Video 1 Presentación

Video 2
El desarrollo web tiene que ver con todo lo que percibimos a través del navegador: páginas, aplicaciones y sitios web como Facebook, Instagram y Twitter.

Para comprender cómo funciona internet necesitamos conocer tres grandes conceptos:

- **Clients** (clientes): son los dispositivos a través de los cuales accedemos a los sitios web: un computador desktop, un teléfono, una laptop, etc.
- **Internet**: es toda la red formada por servidores y clientes que proveen y consumen contenidos web y se comunican entre sí a nivel global.
- **Server** (servidor): es un computador, ubicado en alguna parte de la red, que está prendido todo el tiempo, en el que se alojan o almacenan sitios web y sus recursos y al cual se accede a través de nombres de dominio (.com, .net, .pe, etc.). También se les conoce como hosting.

Profesiones dentro del Desarrollo Web:

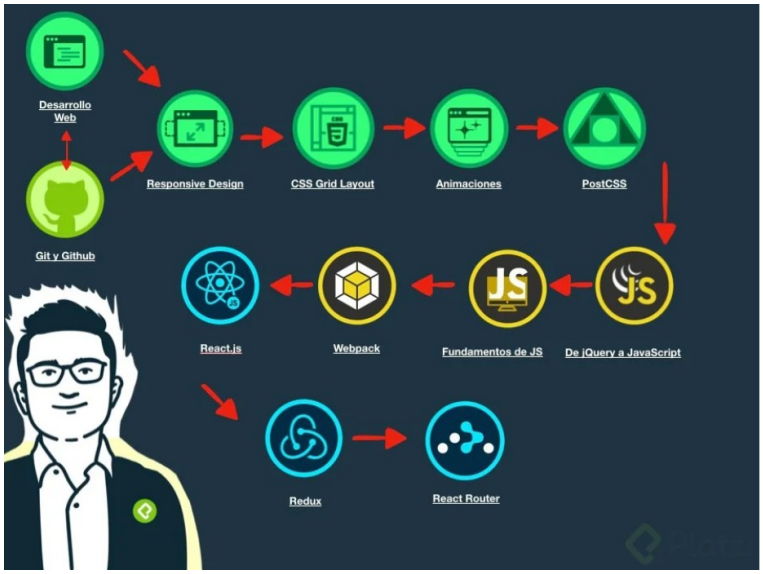
- **Frontend**: son los encargados de cuidar toda la apariencia y experiencia de usuario. Su misión es pasar todo el diseño gráfico de un sitio o aplicación web, a código, y proveer toda la interactividad a los clientes. Esta rama se puede subdividir en algunas especializaciones como: Arquitecto Frontend (entro de las ramas e interfaces), Desarrollador JavaScript (frontend, temas de datos y del clientes), etc.
- **Backend**: resguardan los datos y la seguridad de las aplicaciones y sitios web. Su misión es crear y mantener toda la parte del sitio web que sucede en los servidores. Pueden especializarse aún más en: SysAdmis, DevOps, Desarrollador JavaScript (backend), entre otros.

Las tres tecnologías básicas que debe conocer y manejar un **Frontend** son:

- **HTML (Hypertext Markup Language)**: es el lenguaje de marcado para hacer websites.
- **CSS (Cascade Style Sheet)**: hojas de estilos cascada, el diseño hecho código.
- **JavaScript (El lenguaje de programación)**: es el único lenguaje que funciona actualmente dentro de los navegadores de manera nativa."

El curso de desarrollo web sera muy bueno en el futuro.

La ruta que se tiene que seguir para la programación



Video 3 (curso a la par de Git y Github)
plan de estudios para el día hace 2 años.
Responsive Design nos lleva el curso a utilizarlo en cualquier plataforma.

CSS Grid Layout, una manera mas sencilla de organizar nuestra información.

Curso de animaciones para la web.

PostCSS organiza nuestro código y aprender nuevas sintasys.

De jQuery a JavaScript, sitio web dinámico, conectado API.

Fundamento de JavaScript aprender, sobre las bases técnicas del lenguaje.

Webpack, nos permite que JS sea nuestra principal aplicación.

React. Son las formas escalables del curso.

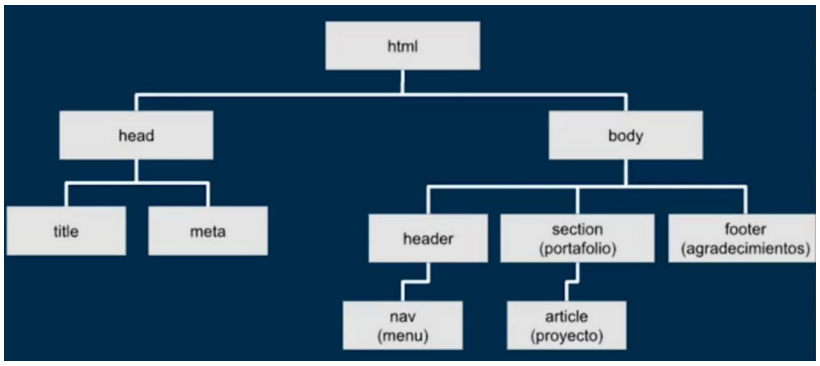
Video 4
Tener un portafolio personal de Fidel.
Los editores de codigo es, SublimeText, VisualStudioCode y Atom.
Instale el seti monoki y se cambio en Code

Video 5
División de nuestro sitio como fronted. Ver secciones principales, luego secciones dentro de secciones.

