Clase grabada

¡Volvemos!

English Always | alkemy

Clase grabada

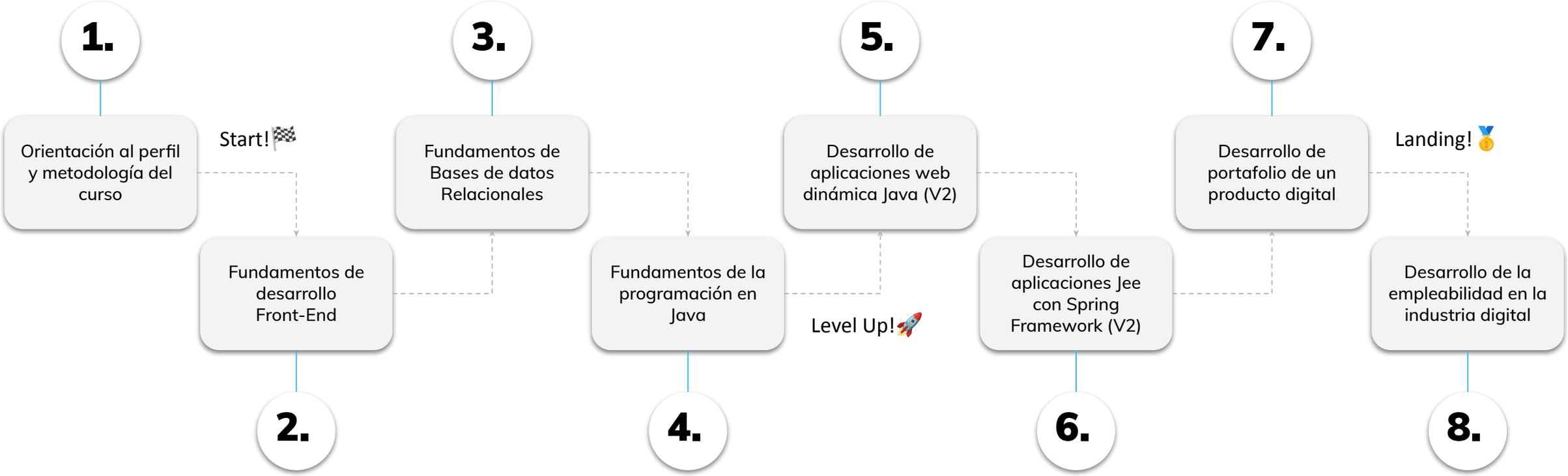
Aplicación de estilos y responsividad
Manejando Hojas de Estilo

Plan formativo: Desarrollo de Aplicaciones Full Stack Java Trainee V2.0

English Always | alkemy

HOJA DE RUTA

¿Cuáles **skill** conforman el programa?



English Always | alkemy

REPASO CLASE ANTERIOR

En la clase anterior trabajamos 📚:

- ✓ *Uso de CSS para incorporar y trabajar el diseño web*
- ✓ *Diferencia y uso los selectores como ID y Class*
- ✓ *Comparación de los estilos en línea, embebidos, archivos externos*

LEARNING PATHWAY

N°2 . ¿Sobre qué temas trabajaremos?

2. Start! 🏁

FUNDAMENTOS DE DESARROLLO FRONT-END → **Aplicación de estilos y responsividad** → **Actividad N° 1: Aplicando Flexbox**

En esta lección, abordaremos dos temas esenciales en el mundo de CSS: como crear un modelo de cajas a partir de las diferentes partes del modelo de cajas que lo componen: Contenido (Content), Relleno (Padding), Borde (Border) y Margen (Margin)

OBJETIVOS DE APRENDIZAJE

¿Qué aprenderemos?

- *Comprender el uso del modelo de Cajas*
- *Aprender a incorporar los estilos más utilizados (fuentes, líneas, cajas, etc...)*
- *Buenas prácticas al construir una hoja de estilos y manejo de assets e imágenes.*

El modelo de Cajas

El modelo de Cajas

El modelo de cajas es uno de los conceptos fundamentales en CSS que describe cómo se representa y se estructura visualmente un elemento HTML en una página web. Cada elemento HTML se considera una "caja" que tiene varias propiedades, como el contenido, el relleno, el borde y el margen.

