina que tenemos un contenedor div al que le asignamos una fuente de tamaño 12px.



Ahora añadimos un párrafo p como elemento hijo del div. Automáticamente éste heredará el mismo tamaño de fuente que su padre: 12px. Pero si ahora le indicamos con CSS que su fuente tenga un tamaño de 2em, esto equivaldrá al doble del tamaño de su elemento padre, por tanto, 24px. Si le asignáramos un tamaño de 0.5em su tamaño sería equivalente a 6px, la mitad

### Rem:

Es una unidad relativa, pero esta vez relativa al tamaño de la fuente del root element, es decir el elemento html. Esto quiere decir que cuando nosotros no especificamos un tamaño de fuente (letra) concreto para el elemento html, éste ya viene con uno predefinido. Normalmente con valor de 16px, pero nosotros podemos especificarle el tamaño que deseemos.



Esta vez p tendrá un tamaño de 32px, ¿por qué? porque da igual el tamaño que tenga su padre, sólo tenemos en cuenta el tamaño del elemento html, que por defecto es de 16px. Es decir que (rem) siempre va a multiplicar 16 que es el valor de la fuente de root element.

## Aplicar estilo al elemento del cuerpo HTML (Style the HTML Body Element).

Ahora comencemos de nuevo y hablemos de la herencia CSS. Cada página HTML tiene un elemento body Podemos probar que el  elemento body existe aquí dándole un background-color color negro. Podemos hacer esto agregando lo siguiente a nuestro elemento style:

body { background-color: black; }.

## Heredar estilos del elemento de cuerpo (Inherit Styles from the Body Element).

El elemento body también se puede diseñar con css. Ejemplo de estilo del body

<style>

  body {

    background-color: black;

    font-family: monospace;

    color:green;

  }

</style>

<h1> hello world </h1>.

## Priorizar un estilo sobre otro (Prioritize One Style Over another).

A veces, sus elementos HTML recibirán varios estilos que entran en conflicto entre sí.Por ejemplo, su elemento h1 no puede ser verde y rosa al mismo tiempo.

¿qué sucede cuando creamos una clase que hace que el texto sea rosa y luego lo aplicamos a un elemento. ¿Nuestra clase anulará la propiedad CSS body del elemento color: green?

<style>

  body {

    background-color: black;

    font-family: monospace;

    color: green;

  }

  .pink-text{

    color:pink;

  }

</style>

<h1 class="pink-text">Hello World!</h1>

Respuesta: si lo elimina haciendo que el elemento class tenga más prioridad esto hace que las palabras queden en rosa.

## Anular estilos en CSS subsiguientes (Override Styles in Subsequent CSS).

En esta sección se creó una segunda clase, .blue-text{color: blue} Darle un color azul al título cuando ya existía el rosado. Esto hizo que las letras se pusieran azul Evidenciando que no importa en qué orden se enumeran las clases en el html. Sin embargo, el orden de las declaraciones class en la sección <style> es lo importante porque la segunda declaración siempre prevalecerá sobre la primera.

## Anular declaraciones de clase por atributos de ID de estilo (Override Class Declarations by Styling ID attributes).

Solo anularemos las clases pink-text y blue-text, y hagamos que el h1 sea anaranjado, otorgándole un identificación (id) y luego la diseñamos.

<h1 id=”orange-text”……. Identificando. html

#orange-text{color: orange;} ………. Diseñando o declando. Css

## Anular declaraciones de clase con estilos en linea (Override Class Declarations with Inline Styles).

<h1 style="color: white;"> esto es un estilo de línea y lo utilizaremos dentro de nuestro h1 para que sea blanco. No importa que en nuestro h1 se encuentre un identificador u otras clases. Se pondrá blanco si lo hacemos directo en el html.

## Anule todos los demás estilos utilizando importante (Overrider All Other Styles by using Important).

Usando !important en unas de las declaraciones de estilos en este caso en pink-text hará que el color rosado sea el que aparezca en el titulo por encima del identificador y del estilo en línea (el que va declarado directamente en el html) ejemplo:

.pink-text { color: pink !important; }.

## Utilice codigo hexadecimal para colores específicos (Use Hex for Specific Colors).

En CSS, podemos usar 6 dígitos hexadecimales para representar colores, 2 para cada uno de los componentes Rojo (r) son los 2 primeros, Verde (g) son los 2 digitos des centro y azul (b) son los 2 últimos digitos. Por ejemplo, #000000 es negro y también es el valor más bajo posible. Decimal son los numero del 0 al 9 y en hexadecimal se utilizan los números de 0 al 9 pero como es hexadecimal (16) se le agrega A,B,C,D,F que representan los valores del 10 al 15.

Con la combinación de estos símbolos se podrán combinar distintos colores.

## Utilice código hexadecimal para mezclar colores (Use hex code to mix colors).

Para revisar, los códigos hexadecimales usan 6 dígitos hexadecimales para representar colores, dos para cada componente rojo (R), verde (G) y azul (B).

A partir de estos tres colores puros (rojo, verde y azul), podemos variar las cantidades de cada uno para crear más de 16 millones de colores más.

Por ejemplo, el naranja es rojo puro, mezclado con algo de verde y no azul. En código hexadecimal, esto se traduce en ser #FFA500.

El dígito 0es el número más bajo en código hexadecimal y representa una ausencia total de color.

El dígito Fes el número más alto en código hexadecimal y representa el brillo máximo posible

Reemplace las palabras de color en nuestro style elemento con sus códigos hexadecimales correctos.

|  |  |
| --- | --- |
| **Color** | **Código hexadecimal** |
| Dodger Blue | #1E90FF |
| Verde | #00FF00 |
| naranja | #FFA500 |
| rojo | #FF0000 |
|  |  |

## Usar código Hexadecimal abreviado (Use abbreviated Hex Code).

Mucha gente se siente abrumada por las posibilidades de más de 16 millones de colores. Y es difícil recordar el código hexadecimal. Afortunadamente, puedes acortarlo.

Por ejemplo, el código hexadecimal de rojo #FF0000se puede abreviar a #F00. Esta forma abreviada da un dígito para el rojo, un dígito para el verde y un dígito para el azul.

Esto reduce el número total de colores posibles a alrededor de 4000. Pero los navegadores interpretarán #FF0000y #F00exactamente del mismo color.

Adelante, intente usar los códigos hexadecimales abreviados para colorear los elementos correctos.

|  |  |
| --- | --- |
| **Color** | **Código hexadecimal corto** |
| Cian | #0FF |
| Verde | #0F0 |
| rojo | #F00 |
| Fucsia | #F0F |

## Utilice valores RGB para colorear elementos (Use RGB values to Color Elements).

Otra forma de representar colores en CSS es mediante el uso de RGB valores. RGB son las siglas de Red-Green-Blue En lugar de usar seis dígitos hexadecimales como lo hace con el código hexadecimal, RGB especifica el brillo de cada color con un número entre 0 y 255.

El RGB valor del negro se ve así:

rgb(0, 0, 0)

El RGB valor del blanco se ve así:

rgb(255, 255, 255)

En lugar de usar seis dígitos hexadecimales como lo hace con el código hexadecimal, RGB especifica el brillo de cada color con un número entre 0 y 255

body { background-color: rgb(0,0,0);} esto crearía un fondo negro en nuestra página.

## Utilice RGB para mezclar colores (Use RGB Mix Colors).

Solo cambiaremos los colores al igual que se hizo con los hexadecimales pero con RGB.

.orchid-text {color: rgb(218, 122, 214);}

.sienna-text {color: rgb(160, 82, 45);}.

## Utilice variables CSS para cambiar varios elementos a la vez (Use CSS Variables to change several elements at once).

Las variables CSS son una forma poderosa de cambiar muchas propiedades de estilo CSS a la vez cambiando solo un valor.

En esta clase solo sustituimos los colores del pingüino que eran negro, blanco y amarillo y fueron reemplazados por gris, blanco y anaranjado. Ejemplo:

<style>

.penguin {

--penguin-skin: gray;

    --penguin-belly: white;

    --penguin-beak: orange;

## Creacion de una variable CSS presonalizada (Create a custom CSS variable).

Para crear una variable en ccs solo se necesita darle un nombre con 2 guiones delante y asignarle un valor ejemplo:

--penguin-skin: gray;

Ahora puede usar esa variable en otra parte de su CSS para cambiar el valor de otros elementos a gris.

## Utilice una variable CSS personalizada (use a custom CSS variable).

Una vez que se creó la variable podemos asignar ese valor a otras propiedades CSS haciendo referencia al nombre que le dimos (--penguin-skin).

**background: var(--penguin-skin);**

Esto cambiará el fondo de cualquier elemento al que esté apuntando a gris porque ese es el valor de la variable  --penguin-skin. Tenga en cuenta que los estilos no se aplicarán a menos que los nombres de las variables coincidan exactamente.

## Adjuntar un valor de respaldo a una variable (Attach a Fallback value to a CSS Variable).

Cuando use su variable como un valor de propiedad CSS, puede adjuntar un valor de respaldo al que su navegador volverá si la variable dada no es válida.

Esto se utiliza para que el navegador tenga un color para mostrar si no puede encontrar la primera variable. Ejemplo.

background: var(--penguin-skin, black);

Esto establecera el fondo en black (negro) si su primera variable no fue configurada.

## Mejorar la compatibilidad con los resguardos del navegador (Improve Compatinility with Browser Fallbacks).

Cuando trabajamos con css posiblemente nos encontremos con navegadores que no son compatibles, por eso hay que tener un plan b para evitar problemas. Cuando un navegador no reconoce una variable css como Internet explorer el navegador usara cualquier otro valor que tenga para esa propiedad y si no consigue otro valor, volverá al valor predeterminado

Por lo tanto debemos otorgarle un respaldo y no es más que agregar otro valor. Declarado más ampliamente admitido antes de la declaración de variable ejemplo.

.red-box {

***background: red;***

    background: var(--red-color);.

## Heredar Variables CSS (Inherit CSS Variables).

Para hacer uso de la herencia, las variables CSS a menudo se definen en el elemento raiz.

:root La [pseudo-clase](https://developer.mozilla.org/es/docs/Web/CSS/Pseudo-classes) **:root**  de [CSS](https://developer.mozilla.org/es/docs/Web/CSS)  selecciona el elemento raíz de un árbol que representa el documento. En HTML: root representa el elemento [<html>](https://developer.mozilla.org/es/docs/Web/HTML/Element/html)y es idéntico al selector html, excepto que su [especificidad](https://developer.mozilla.org/es/docs/Web/CSS/Specificity) es mayor.

 Al crear sus variables en :root, estarán disponibles globalmente y se puede acceder a ellas desde cualquier otro selector en la hoja de estilo.ejemplo.

<style>

:root{--penguin-belly: pink; }

Donde se encuentre el penguin-belly estara de Rosado.