Aqui, el ejemplo que nos muestra el curso.
DOM es el acrónimo de Document Object Model o Modelo de documento que se carga en el navegador web, y es la manera en que se **representa** el contenido del documento, de manera similar a un árbol de nodos.

A continuación, un ejemplo sencillo de la estructura del DOM:

- html
  - head
    - title
    - meta
  - body
    - header
      - nav
    - section
      - article
    - footer



El árbol de nodos sencillo y se puede ver complejo o sencillo.

Video 6 Etiquetas.
Las etiquetas son la representación básica de la información en un documento html. Sirven para crear y organizar el contenido.
La sintaxis general de una etiqueta es:

```
<nombre>contenido</nombre>
```

Hay ciertas etiquetas que tienen una sintaxis diferente, ya que se cierran en sí mismas; es decir, no tienen etiqueta de cierre:

```
<nombre />
```

Algunas de las etiquetas más conocidas y usadas son:

Etiquetas de cabecera (head):

- **`doctype`**: indica al navegador el tipo de documento que se está mostrando.
- **`html`**: es la etiqueta que envuelve todo el documento
- **`head`**: es la cabecera del documento y contiene sub etiquetas que describen al documento o incluyen recursos adicionales.

Etiquetas del cuerpo del documento (body):

- **`article`**: diferencia partes del contenido que pueden vivir por sí mismas.
- **`nav`**: para hacer menús de navegación.
- **`aside`**: contenido menos relevante, como publicidad, etc.
- **`section`**: sirve para diferenciar las secciones principales del contenido.
- **`header`**: cabecera del documento.
- **`footer`**: pie de página del documento.
- **`h1 - h6`**: títulos de nuestro sitio web.
- **`table`**: tablas de contenidos, similar a la estructura de las hojas de calculo.
- **`ul y ol`**: listas de items.
- **`div`**: cualquier división para organizar el contenido.
- **p:** etiqueta para un párrafo.
- **img:** etiqueta que se cierran así misma.

Dentro de las etiquetas pueden venir representados los atributos.

- **Src:**

Y un enlace donde puedo encontrar todas las etiqutas. html5doctor.com

Video 7
En este curso vamos a realizar mi portafolio.

Head line: Hola! soy Jesús Fidel Campa Miranda, Desarrollador y Maestros en Ciencias con pasión por la tecnología.

El proyecto que desarrollaremos en este curso se trata de un Portafolio Personal, incluirá una cabecera, navegación, un hero (área visual con información destacada), un área de proyectos y otra para eventos, y finalmente un pié de página con un formulario de contacto y enlaces a redes sociales.

El archivo **index.html** es el archivo que el navegador abre por defecto al acceder a un directorio en un servidor web.

La estructura básica de un archivo html es la siguiente:

```
<html>
  <head>
    <title> Título de la página title>
  head>
  <body>
    <header> Cabecera del contenido header>
    <section> Sección principal section>
    <section> Otra sección section>
    <footer> Pié de página del documento footer>
  body>
html>
```

un comando ALT+SHIFT y la flecha para abajo de suplica la linea
item de lista li
listas ordenas y desordenadas ol y ul

portafolio, experiencia, trabajemos juntos

Video 8
Seccion principal coloco mi presentación.
Pongo, fechas algunos h1 y parrafos
Github es un sitio que te ayuda a trabajar en repositorios desde la web, para trabajos colaborativos en la industria, utilizado para aumentar las habilidades técnicas y profesionales, con el presento la mejor manera para enseñar el funcionamiento de un curso de diseño web aprendido en platzi.

P cursos que tienen que ver con tecnología, llevados de la la plataforma de capacitación de platzi /p

Video 9
Las etiquetas tags no funcionan solas de html, pueden recibir ciertos atributos.
Empezamos con el primer img.

Los atributos son valores agregados a las etiquetas (tags) html que extienden su habilidad o funcionalidad con información específica.

A continuación, un ejemplo de los atributos más comunes y usados en algunas etiquetas:

Para **img**:

- **src**: específica la *ruta* de la imagen que será mostrada a través de esta etiqueta. La ruta puede ser *absoluta* (cunado especifica una dirección exacta, incluyendo el prefijo *http(s)*) o *relativa* (cuando la referencia a la ubicación de la imagen parte de la ubicación del archivo actual).
- **alt**: indica un texto alternativo que será mostrado en lugar de la imagen cuando ésta no pueda ser mostrada.
- **width**: ancho de la imagen en pixeles.