El modelo de Cajas

Las diferentes partes del modelo de cajas son:

Contenido (Content): Es el área que contiene el contenido real del elemento, como texto, imágenes, videos u otros elementos HTML.

Relleno (Padding): Es el espacio entre el contenido y el borde del elemento. El relleno proporciona un espacio adicional dentro del elemento, lo que afecta la distancia entre el contenido y el borde.

El modelo de Cajas

Las diferentes partes del modelo de cajas son:

Borde (Border): Es una línea que rodea el contenido y el relleno del elemento. El borde puede tener diferentes estilos, anchos y colores.

Margen (Margin): Es el espacio exterior al elemento, que separa al elemento de otros elementos en la página. El margen crea un espacio vacío alrededor del elemento.

El modelo de Cajas

Control Preciso: El modelo de caja permite un control preciso sobre cómo se visualizan y se posicionan los elementos en una página web. Puedes ajustar dimensiones, márgenes, rellenos, bordes y más.

Diseño Responsivo: Facilita la creación de diseños web responsivos, donde los elementos se adaptan a diferentes tamaños de pantalla y dispositivos.

Flexibilidad: *Permite la creación de diseños flexibles y complejos mediante la combinación de márgenes, rellenos, bordes y dimensiones.*

Separación de Contenido y Presentación: *Promueve la separación de contenido y presentación, lo que facilita el mantenimiento y la actualización de un sitio web.*

El modelo de Cajas

Existen diferentes formas de especificar los márgenes en CSS:

Margen individual: Se puede especificar márgenes individualmente para cada lado del elemento utilizando las propiedades margin-top, margin-right, margin-bottom y margin-left.

```
.elemento {
 margin-top: 10px;
 margin-right: 20px;
 margin-bottom: 10px;
 margin-left: 20px;
}
```

En este caso, se establecen márgenes de 10 píxeles en la parte superior e inferior, y márgenes de 20 píxeles en los lados derecho e izquierdo del elemento.

El modelo de Cajas

Existen diferentes formas de especificar los márgenes en CSS:

Margen abreviado: También se puede utilizar la propiedad margin para especificar los márgenes de manera abreviada. Podes proporcionar uno, dos, tres o cuatro valores separados por espacios. Los valores se aplicarán en el orden de arriba, derecha, abajo e izquierda

```
.elemento {
 margin: 10px 20px 10px 20px; //arriba, derecha, abajo e izquierda
}

.elemento {
 margin: 10px 20px; //arriba y abajo, derecha e izquierda
}

.elemento {
 margin: 10px; //en todos los puntos por igual
}
```

El modelo de Cajas

Existen diferentes formas de especificar los márgenes en CSS:

Margen automático: Utilizar “margin: auto” permite centrar horizontalmente un elemento dentro de su contenedor. Esto ajustará automáticamente los márgenes laterales para que el elemento esté centrado.

```
.elemento {
 margin: auto;
}
```

Los márgenes también pueden tener valores negativos, lo que permite que los elementos se superpongan entre sí o se ajusten a un diseño específico. Es importante tener en cuenta que los márgenes se acumulan entre elementos adyacentes. Esto significa que si hay varios elementos uno al lado del otro, los márgenes se sumarán, lo que puede afectar el espacio total entre ellos.

El modelo de Cajas

Existen diferentes formas de especificar el relleno (Padding) en CSS:

Padding individual: Se puede especificar el padding individualmente para cada lado del elemento utilizando las propiedades padding-top, padding-right, padding-bottom y padding-left.

```
.elemento {
 padding-top: 10px;
 padding-right: 20px;
 padding-bottom: 10px;
 padding-left: 20px;
}
```

En este caso, se establece un padding de 10 píxeles en la parte superior e inferior, y un padding de 20 píxeles en los lados derecho e izquierdo del elemento.