## Cambiar una variable para un área específica (Change a variable for a specific area).

Es posible cambiar el valor de una variable asi este declarada en el roon es decir la raíz que por ende ese valor aparecerá en toda la pag. Luego podras escribir la variable en el elemento que quieras solo le cambiaras el valor. Ejemplo

.root { --penguin-belly: pink;}

Otro elemento .penguin{--penguin-belly; White;} se pondrá blanco.

## Use una consulta de medios para cambiar una variable (Use a media query to change a variable).

Las variables de CSS pueden simplificar la forma en que utilizamos las consultas de medios o *medio query*.

Que son los media query: son la forma de poder especificar diferentes **estilos** css para diferentes **tamaño** que tenga el navegador en este caso sobre cualquier dispositivo que se encuentre ya sea una Tablet, un celular o una pc es decir lo media query limitan los estilos css hasta cierto punto en donde cambiaran de acuerdo a las propiedades css que nosotros le definamos

Por ejemplo cuando la pantalla es más pequeña o más grande que punto de interrupción (es el punto que ya se ha declarado puede ser mayor o menor a 350px @media (max-width: 350px) se puede cambiar el valor de la variable y aplicara su estilo donde sea que se use.

<style>

  :root {

    --penguin-size: 300px; aqui se declara al pingüino con 300px normal

    --penguin-skin: gray;

    --penguin-belly: white;

    --penguin-beak: orange;  }

  @media (max-width: 350px) { aquí se declara que si el pingüino está por debajo o por arriba de 350px ocurrirá un cambio.

    :root {

      --penguin-size:200px; el cambio es todo lo declarado en @media

      --penguin-skin:black;  --penguin-beak:yellow; }.

# Diseño visual aplicado

El diseño visual es una combinación de tipografía, teoría del color, gráficos, animación, diseño de página y más para ayudar a transmitir su mensaje único.

En este curso, aprenderá a aplicar estos diferentes elementos de diseño visual a sus páginas web.

## Crear equilibrio visual con la propiedad de alineación de texto (create Visual Balance Using the text-aling Property).

En esta clase veremos cómo alinear nuestro texto, contenidos y títulos con la propiedad ***text-align*.**

**Text-align: Justify**; justifica todo hace que todas las líneas de texto, excepto la última, coincidan con los bordes izquierdo y derecho del cuadro de línea.

**text-align**: center; centra el texto

**text-align: right**; alinea el texto a la derecha

Y **text-align: left**;(el predeterminado) alinea el texto a la izquierda. Ejemplo.

<style>

  h4 {text-align: center;}

  p { text-align: justify; }.

## Ajustar el ancho de un elemento usando la propedad de ancho (Adjust the Width an Element Using the width Property).

Puede especificar el ancho de un elemento usando la propiedad width en CSS. Los valores se pueden dar en unidades de longitud relativa (como em), unidades de longitud absoluta (como px) o como un porcentaje del elemento principal que los contiene. Ejemplo

width: 245px; dentro de la class fullcard.

## Ajustar la altura de un elemento mediante la propiedad de altura (Adjust the Height an Element Using the height Property).

Puede especificar la altura de un elemento usando la propiedad height en CSS, similar a la propiedad width. Ejemplo.

H4{height: 25px}.

## Utilice la etiqueta fuerte para poner el texto en negrita (Use the strong tag to Make text Bold).

Para colocar palabras en **negrita** utilizamos la etiqueta ***strong*.** Agarramos cualquier palabra o varias palabras del párrafo y alrededor de esas palabras colocamos strong ejemplo

<strong>Stanford University</strong> resultado **Stanford University.**

## Utilice la etiqueta u para subrayar el texto (Use the u tag to underline text).

A continuación utilizaremos la etiqueta u para subrayar. Con la etiqueta u, el navegador aplica CSS de text-decoration: underline; al elemento. Ejemplo

<u>Ph.D. students</u> resultado Ph.D. students.

## Utilice la etiqueta em para poner el texto en cursiva (Use the em Tag to Italicize Text).

Para enfatizar el texto, puede usar la etiqueta em. Esto muestra el texto en cursiva, ya que el navegador aplica CSS font-style: italic; al elemento. Ejemplo.

<p>**<em>**Google was founded by Larry Page and Sergey Brin while they were <u>Ph.D. students</u> at <strong>Stanford University</strong>.**</em>**</p> *el resutaldo es que todas la palabras quedan en cursive*.

## Utilice la etiqueta s para tachar el texto (use the s Tag to Strkethrough Text).

Para tachar el texto, que es cuando una línea horizontal atraviesa los caracteres, puede usar la etiqueta s. Muestra que una sección de texto ya no es válida. Con la etiqueta s, el navegador aplica el CSS de text-decoration: line-through; al elemento. <s>Google</s> resultado: ~~Google.~~

## Crea una línea horizontal usando el elemento hr (Create a Horizontal line Using hr Element).

Puede usar la etiqueta hr para agregar una línea horizontal a lo ancho de su elemento contenedor. Esto se puede utilizar para definir un cambio de tema o para separar visualmente grupos de contenido. Ejemplo:

## Ajustar la propiedad de color de fondo del texto (adjust the background-color Property of text).

En esta sección solo agregaremos un color de fondo para nuestro título h4 con los siguientes elementos rgba() que significa:

R= rojo

G= verde

B= azul

A= alfa/nivel de opacidad (más claro o más oscuro), el valor alfa puede variar de 1, que completamente opaco o de un color sólido, a 0 que es completamente transparente o claro. El rgb utiliza los códigos hexadecimales. Ejemplo:

Background-color: rgba(45, 45, 45, 0.1);

## Ajustar el tamaño de un encabezado frente a una etiqueta de párrafo (Adjust the Size of a Header Versus a Paragraph Tag).

Solo agregaremos un tamaño diferente a nuestro encabezado diferente al párrafo o contenido para darle un nivel de importancia tanto visualmente como en el diseño ejemplo.

<style>

H4 {Font-size: 27px} resultado: alphabet.

## Agregar una sombra de caja a un elemento similar a una tarjeta (Add a box-shadow to a Card-like Element).

La propiedad **box-shadow** aplicauna o más sombras a un elemento.

La propiedad box-shadow toma valores para:

* Offset-x: (que tan lejos empujar la sombra horizontalmente del elemento),
* Offser-y: (que tan lejos empujar la sombra verticalmente desde el elemento),
* Blur-radius, (desenfoca)
* Spread-radius (dispersa) (opcional)
* Color, (eleccion).

/\* offset-x | offset-y | blur-radius | color \*/

box-shadow: 10px 5px 5px black;.

#thumbnail{

  box-shadow: 0 10px 20px  rgba(0,0,0,0.19), 0 6px 6px rgba(0,0,0,0.23);

## Disminuir la opacidad de un elemento (Decrease the Opacity an Element).

La propiedad opacity en CSS se utiliza para ajustar la opacidad o, a la inversa, la transparencia de un elemento.

Un valor de 1 es opaco, que no es transparente en absoluto.  
Un valor de 0,5 es medio transparente.  
Un valor de 0 es completamente transparente.

El valor dado se aplicará a todo el elemento, ya sea una imagen con algo de transparencia o los colores de primer plano y de fondo de un bloque de texto.

.links { text-align:

left; color: black;

**opacity: 0.7;**} resultado letras casi transparente.

## Utilice la propiedad de transformación de texto para convertir el texto en mayúsculas (Use the text-transform Property to Make text Uppercase).

La propiedad text-transform en CSS se usa para cambiar la apariencia del texto. Es una forma conveniente de asegurarse de que el texto de una página web aparezca de forma coherente, sin tener que cambiar el contenido del texto de los elementos HTML reales.

La siguiente tabla muestra cómo los diferentes text-transformvalores cambian el texto de ejemplo "Transfórmame".

| **Valor** | **Resultado** |
| --- | --- |
| lowercase | "transfórmame" |
| uppercase | "TRANSFORMAME" |
| capitalize | "Transfórmame" |
| initial | Usa el valor predeterminado |
| inherit | Usa el text-transform valor del elemento padre |
| none | **Predeterminado:** usar el texto original |
|  |  |

Objetivo colocar el h4 en mayúscula

## Establecer el tamaño de fuente para varios elementos de encabezado (Set the Font-size for Multiple Heading Elements).

Solo se establecio el tamaño de los títulos del ha1 hasta el h6 con la propiedad de Font-size de CSS. Ejemplo:

<style>

h1{font-size: 68px;}

h2{ font-size: 52px;}

h3{font-size: 40px;}

h4{font-size: 32px;}

h5{ font-size: 21px;}

h6{font-size: 14px;}

## Establecer el peso de fuente para varios elementos de encabezado (Set the Font-Weight for Multiple Heading Elements).

Se utilizara la propiedad Font-weight para establecer el grosor ejemplo:

<style>

h1{font-size: 68px;

font-weight: 800}

h2{ font-size: 52px;

font-weight: 600}

h3{font-size: 40px;

font-weight: 500 }

h4{font-size: 32px;

font-weight: 400 }

h5{ font-size: 21px;

font-weight: 300 }

h6{font-size: 14px;

font-weight: 200 } resultados letra más gruesas que otras en sentido decreciente.

## Establecer el tamaño del texto del parrafo (Ser the font-size of paragraph text).

Solo se estrablecera el tamaño de párrafo la propiedad de Font-size ejemplo:

<style>

  p { font-size: 16px;} todo le parrado tendrá un tamaño de letra de 16px.

## Establecer la altura de línea de los párrafos (Ser the line-heght of paragraphs).

A continuación estableceremos un espacio entre líneas para nuestro párrafo con la propiedad **line-height** ejemplo:<style>

p { font-size: 16px; line-height: 25px;}. Resultado separación entre lineas de 25px.

## Ajustar el estado de desplazamiento de una etiqueta de ancla (Adjust the Hover State of an Anchor tag).

Lo que haremos a continuación es cambiar el estilo de nuestra etiqueta de anclaje y se cambiara con un :hover que es una pseudo-clase de **CSS,** coincide cuando el usuario interactúa con un elemento con un dispositivo señalador, pero no necesariamente lo activa es decir esto crea estado de desplazamiento que cuando el curso pase por encima de la palabra que almacena un enlace cambiara de color. Ejemplo:

<style>

  a:hover{ con desplazamiento del cursor se pone de rojo.

    color: #00f;

  }

  a {

  color: black; sin desplazamiento queda negro.

  }

## Cambiar la posición relativa de un elemento (Change an Element´s Relative Position).

Para cambiar la posición de un elemento dentro de nuestro HTML usamos la propiedad **position** de CSS que sirve para especificar como un elemento es ubicado o posicionado dentro de nuestro documento o página web. Las propiedades [top](https://developer.mozilla.org/es/docs/Web/CSS/top), [right](https://developer.mozilla.org/es/docs/Web/CSS/right), [bottom](https://developer.mozilla.org/es/docs/Web/CSS/bottom), y [left](https://developer.mozilla.org/es/docs/Web/CSS/left) determinan la ubicación final de los elementos posicionados.