- **height**: alto de la imagen en pixeles.

Para **link**, en la cabecera *head* del documento:

- **rel**: indica la relación del recurso con el contenido.
- **type**: indica el tipo de recurso / formato.
- **href**: indica la ubicación (url) del recurso enlazado.

Para **meta**, también en la cabecera *head* del documento:

- **charset**: indica la tabla de caracteres (utf-8 para caracteres latinos) usada en el documento.

Para **a**:

- **href**: la ubicación o ruta a la que enlaza esta etiqueta de ancla. En el caso de querer enlazar a elementos que se encuentran dentro del mismo documento, este atributo debe indicar el valor del atributo ““id”” de ese elemento destino del enlace.

La sección principal que mas destaca dentro de la industria le pusieron el nombre de hero.

VIDEO 10
Formularios en HTML
Los Formularios en html son unidades de información que nos permiten recolectar información para enviarlos al propietario del website o a un servicio externo. Esta formado por dos partes o contextos: una parte donde se hace el ingreso y modelación de esos datos (en el frontend), y otra parte que se encarga de enviar, procesar y almacenar esos datos (en el backend).

Los formularios se crean con la etiqueta **form**. El atributo principal de un formulario es *action*, ya que contiene la ruta a la que serán enviados los datos recolectados.

Hay diversos elementos html que permiten la captura o recolección de datos, aunque generalmente se usan los elementos creados con la etiqueta *input*. Los inputs también sirven para crear botones, aunque existe una etiqueta especial para ésto llamada *button*… El atributo principal de los inputs es *type*, que indica el tipo de comportamiento o dato que se espera recibir.

Los elementos creados con la etiqueta *label* muestran un texto que se puede asociar con un input para darle mayor significado al campo, principalmente cuando no se usa el atributo *placeholder*.

Espacios en blanco donde se tendria que poner la infomación.


```
<section>
    <form action="/suscripcion/">
        <label for="">Déjame tu email</label>
    <input type="text"placeholder="Déjame tu email" >
    <input type="submit" value="Enviar">
    <button>Enviar</button>
    </form>
```

VIDEO 11
Enlazar nuestro website.

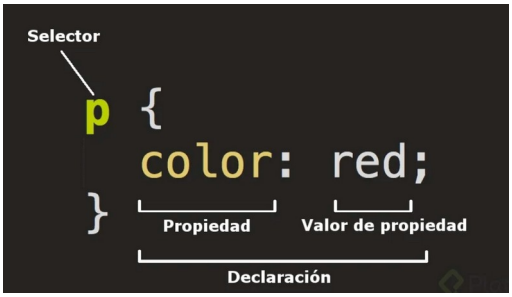
Video 12
Formas de agregar estilos al HTML
Hay tres opciones para incluir estilos que definan la apariencia de tu html:

- **Estilos en línea**: se definen directamente en el elemento html que quieres estilizar, se agregan con el atributo **style**.
- **Estilos con el tag Style**: regularmente este tag se incluye dentro de la etiqueta **head** del html.
- **Estilos enlazados desde un archivo css externo**: utilizando la etiqueta **link** que nos permite enlazar recursos externos.

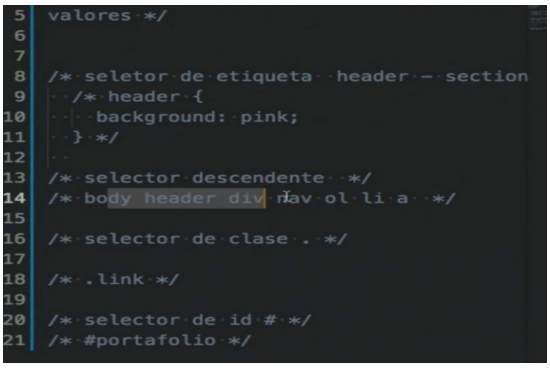
A **CSS**, se le llama **hojas de estilos en cascada** porque los estilos que se definen para una página, se van aplicando de arriba hacia abajo, y de lo más general a lo más particular, teniendo prioridad lo más particular. Esto es, los estilos que prevalecen son los que han sido definidos **en línea**, luego los que fueron definidos mediante la etiqueta **style** en la cabeza o cuerpo del html, y por último los estilos definidos en archivos externos enlazados con la etiqueta **link**. Esta prioridad se puede alterar al usar el modificador **!important"" en la definición de algún estilo en particular, aunque esto no es recomendado.

Un estilo inline tiene mas peso que un estilo en la etiqueta style.
Utilizas un !important para que sea el principal en tu hoja de estilo.