El modelo de Cajas

Existen diferentes formas de especificar el relleno (Padding) en CSS:

Padding abreviado: También se puede utilizar la propiedad padding para especificar el padding de manera abreviada. Al igual que con los márgenes abreviados, podes proporcionar uno, dos, tres o cuatro valores separados por espacios, que se aplicarán en el orden de arriba, derecha, abajo e izquierda.

```
.elemento {
 padding: 10px 20px 10px 20px; //arriba, derecha, abajo e izquierda
}

.elemento {
 padding: 10px 20px; //arriba y abajo, derecha e izquierda
}

.elemento {
 padding: 10px; //en todos los puntos por igual
}
```

El modelo de Cajas

Existen diferentes formas de especificar los márgenes en CSS:

Padding automático: Utilizar “padding: auto” permite centrar horizontalmente un elemento dentro de su contenedor. Esto ajustará automáticamente los márgenes laterales para que el elemento esté centrado.

```
.elemento {
 padding: auto;
}
```

El padding también puede tener valores negativos, lo que permite que el contenido se extienda más allá de los límites del elemento.

El padding es útil para crear espacios internos alrededor del contenido de un elemento y puede ayudar a mejorar la legibilidad y el diseño de una página web.

Estilos más utilizados

Fuentes

La tipografía en diseño web es esencial, afectando legibilidad y estilo. CSS ofrece propiedades para su estilo. Ejemplos comunes incluyen font-family, font-size, font-weight y text-align. Estas controlan cómo se ve y se lee el texto en una página.

Fuentes

Font-family: Esta propiedad se utiliza para especificar el tipo de fuente que se utilizará para el texto. Se puede utilizar nombres de fuentes específicos o una lista de fuentes genéricas, como "Arial", "Helvetica", "Times New Roman", etc. También es posible definir múltiples fuentes en caso de que una no esté disponible en el dispositivo del usuario.

```
.selector {
  font-family: Arial, sans-serif;
  }
```

Fuentes

Font-size: Permite establecer el tamaño de la fuente. Se puede utilizar unidades de píxeles, puntos, porcentaje, em o rem para especificar el tamaño. Al elegir un tamaño de fuente, es importante considerar la legibilidad en diferentes dispositivos y tamaños de pantalla.

Text-align: Esta propiedad se utiliza para alinear el texto dentro de su contenedor. Se puede alinear el texto al centro, a la izquierda, a la derecha o justificarlo.


```
.selector {
 font-size: 16px;
 text-align: center;
 }
```

Fuentes

font-weight: Esta propiedad controla el grosor o la negrita del texto. Se puede utilizar valores como "normal", "bold", "bolder", etc. para establecer diferentes niveles de peso en la fuente.

line-height: Se utiliza para establecer la altura de línea, es decir, el espacio vertical entre líneas de texto. Un valor adecuado para line-height puede mejorar la legibilidad y el flujo visual del texto.


```
.selector {
 font-weight: bold;
 line-height: 1.5;
 }
```

Box-shadow

La propiedad box-shadow en CSS se utiliza para agregar sombras a elementos HTML, como cajas, textos, imágenes y otros elementos

Podes utilizar esta propiedad para crear efectos de profundidad y resaltar visualmente los elementos en tu diseño.

Box-shadow

Desplazamiento horizontal: Especifica la distancia horizontal desde el elemento donde se mostrará la sombra. Un valor positivo desplaza la sombra hacia la derecha, mientras que un valor negativo la desplaza hacia la izquierda.

Desplazamiento vertical: Indica la distancia vertical desde el elemento donde se mostrará la sombra. Un valor positivo desplaza la sombra hacia abajo, mientras que un valor negativo la desplaza hacia arriba.

```
.selector {
 box-shadow: 3px 3px;
 }
```

Box-shadow

Desenfoque (opcional): Controla el nivel de borrosidad de la sombra. Un valor mayor crea una sombra más difuminada, mientras que un valor de 0px produce una sombra nítida y definida.

Extensión (opcional): Permite ampliar o reducir el tamaño de la sombra. Un valor mayor aumenta el tamaño de la sombra, mientras que un valor menor la reduce.