### Posición relativa

Esta propiedad se define a partir de la posición normal, o inicial, del elemento, y no lo libera del flujo de texto. Dando **valores** a left, top, right o bottom, el elemento se desplaza respecto a su posición inicial, pero los elementos que le siguen no cambian su comportamiento. Ejemplo:

<style>

  h2 {position: relative; top: 10px;}.

## Mover un elemento en una posición relativa con compensaciones Css (Move a Relative Positioned Element With Css Offsets).

Lo que aprenderemos aquí es que cuando queremos mover nuestro elemento relativamente: usando las propiedades top (arriba), right (derecha), bottom(abajo), left(izquierda) siempre se moverán en dirección opuesta. Ejemplo

Top (arriba) ------- bajara

Right (derecha) -- ira a la izquierda

Bottom (abajo) – ira para arriba Left (izquierda) – ira para la derecha.

## Bloquear un elemento a su padre con posicionamiento absoluto (Lock an Element to its Parent With Absolute Positioning).

### Posición absoluta

La posición absoluta libera al elemento del flujo normal del texto. Ese elemento queda entonces por encima de todos los demás elementos y no tiene ninguna influencia en la posición de los mismos dentro del diseño (es decir que se posiciona a la altura del elemento padre por encima de todos los elementos).

La posición absoluta se define mediante los **valores** left, top, right y bottom, los cuales se calculan respecto a los márgenes del elemento padre —siempre que el elemento padre tenga algún tipo de posición. Si no existe un elemento padre posicionado, se toma el elemento <body> como elemento padre.

A continuación estableceremos un elemento #searchbar (barra de busqueda) a la posición absulta y lo ubicaremos con sus posiciones ejemlo:

<style>

  #searchbar {

    position: absolute;

    top: 50px;

    right: 50px;

## Bloquear un elemento en la ventana del navegador con posicionamiento fijo (Lock an Element to the Browser Window with Fixed Positioning).

Fixed es un elemento con posición: fijo; se posiciona en relación con la ventana gráfica, es decir al navegador, lo que significa que siempre permanece en el mismo lugar incluso si se desplaza la página hacia abajo o arriba a la izquierda o la derecha. Las propiedades superior, derecha, inferior e izquierda se utilizan para colocar el elemento. Un elemento fijo no deja un hueco en la página donde normalmente se habría ubicado (o sea que cuando se mueve de lugar su antiguo sitio es ocupado).

ejemplo:

#navbar {

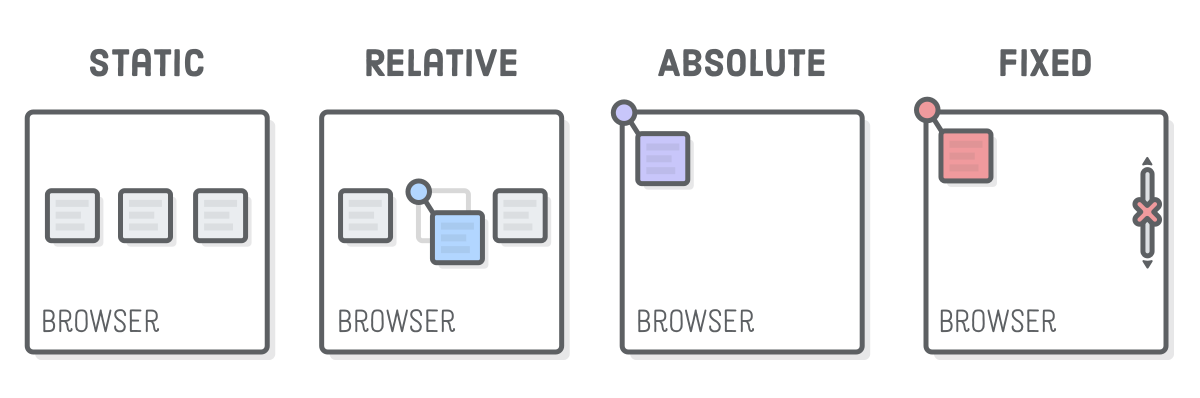
    position: fixed;

    top: 0px;

    left: 0px;

    width: 100%;

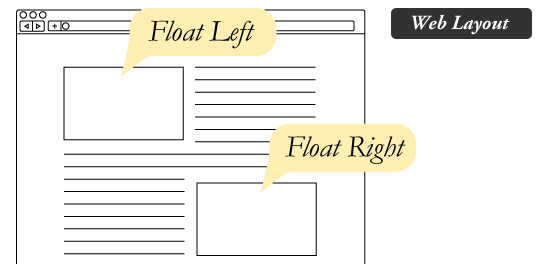
    background-color: #767676;



## Empuje los elementos hacia la izquierd o hacia la derecha con la propiedad flotante (Push Elements Left or Right with the float Property).

La propiedad CSS float ubica un elemento al lado izquierdo o derecho de su contenedor, permitiendo a los elementos de texto y en línea aparecer a su costado. El elemento es removido del normal flujo de la página, aunque aún sigue siendo parte del flujo. Se usa comúnmente con la propiedad width para especificar cuanto espacio horizontal Require el elemento flotante. Ejemplo:

Ejemplo



 #left {

      float: left;

      width: 50%;

    }

    #right {

      float: right;

      width: 40%;

## Cambiar la posición de los elementos superpuesto con la propiedad de índice z (Change the Position of Overlapping Elements with the z-index Property).

La propiedad CSS z-index indica el orden de un elemento [posicionado](https://developer.mozilla.org/es/docs/Web/CSS/position) y sus descendientes. Cuando varios elementos se superponen, los elementos con mayor valor z-index cubren aquellos con menor valor.

Sin embargo, la propiedad z-index puede especificar el orden en que se apilan los elementos uno encima del otro. Debe ser un número entero (es decir, un número entero y no un decimal), y los valores más altos para la propiedad z-index de un elemento lo mueven más arriba en la pila que aquellos con valores más bajos. (Es decir el índex que tenga el valor más alto será el que se pondrá por encima de todos). Ejemplo.

.first {

    background-color: red;

    position: absolute;

**z-index: 2;**

  }

  .second {

    background-color: blue;

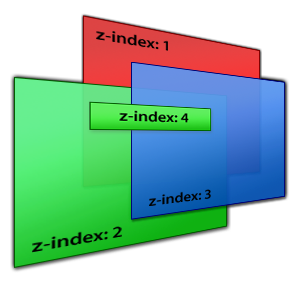
    position: absolute;

    left: 40px;

    top: 50px;

**z-index: 3;**

ejemplo grafico:



## Centrar un elemento horizontalmente usando la propiedad margin (Center an element Horizontally Using the margin Property).

En esta sección utlizaremos la propiedad margin co un valor auto para centrar un elemento de bloque horizontal. Ejemplo

div {

    background-color: blue;

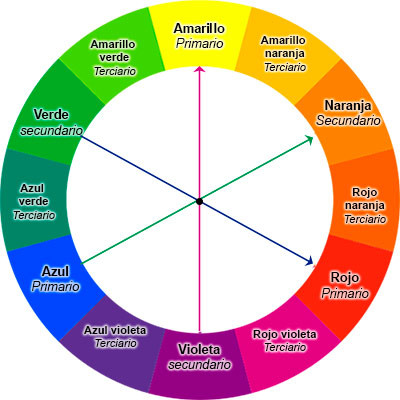
    height: 100px;

    width: 100px;

**margin:  auto;** esto hara que el elemento se centre en la página web.

## Más información sobre los colores complementarios (Learn about Complementary Colors).

Los colores opuestos o complementarios son aquellos colores que se encuentran en una posición oponible dentro del círculo cromático. Ejemplo.



En un sitio web, el color puede llamar la atención sobre el contenido, evocar emociones o crear armonía visual. El uso de diferentes combinaciones de colores realmente puede cambiar el aspecto de un sitio web, y se puede pensar mucho en elegir una paleta de colores que funcione con su contenido.

Algunos ejemplos de colores complementarios con sus códigos hexadecimales son:

rojo (# FF0000) y cian (# 00FFFF)  
verde (# 00FF00) y magenta (# FF00FF)  
azul (# 0000FF) y amarillo (# FFFF00)

**Nota:** Usar color puede ser una forma poderosa de agregar interés visual a una página. Sin embargo, el color por sí solo no debe utilizarse como la única forma de transmitir información importante porque es posible que los usuarios con discapacidad visual no comprendan ese contenido.

## Más información sobre los colores terciarios (learn about Tertiary Colors).

Los colores terciarios son el resultado de combinar un color primario con uno de sus colores secundarios vecinos. Por ejemplo, dentro del modelo de color RGB, el rojo (primario) y el amarillo (secundario) forman el naranja (terciario). Esto agrega seis colores más a una rueda de colores simple para un total de doce.

Existen varios métodos para seleccionar diferentes colores que dan como resultado una combinación armoniosa en el diseño. Un ejemplo que puede usar colores terciarios se llama esquema de color complementario dividido. Este esquema comienza con un color base, luego lo empareja con los dos colores adyacentes a su complemento. Los tres colores proporcionan un fuerte contraste visual en un diseño, pero son más sutiles que usar dos colores complementarios.

Aquí hay tres colores creados usando el esquema de complemento dividido:

| **Color** | **Código hexadecimal** |
| --- | --- |
| naranja | # FF7F00 |
| cian | # 00FFFF |
| frambuesa | # FF007F |

## Ajustar el color de varios elementos a colores complementarios (Adjust the color of Various Elements to Complementary Colors).

Los objetivos en esta sección es establecer colores complementarios a varios elementos.

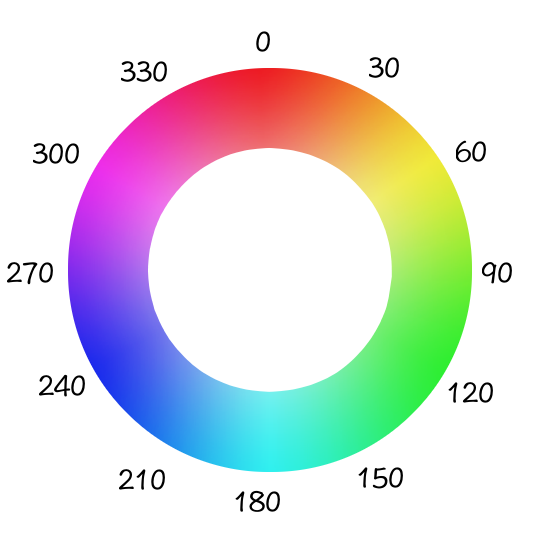
El desafío de los colores complementarios mostró que los colores opuestos en la rueda de colores pueden hacer que los demás parezcan más vibrantes cuando se colocan uno al lado del otro. Sin embargo, el fuerte contraste visual puede ser discordante si se usa en exceso en un sitio web y, a veces, puede hacer que el texto sea más difícil de leer si se coloca sobre un fondo de color complementario. En la práctica, uno de los colores suele ser el dominante y el complemento se utiliza para llamar la atención visual sobre cierto contenido de la página.