Video 13
Reglas, selectores, declaraciones, propiedades y valores.
Un selector en css.
1. Selector de etiquetas
2. Selector de hijos (descendente)
Elemento hijo de un otro elemento.
3. Selector de clase. Y el que mas se estaría utilizando. Es un atributo dentro de html



Las clases se pueden multiplicar a diferencia de los id que son únicos dentro de todo.



Video 14
Los navegadores incluyen estilos predeterminados para cada elemento html. Esto significa que aún cuando no hayas definido o asignado ningún estilo a tus etiquetas, éstas tendrán una apariencia particular que es muy similar en todos los navegadores, aunque no necesariamente idéntica.
Se puede editar.

Video 15
Para aplicar estilos a los componentes html, lo más común y recomendable es hacerlo a través de **clases** que se asignan al elemento html mediante el atributo **class**.

Un elemento html puede tener varias clases, se deben indicar en el mismo atributo class pero separadas por un espacio en blanco.

Al escoger los nombres de clases, debemos tener en cuenta que se puedieran aplicar a muchos elementos, o a elementos particulares, así que la claridad y precisión en su identificación facilitará la contextualización y mantenibilidad en el futuro.

Algunos de los estándares más usados para la identificación de clases son:

- OOCSS
- BEM
- Component CSS

Las clases son los nombres semánticos que tienen los elementos.
Notas cuando se necesita contener o envolver otra clase Container o Writter.

Video 16
Vamos a empezar a escribir CSS saber las unidades de medidas y los colores

Hay varias unidades de medida con las que se puede trabajar en CSS: %, em, rem, px, pt, fr, vw, vh
Las medidas más comunes y utilizadas son los pixeles. Un **píxel** es la menor unidad homogénea en color que forma parte de una imagen digital. Es la unidad más práctica y fácil de utilizar y manipular, y es la que utilizaremos mayormente en este curso.

Los colores en CSS pueden ser representados de al menos tres formas diferentes:

- Representados con **palabras claves** para cada color, como: red, green, blue, pink, yellow, black, etc.
- Usando la composición de tres colores (**rojo**, **verde** y **azul**): para esto podemos usar notación hexadecimal o las funciones rgb() y rgba().
- Usando la composición mediante valores de **Matiz**, **Saturación** y **Luminosidad** con: hls() y hlsa().

Con respecto a los valores hexadecimales, cada color está representado por 6 digitos, que representan 3 pares de hexadecimales: FF - FF - FF (rojo, verde y azul) (#FF0000, #00FF00, #0000FF) en el que cada par puede tomar valores hexadecimales entre 00 y FF. Cada uno equivale a valores decimales entre 0 y 255, donde 0 es la ausencia de ese color y 255 la mayor cantidad disponible. De esta manera cada color se forma por la combinación de diferentes proporciones de rojo, verde y azul.

(Hexadecimales)

- #000000 es equivalente a Negro
- #FF0000 es equivalente a Rojo
- #00FF00 es equivalente a Verde

- #0000FF es equivalente a Azul
- #FFFFFF es equivalente a Blanco

Pixel art, la unidad mas pequeña unidad de medida dentro de css.

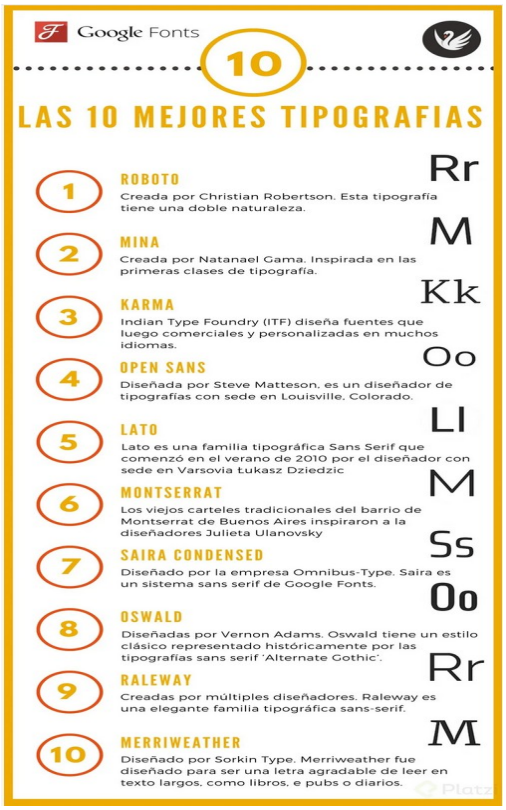
Video17
Inspector de elemento CTRL+SHIFT+i
Para ver y depurar el código de una página html, el navegador incluye una herramienta llamada **Inspector de elementos**, o simplemente **inspector**, que abre, en una sección de la ventana, una serie de espacios con información técnica muy detallada sobre todo lo que sucede en el DOM, incluídos los estilos que tienen aplicados cada uno de los elementos del html.

La mayoría de los navegadores incluye algún tipo de **Inspector**, en el curso usamos Google Chrome, pero la misma herramienta (o similar) la encuentras en Firefox, Opera, Edge, etc.

Utilizando el Inspector podemos hacer modificaciones (temporales) manualmente en el html de cualquier sitio web, consultar sus estilos y recursos enlazados, hacer pruebas en tiempo real con JavaScript, monitorear variables o eventos entre muchas otras tareas útiles para cualquier desarrollador.

Inspector de elemento, un buena ayuda es el Selector de elemento si desconozco como hacer.

Estados de los elementos, cuando estoy encima, hover



Video 18

Fuentes tipograficas,

Los tipos de texto, también conocidos como **tipos de letras** o **fuentes**, son el conjunto de diseños tipográficos que representan a cada una de las letras y los caracteres gráficos en el documento. Su nombre correcto es **tipografía**. Los diferentes tipos de fuente están basados en archivos que existen en cada sistema operativo.

Algunos ejemplos de **tipos de texto** o fuentes, son:

- Arial
- Times New Roman
- Verdana
- DeJaVu
- Lato
- OpenSans
- Roboto

CSS permite utilizar **fuentes** diferentes a las disponibles en el sistema operativo del cliente mediante la importación o el enlace a archivos de fuentes externas. Las más usadas son las que están disponibles a través del sitio web de **Google Fonts**.

Al definir el tipo de texto asociado a una clase css con la propiedad **font-family** indicamos al navegador que debe intentar usar esa fuente en particular para darle la apariencia tipográfica a los textos de ese elemento html.

Crear o importar un nuevo tipo de fuente.

Si una fuente tiene mas de 2 palabras se pone entre comillas simples.
Creando Carpetas de ayudar en el nuevo navegador.

Video 19
Propiedades para los textos.
Además de todas las propiedades comunes que comparten los elementos estándar de html, como: display, position, margin, padding, top, left, right, bottom, border, etc., los elementos que admiten contenidos textuales aceptan una serie particular de propiedades entre las que se encuentran las siguientes:

- **font-family**: define el tipo de fuente aplicado al texto.
- **color**: define el colore del texto.
- **line-height**: define la altura desde la base del texto hasta la base de la siguiente línea de texto.
- **font-size**: define el tamaño del texto, admite cualquiera de las unidades de medida disponibles.
- **letter-spacing**: define el espaciado entre las letras del texto.
- **font-weight**: define el ““peso”” de la letra, negrita, normal, light y normalmente se indica en múltiplos de 100 o usando keywords.
- **text-decoration**: define el decorado del texto como subrayado, tachado, con subrayado superior, etc.
- **text-transform**: permite transformar el estado de mayúsculas / minúsculas en el texto, usando uppercase para mayúsculas sostenidas, lowercase para minúsculas sostenidas, etc.

`line-height` para modificar el alto de linea
`font-size` para modificar el tamaño de la fuente
`font-weight` para modificar el tipo de fuente
`font-style` para modificar el estilo de la fuente
`letter-spacing` para modificar el espacio entre letras
`text-transform` para transformar la fuente (mayusculas, minuculas, etc)
`text-decoration` para moodificar la decoración de la fuente

Video 20
Dimensiones fijas para elementos
Todos los elementos html comparten algunas propiedades de estilo, entre éstas se encuentran las propiedades relacionadas con sus dimensiones: **width** (ancho) y **height** (alto).

Al manipular las propiedades de dimensiones hay que tener en cuenta que si los contenidos de los elementos que estamos estilizando, son más grandes que las dimensiones que hemos indicado, se pudieran generar resultados inesperados en la apariencia, como solapamiento o desbordamiento.

Si yo quisiera cambiarle el tamaño a un solo elemento me voy a project y le pongo width.
Se puede solapar elementos.

Comentarios rapidos en html

Video 21
Manipular color fondo
Algunas de las propiedades de css relacionadas con la apariencia del fondo de los elementos son:

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

- Color// esta propiedad le da color a los textos.

Los valores en Hexadecimales
Gris oscuro = #1d252c
Gris claro = #626262
Gris de fondo= #1b2127
Mi primer Selector descendente .header a

selectores descendentes (anidados).
Los bacgrounds

Video 22
Workshop toca el turno a los bordes.
Todos los elementos html admiten la propiedad de css **border**, que define la apariencia que tendrá el contorno del componente.
El borde puede ser de muchos estilos, y al igual que las propiedades margin y padding que aprenderás más adelante, a los bordes se les puede colocar estilos tanto de forma general con la propiedad **border**, como de acuerdo al lado del elemento que se indique: border-top, border-right, border-bottom y border-left.

Con la propiedad **border-radius** se define el redondeado de las esquinas de los bordes.