## Ajustar el tono de un color (Adjust the Hue of a Color).

Los colores tienen varias características que incluyen tono, saturación y luminosidad. CSS3 introdujo la propedad hsl() como una forma alternativa de elegir un color indicando directamente estas características.

El elemento hsl significa:

H= **tono**: es el color que observamos y lo escogemos mediante este círculo seleccionando un número desde 0 a 360.



S=**saturación**: es la cantidad de gris en un color. Un color completamente saturado no tiene gris, y un color mínimamente saturado es casi completamente gris. Esto se da como un porcentaje con 100% completamente saturado

L= **luminosidad:** es la cantidad de blanco o negro en un color. Se da un porcentaje que va del 0% (negro) al 100% (blanco), donde el 50% es el color normal. Ejemplo.

.green {

    background-color: hsl(120, 100%, 50%)

}

.cyan {

    background-color: hsl(180, 100%, 50%);

  }

  .blue {

    background-color: hsl(240, 100%, 50%);

  }

## Ajustar el tono de un color (Adjust the Tone of a Color).

En esta sección ajustamos las características de un color en uso, queremos usar el mismo color pero un poco más oscuro. Ejemplo:

header {

background-color: hsl(180, 90%, 35%); color principal.

 nav {

background-color: hsl(180, 80%, 25%); mismo color modificado.

}

## Crear un degradado lineal Css gradual (Create a Gradual Css Linear Gradient).

Para crear un degradado lineal se hace mediante la función de **background** usando la propiedad **linear-gradient()** ejemplo:

**Background: linear-gradient()**

El primer argumento (lo primero que se escribe dentro de los perentesis) especifica la dirección desde la que comienza la transición de color; se puede expresar como un grado, donde 90deghace un degradado horizontal (de izquierda a derecha) y 45deghace un degradado diagonal (de abajo a la izquierda a arriba a la derecha). Los siguientes argumentos especifican el orden de los colores utilizados en el degradado ejemplo.

**Background: linear-gradient(35deg, #ccffff, #ffcccc );**

## Utilice un degradado lineal CSS para crear un elemento rayado (Use a CSS Gradient to Create a Strped Element).

La función repeating-linear-gradient() es muy similar linear-gradient()con la principal diferencia de que repite el patrón de degradado especificado. repeating-linear-gradient() acepta una variedad de valores, pero para simplificar, trabajará con un valor de ángulo y valores de parada de color en este desafío.

background: repeating-linear-gradient(

      45deg,

      yellow 0px,

      yellow 40px,

      black 40px,

      black 80px

    );

Si cada dos valores de parada de color son del mismo color, la fusión no se nota porque está entre el mismo color, seguido de una transición difícil al siguiente color, por lo que terminas con rayas.

En el ejemplo que se muestra en el editor de código, el degradado comienza con el color yellow a 0 píxeles que se fusiona con el segundo color black 40 píxeles del inicio. Dado que la siguiente parada de color también está en 40 píxeles, el degradado cambia inmediatamente al tercer color green, que a su vez se fusiona con el cuarto valor de color, red ya que está a 80 píxeles del comienzo del degradado.

## Cree textura agregando un patrón sutil como imagen de fondo (create texture by Adding a Subtle Pattern as a Background Image).

A continuación agragaremos una imagen de fondo utilizando la propiedad background y le agregamos la función url() para vincular una imagen ejemplo:

<style>

  body {

background: url(<https://cdn-media-1.freecodecamp.org/imgr/MJAkxbh.png>).

## Utilice la propiedad de escala de transformación Css para cambiar el tamaño el tamaño de un elemento (Use the CSS Transform scale Property to Change the size of an element).

Para cambiar la escala (tamaño) de un elemento, CSS tiene la propiedad *Transform* junto a su función *scale()* ejemplo:

#ball2 {

    left: 65%;

    transform: scale(1.5);

## Utilice la propiedad de escala de transformación Css para cambiar el tamaño el tamaño al pasar el mouse (Use the CSS Transform scale Property to Scale an Elemento n Hover).

La propiedad *transform* tiene una variedad de Funciones que permiten escalar (subir el tamaño), mover, rotar, sesgar, entre otras a los elementos pero cuando se usa la pseudoclase *:hover*. La pseudo-clase :**hover** de **CSS** coincide cuando el usuario interactúa con un elemento con un dispositivo señalador, pero no necesariamente lo activa. Generalmente se activa cuando el usuario se desplaza sobre un elemento con el cursor (puntero del mouse)Ejemplo:

<style>

  div {

div:hover {

    transform: scale(1.1);}

}

## Utilice la propiedad de transformación CSS SkewX para sesgar un elemento a lo largo del eje X (Use the CSS Transform Property skewX to Skew an Element Along the X-Axis).

La siguiente función de la propiedad *transform* es *skewX(),* que sesga(Significa cortar o partir una cosa en diagonal) el elemento a lo largo de su ejex (horizontal).

Otra más de las transformaciones que nos permite realizar CSS3 es el sesgado (o skew), que consiste en torcer la figura de forma oblicua, en el caso de un cuadrado lo podríamos transformar en un rombo

Ejemplo

div {

    width: 70%;

    height: 100px;

    margin:  50px auto;

}

  #bottom {

    background-color: blue;

    transform: skewX(24deg);

## Utilice la propiedad de transformación CSS SkewX para sesgar un elemento a lo largo del eje Y (Use the CSS Transform Property skewX to Skew an Element Along the Y-Axis).

Haremos lo mismo que la sección anterior pero esta vez sobre el eje Y ejemplo:

<style>

  div {

    width: 70%;

    height: 100px;

    margin: 50px auto;

#top {

    background-color: red;

    transform: skewY(-10deg).

## Crear un gráfico usando Css (Create a Graphic Using CSS).

Al manipular diferentes selectores y propiedades, puede crear formas interesantes. Uno de los más fáciles de probar es la forma de una luna creciente. Para este desafío, debe trabajar con la propiedad **box-shadow** que establece la sombra de un elemento, junto con la propiedad border-radius que controla la redondez de las esquinas del elemento

La propiedad **box-shadow** aplicauna o más sombras a un elemento.

La propiedad box-shadow toma valores para:

* Offset-x: (que tan lejos empujar la sombra horizontalmente del elemento),
* Offser-y: (que tan lejos empujar la sombra verticalmente desde el elemento),
* Blur-radius, (difumina)
* Spread-radius (radio de extensión) (opcional)
* Color, (eleccion).

En esta clase se creara una luna creciente: ejemplo de código.

<style>

  .center {

    position: absolute;

    margin: auto;

    top: 0;

    right: 0;

    bottom: 0;

    left: 0;

    width: 100px;

    height: 100px;

    background-color: transparent;

    border-radius: 50%;

    box-shadow: 25px 10px 0px 0px blue;

## Crear una forma más compleja usando Css y html (Create a More Complex Shape Using CSS and HTML).

En este desafio usaremos ***::before*** y ***::after*** que significa antes y después y son pseudo-elementos. Y se utilizan para agregar algo antes o después de un elemento seleccionado. Para el pseudo-elementos **::before** y **::after** para que funcione correctamente, deben tener definida una propiedad ***Content:”;*** esta propiedad se usa generalmente para agregar cosas como una foto o texto al elemento seleccionado. Cuando el pseudo-elemento ::before y ::after se utilizan para formas maquillaje, la propiedad content sigue siendo necesaria, pero se estable en una cadena vacía.

Para utilizar el before o after debemos utilizar los selectores que deseemos con todos sus elementos y colocar al before o after como primer hijo del selector o elemento seleccionado ejemplo:

<style>

  .heart {

    position: absolute;

    margin: auto;

    top: 0;

    right: 0;

    bottom: 0;

    left: 0;

    background-color: pink;

    height: 50px;

    width: 50px;

    transform: rotate(-45deg);

  }

  .**heart::after** {

    background-color: pink;

    content: "";

    border-radius: 50%;

    position: absolute;

    width: 50px;

    height: 50px;

    top: 0px;

    left: 25px;

  }

  .heart**::before {**

**content: "";**

    background-color: pink;

    border-radius: 50%;

    position: absolute;

    width: 50px;

    height: 50px;

    top: -25px;

    left: 0px;

## Aprenda cómo funcionan las propiedades de animación y @kayframes CSS (learn How the Css @kayframes and animation Properties work).

Para animar un elemento necesitamos conocer las propiedades y las reglas @kayframes. Las propiedades controlan como debe comportarse la animación y las reglas (lo que va incluido en @kayfreames) controla lo que sucede durante la esa animación.

Nota: son 8 propiedades pero usaremos 2.

La propiedad *animation-name­* establece el nombre de la animación que luego se usa en @kayframes para decirle que reglas van con que animaciones. Ejemplo:

 #rect { < lleva un identificador id,

**animation-name**: rainbow; < propiedades comportamiento de la animación

**animation-duration**: 4s; <propiedades comportamiento de la animación

  }

  @keyframes **rainbow**{ <todo después del @ entra en las reglas keyframe

    0%{

      background-color: blue;

    }

  50% {

    background-color: green;

  }

  100%{

    background-color: yellow;

  }

  }

@kayframes es como especificar que sucede dentro de la animación durante la duración. Esto se hace dando propiedades Css para “cuadros” especifico durante la animación, con porcentajes de 0% al 100%. Es decir lo que la animación muestre es lo que nosotros le hayamos declarado en los porcentajes en el ejemplo se muestra 3 declaraciones de porcentaje 0%, 50%, 100% que cada una tiene un color diferente y se resaltaran durante la animación, empezara con el azul porque está en 0% que vendría siendo el inicio, después verde que es 50%, y por ultimo amarillo que es el 100%. Esto lo veremos durante el tiempo declarado en la propiedad animation-duration.

## Utilice la animación CSS para cambiar el estado de desplazamiento de un botón (Use CSS animation to change the Hover state of a Button).

Seguiremos utilizando *@kayfremes* para cambiar el color de un botón en estado de desplazamiento es decir cuando el ratón o el cursor pase por encima. Ejemplo

button:hover {

    animation-name: background-color;

    animation-duration: 500ms;

  }

  @keyframes background-color {

    100% {

      background-color: #4791d0;

    }

  }

## Modificar el relleno de una aplicación (modify Fill mode of an Animation).

A continuación haremos que el color que le pusimos al botón no se desaparezca después del medio segundo 500ms si no que permanezca usando como propiedad *animation-fill-mode* que significa modo de relleno de animación.