Donde puedo modificarle el Tamaño, el tipo, Estilo

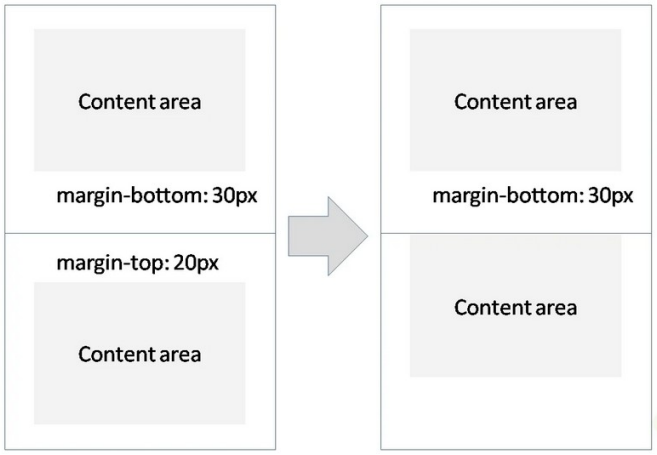
El border se pued ecompartar solo de un tipo de color y lo demas lo toma del color de las letras fonts.

Hay borders puntito, degradado, estilos etc.

Video 23 Margenes.

Separación dentro de los elementos

Los márgenes en CSS son el espacio que separa a los elementos html entre sí. Hay elementos de html que traen márgenes predefinidos (por defecto) en los estilos propios del navegador como el caso de: body, h1, h2, h3, h4, h5, h6, ol, ul, li, p, y muchos otros.



Cuando hay dos márgenes de elementos diferentes que colindan entre sí, se presenta una situación llamada ““margin collapsing”” en la que el mayor margen de los dos se superpone al otro.

Se puede asignar una medida de margin para los cuatro lados del elemento, o márgenes individuales para cada uno de los lados con: margin-top, margin-right, margin-bottom y margin-left.

Se puede centrar un elemento html colocándole el valor de **margin: 0 auto**, cuando dicho elemento tiene display *block*.

Video 24

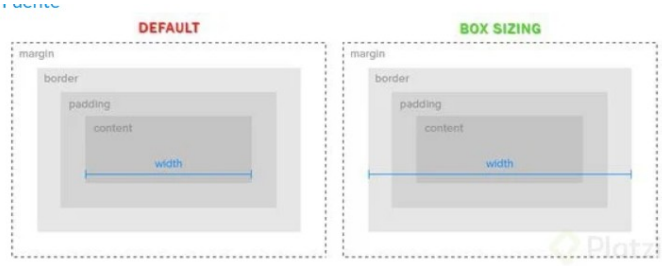
Afuera de los elementos y dentro de los elementos Rellenos

Así como el margen separa a los elementos html entre sí, la propiedad **padding** de relleno, permite definir una separación entre el contenido interno y el *borde* de un elemento.

Al inspeccionar los elementos html en el navegador, se puede apreciar el margin con color naranja y el padding con color verde.

Una forma de identificar cuándo es mejor usar margin o padding en un elemento, es evaluando la necesidad de usar borde o background, ya que son éstos: el borde y el background, los que realmente diferencian el uso de uno u otro.

Video 25
El modelo de Caja.
El modelo de caja es un concepto teórico de css que representa a cada elemento html en base sus propiedades de: **margin**, **border**, **padding** y **dimensiones** (alto y ancho).
Para visualizar un elemento html en su representación como modelo de caja debemos irnos a la parte baja de la sección *styles* del inspector de elementos, o en la sección llamada **Computed**.
En el modelo de caja, el **ancho total** de un *elemento html* equivale a la sumatoria de los valores de: **width**, **padding-left**, **padding-right**, **border-left-width**, **border-right-width**. De manera similar aplica para el **alto total** de cada *elemento*. Aunque **margin-left** y **margin-right**, forman parte del modelo de caja, no se incluyen para el calculo del ancho total.



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

Video 26
Tipos de display
Display es la propiedad de css que indica cómo debe ser mostrado un elemento html. Todos los elementos tienen algún tipo de display. Si un elemento no se ve en pantalla es porque seguramente su display es none.
Los valores más comunes que puede recibir la propiedad **display** son:

- **block**: el elemento intenta abarcar todo el ancho posible.
- **inline**: reduce su tamaño exclusivamente hasta lo que abarca su contenido, descartando las propiedades width y height.
- **inline-block**: combina lo mejor de block e inline, ya que respeta las dimensiones indicadas en las propiedades width y height, pero coloca el elemento en línea (al costado) de elementos hermanos que también tengan display: inline o inline-block.
- **flex**: asume algunas propiedades por defecto que favorecen la alineación de los elementos internos.
- **grid**: similar a flex, asume algunas propiedades por defecto organizando los contenidos en filas y columnas.
- **none**: oculta el elemento.

Display:flex, en la zona de articule me acomoda la zona lo mejor acomodado posible
NOTA: en display:inline- table me mantiene siempre dentro del margen mi titulo h1

Video 27
Propiedades de Flexbox
Flexbox se refiere al tipo de display en css que permite un manejo *flexible* de la alineación, dimensionamiento y distribución de elementos html.
Esta propiedad se aplica a un elemento padre, pero va a afectar principalmente a sus elementos hijos directos. Por defecto, los elementos internos quedan alineados unos seguidos de los otros. El

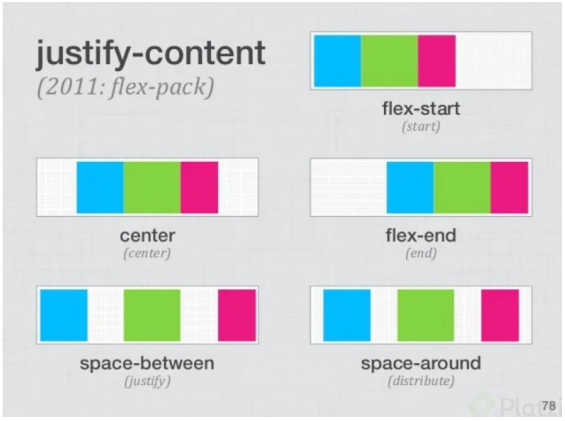
comportamiento del modelo de caja de estos elementos hijos también se ha modificado, ya que pierden el efecto de su propiedad margin.

Los elementos hijos de un padre con propiedad **display: flex** tienen a su disposición algunas nuevas propiedades que aportan mayor flexibilidad a su comportamiento. Una de estas propiedades es **flex-shrink** que, junto a la propiedad **flex-wrap** del padre, permite adaptar y distribuir los elementos de manera dinámica en el espacio horizontal disponible hasta ocupar todo el espacio, y luego pasar a ocupar dinámicamente las siguiente filas hacia abajo.

Margin // los margenes que tiene mi caja
widht // lo ancho
height// lo alto

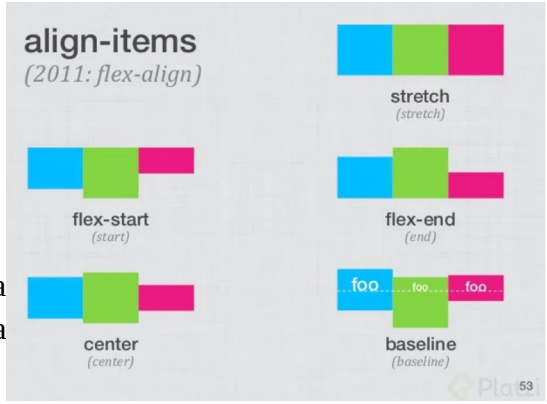
Video 28
Alineado de manera horizontal
La propiedad de css que nos permite definir la forma en que se alinearán o distribuirán los hijos de un elemento al que se le ha asignado un *display flex* es: **justify-content**. Y puede tomar entre otros valores, los siguientes:

- **flex-start**: para alinear todos los elementos hacia el inicio del espacio disponible.
- **flex-end**: para alinear todos los elementos hacia el final, a la derecha.
- **center**: para alinear todos los elementos al centro del espacio disponible.
- **space-between**: para distribuir los elementos con un espacio proporcional e igual entre ellos.
- **space-evenly**: para distribuir los elementos con un espacio proporcional e igual entre ellos (incluyendo el primer y último elementos con respecto a los extremos del espacio disponible).
- **space-around**: similar a *space-evenly* pero tanto en el primero como en el último elemento, el espacio hacia los extremos es la mitad del espacio usado entre los elementos.



Video 29
Elemento de manera Vertical
Similar a como sucede con *justify-content*, es posible alinear y distribuir los elementos internos en el espacio vertical disponible usando la propiedad **align-items**, que puede tomar también los valores de: **flex-start**, **flex-end** y **center**.

Algo que es muy importante y se debe tener en cuenta a la hora de usar **align-items** y **justify-content** es que dependiendo de la propiedad **flex-direction** que se haya definido, el efecto de



ambos se invierte, no en cuanto a sus elementos internos, sino en cuanto a si se debe usar uno u otro de manera vertical u horizontal.

IMPORTANTE: Cuando la propiedad **flex-direction** se ha definido como **column**, la propiedad **justify-content** ya no va a aplicar sobre la alineación horizontal, sino sobre la vertical. Y **align-items** ya no aplicaría sobre la alineación vertical sino la horizontal. Se intercambian sus efectos.

```
.flexbox {
   display: flex;
   flex-wrap: wrap;
   height: 200px;
   justify-content: space-evenly;
   align-items: center;
   border: 1px solid blue;
}
```

Para centrar completamente los elementos internos de manera vertical y horizontal en su elemento padre, debemos usar el valor **center** en ambas propiedades.