Lo interesante de esta propiedad es que podemos agregarla sin eliminar la que se encontraba anteriormente que era por duración. Es decir que podemos incluirla después de a ver terminado la animación ejemplo:

button:hover {

    animation-name: background-color;

    animation-duration: 500ms;

    animation-fill-mode: forwards;    🡨 🡨 🡨 🡨

  }

## Crear movimiento usando animación CSS (Create movement Using Css animation).

Podemos usar las propiedades de right, left, top y bottom para crear movimientos en las reglas de animación. Ejemplo:

@keyframes rainbow { colocar el top, botton, left y right dentro de @keyframes

    0% {

      background-color: blue;

**top: 0px;**

**left: 0px;**

    }

    50% {

      background-color: green;

**top: 50px;**

**left: 25px**;

    }

    100% {

      background-color: yellow;

**top: 0px;**

**left: -25px**;

    }

## Cree una dirección visual desvaneciendo un elemento de izquierda a derecha (Create visual Direction by Fading an Element fron left to Rigth).

Para lograr este desafio agregaremos una nueva regla a nuestro *@keyframes* que es *opacity* que significa opacidad Falta de claridad o transparencia ejemplo.

 animation-name: fade;

    animation-duration: 3s;

  }

  @keyframes fade {

    50% {

      left: 60%;

    opacity: 0.1;

    }

  } esto hará que la esfera que se creó se desvanezca cuando se mueva a la derecha.

## Animar elementos continuamente con un numero de animaciones infinito (Animate elements Continually Using an Indinite Animation Count).

En este desafío estableceremos una nueva propiedad animación existe una propiedad en uso: *animation-interation-count* que significa recuento de animaciones y permite controlar cuantas veces le gustaría recorrer la animación. Ejemplo.

animation-iteration-count: 3; quiere decir que la animación se moverá 3 veces.

Pero para que la animación se mueva infinitamente solamente cambiamos el 3 y agregamos infinite ejemplo:

animation-iteration-count: infinite;

## Hacer un latido de CSS usando un conteo de animacion infinito (Make a CSS Heartbeat using an Infinite Animation Count).

En esa clase usaremos la propiedad *animation-iteration-count* que significa recuento de iteraciones de animación. A esta propiedad le agregaremos el valor de *infinite* ejemplo

animation-iteration-count: infinite;

## Animar elementos a tasas variables (Animate Elements at Variable Rates).

Para lograr este objetico cambiaremos las reglas en @kayframes el valor de la estrella 1 que está en 20% lo cambiaremos a 50% para que su velocidad aumente ejemplo de código:

@keyframes twinkle-1 {

**50% {**

      transform: scale(0.5);

      opacity: 0.5;

    }

  }

  @keyframes twinkle-2 {

**20% {**

      transform: scale(0.5);

      opacity: 0.5;

## Animar varios elementos a velocidades variables (Animate Multiple Elements at variable Rates).

En esta clase también cambiaremos la velocidad con que parpadean las estrellas pero desde la propiedad *animation-duration* cambiándole el valor a cada una ejemplo:

.star-1 {

    margin-top: 15%;

    margin-left: 60%;

**animation-duration: 1s;**

    animation-name: twinkle;

  }

  .star-2 {

    margin-top: 25%;

    margin-left: 25%;

**animation-duration: 0.9s;**

    animation-name: twinkle;

  }

  .star-3 {

    margin-top: 10%;

    margin-left: 50%;

**animation-duration: 1.1s**;

    animation-name: twinkle;

Esto hará que las estrellas no parpadeen al mismo tiempo.

## Cambiar el tiempo de animación con palabras clave (change Animation timng with keywords).

En las animaciones CSS, la propiedad ***animation-timing-function*** controla la rapidez con la que cambia un elemento animado durante la duración de la animación. Si la animación es un automóvil que se mueve del punto A al punto B en un tiempo determinado (su *animation-duration*), *animation-timing-function* dice cómo acelera y desacelera el automóvil en el transcurso del viaje.

Esta propiedad tiene varios valores.

***ease*:** comienza lento, se acelera en el medio y vuelve a disminuir al final.

***Ease-out***: es rápido al principo y luego se ralentiza.

***Ease-in:*** es lento al principio y luego se acelera al final.

***Linear:*** aplica una velocidad de animación en todo momento.

Ejemplo:

#ball1 {

    left:27%;

    animation-timing-function: linear;

  }

  #ball2 {

    left:56%;

    animation-timing-function: ease-out;

  }

## Descubra cómo funcionan las curvas de Bézier (Learn How Bezier Curves Work).

Para saber cómo funcionan las curvas de Bezier ***cubic-bezier*** seguiremos utilizando la propiedad ***animation-timing-function***.

Una curva de Bézier cúbica está definida por cuatro puntos P0, P1, P2 y P3. P0 y P3 son el inicio y el final de la curva y, en CSS, estos puntos son fijos ya que las coordenadas son proporciones. P0 es (0, 0) y representa el tiempo inicial y el estado inicial, P3 es (1, 1) y representa el tiempo final y el estado final.

La función cubic-bezier () se puede utilizar con la propiedad [***animation-timing-function***](https://www.w3schools.com/cssref/css3_pr_animation-timing-function.asp) y la propiedad [***transition-timing-function***](https://www.w3schools.com/cssref/css3_pr_transition-timing-function.asp) .

Las funciones de la propiedad animation-timing-function que son **ease**, **linear,** ease-**in** y **ease-out** todas estas funciones estan basadas en la función cubic-bezier ejemplo:

* **ease**: especifica que la transición CSS comenzara lentamente, luego ira mas rápida, y terminara reduciendo su velocidad otra vez, lo que equivalente a cubic Bezier (0.25,0.1,0.25,1)
* **linear**: especifica que la transición sera constante, con la misma velocidad de principio a fin. Equivalente a cubic-bezier (0,0,1,1).
* **ease-in**: en este caso la transición CSS tendrá un comienzo lento equivalente a cubic-bezier (0.42,0,1,1).
* **ease-out**: en el extremo opuesto al anterior, tenemos una transición con un final lento, lo que equivale a cubic-bezier (0,0,0.58,1).
* ease-in-out: si combinamos los dos anteriores, el resultado seria un inicio y un final lento. Equivalente a cubic-bezier (0.42,0,0.58,1).

Por eso en la función cubic-bezier podemos adaptarla a nuestro gusto.

Ejemplo:

animation-timing-function: cubic-bezier(0.25, 0.25, 0.75, 0.75);

## Usar una curva de Bézier para mover un gráfico (Use a Bezier Curve to Move a Graphic).

El objetivo del desafío es animar a 2 elementos con la misma velocidad uno se encuentra bajo la función ease-out y el otro lo igualaremos utilizando una función de curva de Bézier personalizada. Ejemplo:

#red {

    background: red;

    left: 27%;

**animation-timing-function: cubic-bezier(0,0,0.58,1);**

  }

  #blue {

    background: blue;

    left: 56%;

**animation-timing-function: ease-out;**

ambos elementos tienen el mismo movimiento.

## Haz que el movimiento sea más natural usando una curva Bézier (Make Motion More Natural Using a Bezier Curve).

# Accesibilidad Aplicada

En el desarrollo web, la accesibilidad se refiere al contenido web y una IU (interfaz de usuario) que puede ser entendida, navegada e interactuada por una amplia audiencia. Esto incluye a personas con discapacidades visuales, auditivas, de movilidad o cognitivas.

En este curso, aprenderá las mejores prácticas para crear páginas web accesibles para todos.

## Agregue una alternativa de texto a las imágenes para la accesiblidad para personas con discapacidad visual (Add a Text Alternative to images for Visually Impaired Accessibility).

Alt es un atributo que viene en la etiqueta img, que sirve para describir el contenido de una imagen. Un atributo alt ayuda en los casos en que la imagen no se carga o un usuario no puede verla. Los motores de búsqueda también lo utilizan para comprender que contiene una imagen e incluirla en los resultados de búsqueda. Ejemplo.

<img src="importantLogo.jpeg" alt="Company logo">.

## Sepa cuando el texto alternativo debe dejarse en blanco (Know when alt Text Should be left blank).

En el desafio anterior entendimos que es obligatorio incluir el atributo alt en la etiqueta img, pero cuando las imagines se usan para decoración o la imagen ya viene explicita en el contenido de texto igual se utiliza el atributo alt pero en una cadena vacia ejemplo

<img src="visualDecoration.jpeg" alt="">

Las imágenes de fondo también suelen incluirse bajo la etiqueta "decorativa". Sin embargo, normalmente se aplican con reglas CSS y, por lo tanto, no forman parte del proceso de los lectores de pantalla de marcado.

**Nota:** para las imágenes con un título, es posible que desee incluir alt texto, ya que ayuda a los motores de búsqueda a catalogar el contenido de la imagen.

## Utilice encabezados para mostrar relaciones jerárquicas de contenido (Use Headings to Show Hierarchical Relationships of Content).

Los encabezados (del h1 al h6) son etiquetas que ayudan a proporcionar estructura y etiquetado al contenido. Ya que los lectores de pantallas pueden configurarse para leer solo los encabezados de una página para que el usuario obtenga un resumen, esto significa que las etiquetas de encabezado en su marcado tienen un significado semántico y se relacionan entre sí, es decir, que el titulo tiene que tener relación con su contenido

Los encabezados con rango igual (o superior) comienzan nuevas secciones implícitas (título nuevo), los encabezados con rango más bajo comienzan subsecciones del anterior, es decir, vendrían siendo subtítulos.

Por ejemplo, una página con un elemento h2 seguido de varias subsecciones etiquetadas con etiquetas  h4 confundiría al usuario de un lector de pantalla. Con seis opciones, es tentador usar una etiqueta porque se ve mejor en un navegador, pero puede usar CSS para editar el tamaño relativo

Obetivo de la clase ordenar los encabezados (h1, h2, h3, h4, h5, y h6) en su respectivo orden.

## Ir directamente al contenido usando el elemento principal (jump Straight to the Content Using the main Element).

El elemento *main* se una para envolver el contenido principal y se debe usar uno solo por página. Esta destina a rodear la información relacionada con el tema central de la página.

La etiqueta ***main*** también tiene una característica de referencia incorporada que la tecnología de asistencia puede usar para navegar al contenido principal rápidamente.

 Las tecnologías de asistencia pueden acceder a esta información para proporcionar un mejor resumen de la página o opciones de navegación a sus usuarios.

<main> </main>

## Envolver un contenido en el elemento articulo (Wrap Content in the article Element).

***Article*** es un elemento de seccionamiento y se utiliza para en envolver contenido independiente y autónomo. La etiqueta funciona con entradas de blog, publicaciones en foros o artículos de noticias.

**Nota:** El elemento section también es nuevo en HTML5 y tiene un significado semántico ligeramente diferente al de article. Un article es para contenido independiente y  section es para agrupar contenido relacionado temáticamente. Se pueden utilizar entre sí, según sea necesario. Por ejemplo, si un libro es el article, entonces cada capítulo es un section. Cuando no exista una relación entre los grupos de contenido, utilice a div.

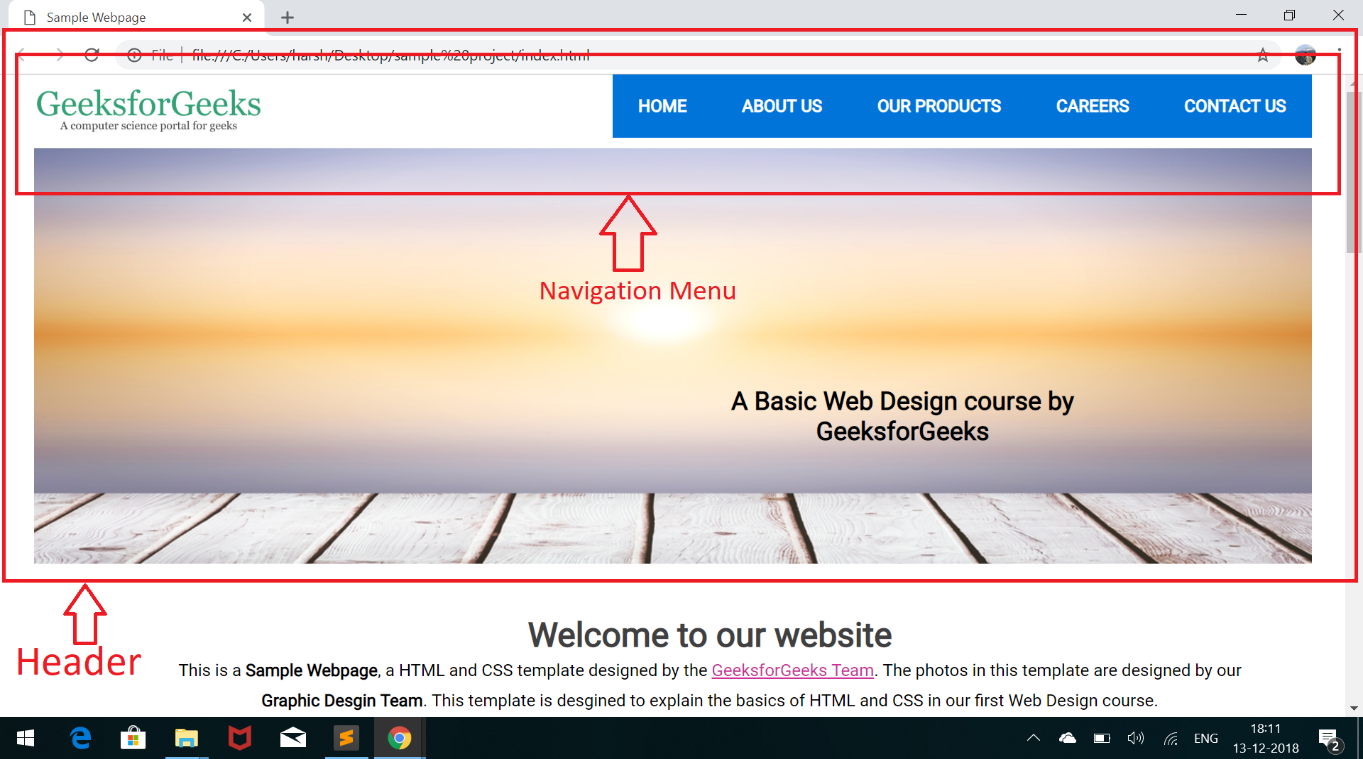
**<div>-** agrupa contenido **<section>-** agrupa contenido relacionado - agrupa contenido **<article>**independiente e independiente.

En fin el article va dentro del main con contenidos que no tienen relación con el contenido principal de la página.

## Facilite la navegacion del lector de pantalla con el encabezado landmark (Make Screen Reader Navigation easier with the header landmark).

Landmark: significa punto de referencia.

En esta sección utilizaremos la etiqueta ***header*** querepresenta un grupo de ayudas introductorias o de navegación. Puede contener algunos elementos de encabezado, así como también un logo, un formulario de búsqueda, un nombre de autor y otros componentes. Ejemplo:



Ejemplo de codigo :

<body>

  <header>

    <h1>Training with Camper Cat</h1>

  </header>

## Facilite la navegacion del lector de pantalla con nav landmark (Make Screen Reader Navigation Easier with the nav Landmark).

En esta clase usaremos el elemento ***nav*** querepresenta una sección de una página cuyo propósito es proporcionar enlaces de navegación, ya sea dentro del documento actual o a otros documentos. Ejemplos comunes de secciones de navegación son menús, tablas de contenido e índices. Ejemplo:

<nav>

      <ul>

        <li><a href="#stealth">Stealth &amp; Agility</a></li>

        <li><a href="#combat">Combat</a></li>

        <li><a href="#weapons">Weapons</a></li>

      </ul>

    </nav>

## Facilite la navegación del lector de pantalla con el pie de página landmark (Make Screen Reader Navigation Easier with the footer Landmark).

De manera similar a ***header*** y ***nav***, el elemento ***footer*** tiene una función de referencia incorporada que permite que los dispositivos de asistencia naveguen rápidamente hacia él. Se usa principalmente para contener información de derechos de autor o enlaces a documentos relacionados que generalmente se encuentran en la parte inferior de una página. Ejemplo:

<footer>&copy; 2018 Camper Cat</footer>

## Mejore la accesibilidad del contenido de audio con el elemento de audio (Improve Accessibilty of Audio Content With the audio Element).

A continuación agragaremos un audio a nuestro página. El elemento audio se usa para insertar contenido de audio en un documento HTML

La etiqueta <audio> </audio> lleva el atributo *controls*. Esto hace que se muestre un reproductor donde puedes pausar, subir el volumen, entre otras cosas y se puede utilizar con el teclado.

También lleva incluido **src** es la ubicación del recurso multimedia, requerido por los elementos [<audio>](https://developer.mozilla.org/es/docs/Web/HTML/Element/audio) y [<video>](https://developer.mozilla.org/es/docs/Web/HTML/Element/video).

**Src** La URL del audio que se va a insertar. Está sujeta a los [Controles de acceso HTTP](https://developer.mozilla.org/en-US/HTTP_access_control). Es opcional; en su lugar puedes usar el elemento [source](https://developer.mozilla.org/en-US/HTML/Element/Source) dentro del bloque de audio para especificar el audio que se va a insertar.

El **elemento HTML <source>**especifica recursos de medios múltiples para los elementos [<picture>](https://developer.mozilla.org/es/docs/Web/HTML/Element/picture), [<audio>](https://developer.mozilla.org/es/docs/Web/HTML/Element/audio), o [<video>](https://developer.mozilla.org/es/docs/Web/HTML/Element/video). Es un elemento vacío. Normalmente se utiliza para servir el mismo contenido multimedia en [varios formatos soportados por diferentes navegadores](https://developer.mozilla.org/en-US/docs/Web/Media/Formats).

Ejemplo:

<**p**><**audio controls**> <**source src**="https://s3.amazonaws.com/freecodecamp/screen-reader.mp3"

**type**="audio/mpeg">

A sound clip of Zersiax's screen reader in action.</**audio**></**p**>

## Mejore la accesibilidad del gráfico con el elemento de figura (Improve Chart Accessibility with the figure Element).

El elemento ***figure*** cuando se usa con el elemento ***figcaption*** crean una representación visual (como una imagen, diagrama o gráfico) junto con su título.

***Figcaption*** es para darte un texto extra o un título a la imagen diagrama grafico que insertaremos que siempre sera visible a diferencia de ***alt*** sirve para describir el contenido de una imagen. Un atributo alt ayuda en los casos en que la imagen no se carga o un usuario no puede verla.

***Figure*** no daña el flujo del contenido es decir que no al usa el elemento figure no estará entre el contenido que agregamos o está por arriba, en el medio o al final pero nunca entre las línea ejemplo.

<figure>

        <figcaption>Breakdown per week of time to spend training in stealth, combat, and weapons.</figcaption></figure>

## Mejore la accesibilidad del campo de formulario con el elemento de etiqueta (Improve Form Field Accessibility With the label Element).

A continuación se utilizara un elemento ***label*** que sirve como etiqueta y sirve principalmente, para mostrar información, como por ejemplo, exhibir un título, el nombre de una casilla, el elemento label utiliza la etiqueta For es el único atributo que suele utilizarse en la etiqueta <label> y sirve para indicar el identificador (atributo id) del campo de formulario para el que esta etiqueta hace de título.es decir el for llevara un nombre que ese mismo nombre se usara en el imput con un id esta los vinculara ejemplo:

<label for="email">Email:</label>

      <input type="text" id="email" name="email">

Otra función que te daran estos elementos es que cuando cliques el nombre del imput activara la casilla.

## Envuelva los botones de radio en un elemento de conjunto de campo para una mejor accesibilidad (Wrap Radio Buttons in a fieldset Element for better Accessibility).

A Continuación utilizaremos la etiqueta ***fieldset*** que sirve para rodear toda la agrupación de botones de opción y se le asigna la etiqueta ***legend*** que sirve para proporcionar una descripción a la agrupación o darle un título a la agrupación de botones. Ejemplo:

<fieldset>

        <legend>What level ninja are you?</legend>

## Agregar un selector de fecha accesible (Add an accesible Date Piker).

En esta clase seguiremos usando ***input*** pero esta vez con un ***type*** ***date*** lo cual creara un campo y dentro del campo un formato de fecha dia/mes/año/. Ejemplo

<input type="date" id="pickdate" name="date">.

## Estandarice las horas con el atributo de fecha y hora de HTML5 (Standardize Times With the HTML5 datetime Attribute).

El **elemento HTML <time>** representa un periodo específico en el tiempo. Puede incluir el atributo datetime para convertir las fechas en un formato interno legible por un ordenador, permitiendo mejores resultados en los motores de búsqueda o características personalizadas como recordatorios. Ejemplo:

**<p>** Master Camper Cat ofició el partido de jaula entre Goro y Scorpion **<time datetime** = "2013-02-13"> el miércoles pasado **</time>**, que terminó en empate. **</p>**

Preguntas : para que sirve el element time?

## Haga que los elementos solo sean visibles para un lector de pantalla mediante CSS personalizado (Make Elements Only Visible to a Screen Reader by Using Custom Css).