Video 30

`justify-content: space-between;` distribuye los espacios entre elementos dentro del contenedor tipo flex de forma equitativa sin dejar espacios en los extremos. Al tener solo dos elementos logra el efecto deseado de mandarlos a los extremos.

Nota: recuerda que el `display:flex;` solo se aplica a los hijos directos.

Se pueden ver los ejemplos que han desarrollado otros

Video 31

En esta clase aplicaremos a nuestro proyecto los estilos necesarios para que la sección principal o **hero** de nuestro portafolio se vea correctamente. También completaremos la información de nuestro footer y le aplicaremos los respectivos estilos.

En esta clase vemos que además de las propiedades relacionadas con *display:flex*, en lo que se refiere a la alineación de elementos internos, existen también propiedades equivalentes que nos permite alinear textos dentro de un contenedor, estas propiedades son: **text-align** y **vertical-align**.

Div // etiqueta para tener divisiones.

Se puede usar otra tecnica para alinear

vertical-align, cuando tienes con imagenes y seleccionar donde ponerlo.

Utilizamos para mejorar el tamaño de las imagenes

```
object-fit: cover;
```

Cuando un elemento se desborde, cuando una imagen salga de una caja se puede usar `overflow: hidden;`

aplicando el `flex-shrink: 0;`
Siempre se van a quedar en su tamaño, las imagenes

posiciones relativas para poder superponer elementos sobre otros en event-detail
```
position: relative;
```

Video 35
Hacer un Contenedor.
En esta clase agregamos algunos **divs** a manera de *contenedores* para central todo el contenido de la página y le definiremos algunos estilos con la clase **.container**. Para resolver algunos detalles generados por la implementación de los contenedores, aplicaremos una solución un tanto creativa heredando varias *propiedades* de los elementos padre con el keyword **inherit**.

Te propongo adicionalmente el reto de aplicar estilos adicionales para corregir algún otro detalle que hayas logrado identificar para completar la estilización de todos los elementos de nuestro portafolio.

Justify horizontal
align vertical

```
object-fit: cover;
```
me ajusta la imagen

1 ¿Cuántos pixeles a la derecha hay en la siguiente declaración de CSS?

```
{ margin: 10px11px12px13px }
```

2 ¿Cuántos pixeles a la izquierda hay en la siguiente declaración de CSS?

```
{ margin: 10px11px12px }
```

3 ¿Cuántos pixeles hacia arriba hay en la siguiente declaración de CSS?

```
{ padding: 05px }
```

4 ¿Cuánto ancho ocupa un elemento con las siguientes propiedades?:

```
{
display:block;
width: 99px;
padding: 2px:
border: 1px solid pink
}
```

5 ¿Cuánto alto ocupa un elemento con las siguientes propiedades?

```
{
display:block;
height: 50px;
padding: 020px20px0
}
```

6 ¿Cuál es el último carácter de un hexadecimal?

7 ¿Qué color se produce de este hexadecimal? #FFFFFF

8 ¿Qué color se produce de este hexadecimal? #FF0000

9 ¿Qué color se produce de este hexadecimal? #00FF00

10 ¿Qué color se produce de este hexadecimal? #0000FF

11 ¿Qué color se produce de este hexadecimal? #000000

12 ¿Con qué propiedad de CSS puedo subrayar un texto?

13 ¿Con qué propiedad de CSS puedo poner un texto en mayúsculas?

14 ¿Con qué propiedad de CSS puedo poner un texto en negritas?

15 Si tengo un elemento con display flex ¿Con qué propiedad centro su contenido verticalmente?

16 Si tengo un elemento con display flex y flex-direction column ¿Con qué propiedad centro su contenido verticalmente?

17 Si tengo un elemento con display flex ¿Con qué propiedad centro su contenido horizontalmente?

18 Si tengo un elemento con display flex y flex-direction column ¿Con qué propiedad centro su contenido horizontalmente?

19	Dentro del desarrollo web el código de las interfaces de usuario es parte del:
20	Dentro del desarrollo web el encargado de respaldar datos en una base de datos es el:
21	Un selector de *clase* se inicia con:
22	Un selector de id se inicia con:
23	¿Con qué propiedad de CSS puedo poner una imagen de fondo?
24	¿Con qué valor puedo heredar el valor de la misma propiedad del elemento padre?
25	El modelo de caja está formado por:
26	¿Con qué atributo HTML le indicamos a una etiqueta img cual es la ruta de la imagen a mostrar?
27	¿Cuál es la etiqueta de encabezado más importante?
28	¿Dentro de qué etiqueta html ubicamos el <title></title>?
29	¿Con qué etiqueta puedo enlazar una hoja de estilos con un archivo html?
30	¿Con qué atributo le indicamos a un tag <a> la ruta del destino del enlace?