El objetivo de esta clase es ocultar visualmente el contenido (una tabla) usando css y solo se obtenga la información usando lectores de pantalla. Ejemplo

.sr-only {

    position:absolute;

    left:-10000 ;

    width:1px ;

    height:1px ;

    top: auto;

    overflow: hidden;

  }

Explicación de clase :S

## Mejore la legibilidad con texto de alto contraste (Improve Readability with High Contrast tet).

En esta sección aprenderemos sobre las Pautas de Accesibilidad al Contenido Web que explican cómo hacer que el contenido Web sea accesible para personas con discapacidad.

Que te aconsejan que te dice que el color de las letras deben ir en contraste con el color del fondo y para esto hay una calculado.

Existe una norma que recomiendan al menos una relación de contraste de 4.5 a 1 para texto normal. Ejemplo:

color: #636363; gris oscuro

    background-color: #FFF; gris claro

esto creo un relación de contraste de 6:1.

## Evite los problemas de daltonismo utilizando suficiente contraste (Avoid Colorblindness Issues by Using Sufficient Contrast).

En esta sección se cambió la luminosidad de los colores usando hsl() para crear un contraste de 5.9:1 ejemplo:

color: hsl(0, 55%, 15%); la luminosidad estaba en 20%

background-color: hsl(120, 25%, 55%); la luminosidad esta en 35%

## Evite los problemas de daltonismo eligiendo con cuidado colores que transmitan información (Avoid Colorblindness Issues by Carefully Choosing colors that Convey Information).

Otra regla que encontramos en esta seccion es que los colores cercanos pueden considerarse vecinos en la *rueda de colores*, lo cual debería evitarse como combinaciones para transmitir información.

**Ejemplo incorrecto:**

color: #33FF33;

    background-color: #FFFF33;

**Ejemplo correcto:**

color: #003366;

background-color: #FFFF33;

## Dar significadp a los enlaces mediante el uso de texto descriptive del enlace (Give Links by Using Descriptive link Text).

Lo que se aprendió en esta clase es que para los lectores de pantallas no le es útil leer un enlace que diga “haz clic aquí” o “leer más” por lo cual utilizamos un texto más descriptivo dentro de las etiqueta de anclaje. Ejemplo:

…But chin up, fellow fighters, our time for victory may soon be near.Click here for **<a href="">**information about batteries**</a>**</p>.

## Haga que los enlaces sean navegables con claves de acceso HTML (Make Links Navigable with HTML Access Keys).

Es esta clase utilizaremos en altributo ***accesskay*** que sirve para especificar una tecla del teclado de método abreviado para activar o enfocar un elemento.

Para los usuarios que solo usan el teclado puede ser un método eficiente. Ejemplo

Agregaremos este atributo a los títulos o cualquier otro elemento, de manera que los usuarios puedan seleccionar los títulos botones y enlaces con el teclado, y señalando la tecla seleccionada. También es importante saber el navegar que se está utilizando ya que todos no se manejan de la misma manera el teclado.

(¿Cómo el usuario sabes cuál es la tecla?).

Muchas veces se usa el tabulador o la tecla tab.

## Use tabide para agregar el enfoque del teclado a un elemento (Use tabinde to Add Keyboard Focus to an Element).

**tabindex**indica si su elemento puede ser enfocado, y si participa en la navegación secuencial del teclado (usualmente con la tecla *Tab*, de ahí el nombre). Acepta un entero como valor, con diferentes resultados que dependen de dicho valor:

* un *valor negativo*(usualmente tabindex="-1")significa que el elemento debe ser enfocado, pero no debe de ser accesible a través de la navegación secuencial del teclado.
* tabindex="0" significa que el elemento debe ser enfocado y ser accesible a través de la navegación secuencial del teclado,

ejemplo:

<p tabindex="0"> esta hara que cuando usemos el tabulador se selecciones el elemento párrafo.

## Use tabindex para especificar el orden de enfoque del teclado para varios elementos (Use tabindex to Specify the Order of Keyboard Focus for Several Elements).

El atributo ***tabindex*** específica el orden de tabulación exacto de los elementos. Esto se logra cuando el valor del atributo se estable en un numero positivo de 1 en adelante. Ejemplo:

Cuando usamos el atributo tabindex=”1” en el elemento de título lo primero que el tabulador marcara será el título porque allí esta ubicado nuestro ***tabindex*** con el número 1, luego seguirá donde se ubiqué el 2 y el 3 etc.

Cuando el orden de tabulación se establece de esta manera, anula el orden predeterminado (que usa la fuente HTML). Ejemplo:

<div **tabindex="1">**I get keyboard focus, and I get it first!</div>

<**div tabindex="2">**I get keyboard focus, and I get it second!</div>

# Principio de Diseño web receptivo

Hay muchos dispositivos que pueden acceder a la web y vienen en todas las formas y tamaños. El diseño web receptivo es la práctica de diseñar sitios web flexibles que pueden responder a diferentes tamaños de pantalla, orientaciones y resoluciones.

En este curso, aprenderá a usar CSS para hacer que sus páginas web se vean bien, sin importar en qué dispositivo se vean.

## Crear una consulta de medios (Create a Media Query)

En esta sección recordamos para que sirven las ***media query*** sirven para cambiar el tamaño o el ancho de un elemento después de haberse declarado. es decir son la forma de poder especificar diferentes **estilos** css para diferentes **tamaño** que tenga el navegador en este caso sobre cualquier dispositivo que se encuentre ya sea una Tablet, un celular o una pc es decir lo media query limitan los estilos css hasta cierto punto en donde cambiaran de acuerdo a las propiedades css que nosotros le definamos. Ejemplo:

<style>

  p {

    font-size: 20px;

  }

@media (max-height: 800px){

P{Font-size:10px}}

</style>

Esto hara que las letras tendrán un tamaño de 10px cuando la altura del dispositivo sea menor o igual a 800px.

## Hacer una imagen receptiva (Make an Image Responsive).

Hacer que las imágenes respondan con CSS es realmente muy simple. Solo necesita agregar estas propiedades a una imagen:

img {

max-width: 100%;

height: auto;

}

El ***max-widthde 100%*** se asegurará de que la imagen nunca sea más ancha que el contenedor en el que se encuentra, y el***height*** de ***auto*** hará que la imagen mantenga su relación de aspecto original.

## Utilice una imagen Retina para pantallas de mayor resolución (Use a Retina for Higher Resolution Displays).

En esta sección veremos la densidad de pixeles que es un aspecto que podría ser diferente en un dispositivo de otros es decir nos indica la calidad de una pantalla y es fundamental a la hora de realizar un diseño para móviles, esta densidad se conoce como pixeles por pulgadas (PPI) o puntos por pulgadas (DPI).

La forma más sencilla de hacer que sus imágenes aparezcan correctamente en pantallas de alta resolución, como la "pantalla retina" de MacBook Pro, es definir sus valores ***width*** y***height*** a la mitad de lo que es el archivo original.

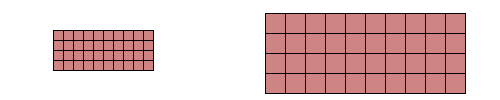
A la hora de diseñar para múltiples dispositivos, tanto en dispositivos iOS como Android, corremos el peligro de no tener en cuenta las a veces enormes diferencias entre estos y que lo que en un dispositivo se ve bien, en otro aparezca muy pequeño o muy grande. En este artículo, intento abordar de una manera global el diseño tanto en apps como en html para dispositivos móviles, dado que en ambos casos tenemos que tener en cuenta varios aspectos para que la experiencia usuario sea óptima. Para ello es fundamental conocer varios conceptos que vamos a tratar a continuación.

Para empezar, sabemos que **un pixel** es un punto que emite un único color o escala de grises en una pantalla. Es obvio que cuanto mayor sea el número de pixels, más detalle y nitidez tendrá la imagen que estemos viendo, aquí es donde entra en juego la densidad de pixel.

Las pantallas se miden en **pulgadas** (1 **pulgada** = 2,54 centímetros), y cuando damos la medida de una pantalla, lo hacemos tomando como referencia su diagonal, cuando nos dicen que una pantalla es de 25 pulgadas, nos están diciendo que su diagonal mide 25 pulgadas, es decir, 25×2,54 cm

La **densidad de píxeles** es la cantidad de pixels que hay en una pulgada de una pantalla. Cuanto mayor sea este número, más nítida será la imagen y con más detalle. Esta cantidad de pixels que hay en una pulgada, se mide en pixels por pulgada **ppp** o en **ppi**(pixels per inch) o **dpi**  (Dots per Inch). Con lo cual una**pantalla de alta densidad** será aquella que más ppp tenga, esto significa que si hablamos de tamaños, cuando hacemos un botón por ejemplo de 100x60px, se verá más grande en una pantalla de baja densidad que en una de alta densidad.  
Para calcular la **densidad de una pantalla**; densidad de pantalla = **anchura de la pantalla en pixels / anchura de la pantalla en pulgadas**

En el siguiente gráfico, podéis ver dos botones, los dos con las mismas dimensiones, 10 x 4px. El de la izquierda sería el resultado de mostrarlo en una pantalla de más densidad que el de la derecha, concrétamente 4 veces más pixels por pulgada que la de la derecha.



## Hacer que la tipografía sea receptiva (Make Typgraphy Respsive).

En lugar de usar ***em*** o ***px*** cambiar el tamaño del texto, puede usar unidades de ventana gráfica para una tipografía receptiva. Las unidades de la ventana gráfica, como los porcentajes, son unidades relativas, pero se basan en elementos diferentes. Las unidades de la ventana gráfica son relativas a las dimensiones de la ventana gráfica (ancho o alto) de un dispositivo, y los porcentajes son relativos al tamaño del elemento contenedor principal.

Las cuatro unidades de ventana gráfica diferentes son:

* vw(ancho de la ventana gráfica): 10 vw sería el 10% del ancho de la ventana gráfica.
* vh(altura de la ventana gráfica): 3vh sería el 3% de la altura de la ventana gráfica.
* vmin(ventana gráfica mínima): 70vminsería el 70% de la dimensión más pequeña de la ventana gráfica (alto o ancho).
* vmax(ventana gráfica máxima): 100vmaxsería el 100% de la dimensión más grande de la ventana gráfica (alto o ancho).

Ejemplo:

<style>

h2{width: 80vw; }

p{width: 75vmin;}

</style>

Que es la ventana grafica?

La ventana gráfica varía con el dispositivo, y será más pequeño en un teléfono móvil que en una pantalla de ordenador.

Antes de las tabletas y los teléfonos móviles, las páginas web se han diseñado sólo para las pantallas de ordenador, y era común para las páginas web que tengan un diseño estático y un tamaño fijo.

Entonces, cuando comenzamos a navegar por internet usando las tabletas y los teléfonos móviles, páginas web tamaño fijo eran demasiado grandes para caber la ventana gráfica. Para solucionar este problema, los navegadores de los dispositivos de escala reducida toda la página Web para ajustarse a la pantalla.

# CSS Flexbx

Flexbox es un método de diseño potente y bien compatible que se introdujo con la última versión de CSS, CSS3. Con flexbox, es fácil centrar elementos en la página y crear interfaces de usuario dinámicas que se encogen y expanden automáticamente.

En este curso, aprenderá los fundamentos de flexbox y diseños dinámicos al crear una tarjeta de Twitter.

## Pantalla de uso: flexion para colocar dos cajas (Use Dispaly: flex to Position two boxes).

La propiedad **CSS display** especifica si un elemento es tratado como block or inline (bloque o en linea),

***Flex:*** aporta una magnífica solución para todos nuestros desarrollos “responsive”. Lo que nos permite es crear un conjunto de elementos flexibles que se adaptan automáticamente a su contenedor y con el que podemos controlar parámetros tales como la alineación, dirección (horizontal/vertical), ajuste de la fila según tamaños.

Ejemplo:

<style>

  #box-container {

    height: 500px;

    display: flex;

## agregue Flex Superpowers al tweet embed (Add flex superpowers to the tweet Embed).

En esta seccion se utilize la propiedad ***Display: flex;*** varias veces para ajustar la posición de los elementos de un tweet.

## Utilice la propiedad flexible de dirección flexible para hacer una fila (Use the flex-direction Property to Make a Row).

Aparte de usar ***display:flex;*** para alinear también podemos usar la propiedad ***flex-direction*** y configurarla si queremos que se ordenen en filas o columnas.

***Flex-direction***: trabaja con los atributos de ***row-reverse*** esto alineara los elementos horizontalmento pero en reversa como lo dice su nombre y el atributo ***column-reverse***: esto crea una fila vertical.

## Aplicar la propiedad de dirección flexible para crear filas en el Tweet incrustado (Apply the flex-direction Property to Create Rows in the Tweet Embed).

En esta clase solo insertaremos la propiedad ***flex-direction: row;*** al header y al footer.

## Utilice la propiedad de dirección flexible para hacer una columna (Use the flex-direction Property to Make a Column).

En esta clase solo insertaremos la propiedad ***flex-direction: column;*** para hacer una columna.

## Aplicar la propiedad de dirección flexible para crear una columna en el tweet insertar (Apply the flex-direction Property to Create a Column in the Tweet Embed).

Se utilizó la propiedad ***flex-direction: column;*** para colocar algunas palabras en columna.

## Alinear elementos usando la propiedad Justify-content (Align Elements Using the justify-content Property).

En esta sección aprenderemos como usar la propiedad ***Justify-content***  que significa justificar el contenido o alinearlos. Es similar a la función de que te da Word para alinear los texto puede ser a la derecha a la izquierda centrar y justificar pero aquí encontraremos más opciones. Ejemplo.

* **flex-start**: alinea los elementos al inicio del contenedor flexible. Para una fila, esto empuja los artículos a la izquierda del contenedor. Para una columna, esto empuja los artículos a la parte superior del contenedor. Esta es la alineación predeterminada si no justify-contentse especifica.
* **flex-end:** alinea los elementos al final del contenedor flexible. Para una fila, esto empuja los elementos a la derecha del contenedor. Para una columna, esto empuja los artículos al fondo del contenedor.
* **space-between:** alinea los elementos con el centro del eje principal, con espacio adicional entre los elementos. Los primeros y últimos elementos se empujan hasta el borde del contenedor flexible. Por ejemplo, en una fila, el primer artículo está contra el lado izquierdo del contenedor, el último artículo está contra el lado derecho del contenedor, luego el espacio restante se distribuye uniformemente entre los otros artículos.
* **space-around**: similar space-betweenpero el primer y último artículo no están bloqueados en los bordes del contenedor, el espacio se distribuye alrededor de todos los artículos con medio espacio en cada extremo del contenedor flexible.
* **space-evenly**: Distribuye el espacio de manera uniforme entre los elementos flexibles con un espacio completo en cada extremo del contenedor flexible

## Utilice la propiedad Justify-content en el twee Embed (Use the Justify-content Propery in the tweet Embed).

A continuación usaremos la propiedad ***justify-content*** con cualquier atributo antes mencionado para insertarlo en un tweet. Ejemplo.

 header .profile-name {

    display: flex;

    flex-direction: column;

**justify-content: flex-end;**

    margin-left: 10px;

## Alinear los elementos usando la propiedad align-items (Align Elements Using the align-items Property).

CSS ofrece la propiedad ***align-items*** de alinear elementos flexibles a lo largo del eje transversal. Para una fila, le dice a CSS cómo empujar los elementos en toda la fila hacia arriba o hacia abajo dentro del contenedor. Y para una columna, cómo empujar todos los elementos hacia la izquierda o hacia la derecha dentro del contenedor.

Es decir si tenemos una palabra dentro de un colomna la podemos colocar en el principio a medio o al final de la columna sin que se salga de misma usando estos atributos:

* **flex-start:** alinea los elementos al inicio del contenedor flexible. Para las filas, esto alinea los artículos con la parte superior del contenedor. Para las columnas, esto alinea los elementos a la izquierda del contenedor.
* **flex-end:** alinea los elementos al final del contenedor flexible. Para las filas, esto alinea los artículos con la parte inferior del contenedor. Para las columnas, esto alinea los elementos a la derecha del contenedor.
* center: alinea los elementos al centro. Para las filas, esto alinea los elementos verticalmente (el mismo espacio por encima y por debajo de los elementos). Para las columnas, esto las alinea horizontalmente (el mismo espacio a la izquierda y a la derecha de los elementos).
* **stretch:** estire los artículos para llenar el contenedor flexible. Por ejemplo, los elementos de las filas se estiran para llenar el contenedor flexible de arriba a abajo. Este es el valor predeterminado si no align-itemsse especifica ningún valor.
* **baseline:** alinea los elementos con sus líneas de base. La línea de base es un concepto de texto, considérelo como la línea en la que se asientan las letras.

## Utilice la propiedad aling-items en el Tweet embed (Use the align-items Propery in the Tweet Embed).

A continuación usaremos la propiedad ***align-items*** con el atributo ***center*** para insertarlo en un tweet. Ejemplo.

  header .follow-btn {

    display: flex;

**align-items: center;**

    margin: 0 0 0 auto;

## Utilice la propiedad flex-wrap para envlver una fila o clumna (Use the flex-wrap Property to Wrap a Row or Column).

la propiedad ***flex-wrap*** le dice a CSS que ajuste los elementos. Esto significa que los elementos adicionales se mueven a una nueva fila o columna.

El punto de ruptura del lugar donde se envuelve depende del tamaño de los artículos y del tamaño del contenedor ejemplo: si todos los elementos tienen la misma medida su tamaño se ajustara para estar en una línea, pero si agregamos ***flex-wrap*** con el atributo ***wrap*** los elementos saltaran a otra línea y obtendrán su tamaña original decalrado.

* **nowrap:** esta es la configuración predeterminada y no envuelve elementos.
* **wrap:** envuelve los elementos en varias líneas de arriba hacia abajo si están en filas y de izquierda a derecha si están en columnas.
* **wrap-reverse**: envuelve los elementos en varias líneas de abajo hacia arriba si están en filas y de derecha a izquierda si están en columnas.

## Utilice la propiedad flex-shrink para encoger elementos (Use the flex-shrink Property to Shrink Items).

La propiedad ***flex-shrink*** sirve para encoger elementos y toma números como valores, cuanto mayor sea sea el número, más se encogerá en comparación con los otros elementos del contenedor. Ejemplo:

 Si un elemento tiene un **flex-shrink** valor de 1y el otro tiene un **flex-shrink** valor de 3, el que tiene el valor de 3 se reducirá tres veces más que el otro.

## Utilice la propiedad flex-grow para expandir elementos (Use the flex-grow Property to Expand Items).

La propiedad ***flex-gorw*** sirve para crecer o expandir elementos y toma números como valores, cuanto mayor sea el número, más crecerá en comparación con los otros elementos del contenedor. Ejemplo:

 Si un elemento tiene un **flex-grow** valor de 1 y el otro tiene un **flex-grow** valor de 3, el que tiene el valor de 3 crecerá tres veces más que el otro.

## Utilice la propiedad de base flexible para establecer el tamaño inicial de un artículo (Use the Flex-basis property to Set the Initial Size f an Item).

La propiedad flex-basis sirve para establecer tamaño inicial de un elemento. Las unidades usadas por la propiedad ***flex-basis*** son las misma que se usan en las propiedades de tamaño (px, em,%, etc).

## Usar la propiedad taquigrafía flex (Use the flex Shorthand Property).

En esta sección a aprenderemos un atajo para configurar varias propiedades ***flex*** a la vez. Que son las propiedades ***flex-grow***(expandir o crecer), ***flex-shrink***(encoger) y ***flex-basis***(base flexible). Ejemplo:

**flex: 1 0 10px;**

Se ajusta el elemento **flex-grow: 1**;, **flex-shrink: 0**; y **flex-basis: 10px**;. Y su configuradcion predeterminada es ***flex: 0 1 auto;***

## Utilice la propiedad de orden para reorganizar los elementos (Use the order Property t Rearrange Items).

La propiedad [CSS](https://developer.mozilla.org/es/docs/Web/CSS) **order** especifica el orden utilizado para disponer los elementos en su contenedor flexible. Los elementos estarán dispuestos en orden ascendente según el valor de order. Los elementos con el mismo valor de order se dispondrán en el orden en el cual aparecen en el código fuente.

**Nota**: order sólo pretende afectar el orden visual de los elementos y no su orden lógico u orden de tabulación

**Nota:** Si el elemento no es un elemento flexible, la order propiedad no tiene ningún efecto

Se puede decir que en el orden que usemos la propiedad **order** y su valoración así será el orden que lo veremos en nuestra página web.

## Usar la propiedad align-self (Use the align-self Property).

La propiedad ***align-self*** le permite ajustar la alineación de cada elemento individualmente, en lugar de configurarlos todos a la vez.

***Align-self*** acepta los mismos valores que ***align-items*** y anulará cualquier valor establecido por la  propiedad ***align-items.***

Otra definición que podemos estables es que La propiedad align-self especifica la alineación del elemento seleccionado dentro del contenedor flexible.

**Nota:** La align-self propiedad anula la propiedad [align-items](https://www.w3schools.com/cssref/css3_pr_align-items.asp) del contenedor flexible.

# Cuadricula CSS

La cuadrícula CSS es un estándar más nuevo que facilita la creación de diseños complejos con capacidad de respuesta. Funciona convirtiendo un elemento HTML en una cuadrícula y le permite colocar elementos secundarios en cualquier lugar dentro.

En este curso, aprenderá los fundamentos de la cuadrícula CSS mediante la creación de diferentes diseños complejos, incluido un blog.

## Crea tu primera cuadricula CSS (Create Your First CSS Grid)