Color (opcional): Especifica el color de la sombra. Podes utilizar valores de color en formato hexadecimal, RGB o nombres de colores predefinidos.


```
.selector {
  box-shadow: 4px 2px rgba(0, 0, 0, 0.3);
}
```

Flexbox

Flexbox

Flexbox es un módulo de diseño en CSS que proporciona una forma flexible y eficiente de organizar y distribuir elementos en un contenedor.

Con Flexbox, podes crear diseños de cajas flexibles y receptivos que se adaptan automáticamente a diferentes tamaños de pantalla y orientaciones.

Flexbox

El modelo de caja flexible se basa en dos conceptos clave:

- **el contenedor flex (flex container)**
- **los elementos flexibles (flex items).**

Para utilizar Flexbox, debes seguir estos pasos:

Aplica la propiedad display: flex; al contenedor que deseas convertir en un contenedor flex.

Esto define un nuevo contexto de formato flexible para sus elementos secundarios.

```
.selector {
 display: flex;
 }
```

Box-shadow

Configurar la dirección y el flujo:

Utiliza la propiedad **flex-direction** para establecer la dirección de los elementos secundarios dentro del contenedor flex. Puede ser **row (horizontal), column (vertical), row-reverse** o **column-reverse.**

Utiliza la propiedad **flex-wrap** para permitir que los elementos secundarios se envuelvan en múltiples líneas si no caben en una sola línea.

```
.selector {
 display: flex;
 flex-direction: row;
 }
```

```
.selector {
 display: flex;
 flex-wrap: wrap;
 }
```

Box-shadow

Controlar la distribución y el espacio:

Utiliza la propiedad **justify-content** para alinear los elementos secundarios a lo largo del eje principal del contenedor flex. Puede ser **flex-start, flex-end, center, space-between, space-around** o **space-evenly.**

```
.selector {
 display: flex;
 justify-content: center;
 }
```

Utiliza la propiedad **align-items** para alinear los elementos secundarios a lo largo del eje transversal del contenedor flex. Puede ser **flex-start, flex-end, center, stretch** o **baseline.**

```
.selector {
 display: flex;
 align-items: flex-start;
 }
```

Utiliza la propiedad **align-content** para controlar el espacio entre las líneas de elementos secundarios si hay varias líneas dentro del contenedor flex. Puede ser: **flex-start, flex-end, center, space-between, space-around, y stretch.**

```
.selector {
 display: flex;
 align-content: space-between;
 }
```

Box-shadow

Estilizar los elementos secundarios:

Aplica propiedades específicas a los elementos secundarios, como **flex-grow, flex-shrink y flex-basis**, para controlar cómo se distribuye el espacio disponible entre ellos.

Utiliza la propiedad **order** para cambiar el orden de los elementos secundarios dentro del contenedor flex. La propiedad order se utiliza para cambiar el orden visual de los elementos secundarios dentro del contenedor flex. Los elementos con un valor menor de order se mostrarán antes que los elementos con un valor mayor.

```
.selector {
 display: flex;
 flex-grow: 2;
 flex-shrink: 1;
 flex-basis: 200px;
 }
```

```
.selector {
 display: flex;
 order: 2;
 }
```

Buenas prácticas al construir una hoja de estilos

Buenas prácticas

Al seguir estas buenas prácticas, podrás crear hojas de estilos CSS más mantenibles, escalables y fáciles de trabajar, lo que facilitará el desarrollo y el mantenimiento de tu proyecto a largo plazo.

Buenas prácticas

Utilizar comentarios: Agrega comentarios en tu código CSS para explicar su propósito y proporcionar documentación. Esto ayuda a otros desarrolladores (y a ti mismo en el futuro) a comprender rápidamente el propósito de ciertas reglas y facilita la tarea de hacer cambios o modificaciones.

Nomenclatura consistente: Utiliza una nomenclatura coherente y descriptiva para nombrar tus clases y selectores. Esto ayuda a comprender el propósito y el contexto de cada estilo. Podes utilizar convenciones como BEM (Block Element Modifier) o SMACSS (Scalable and Modular Architecture for CSS) para estructurar y nombrar tus estilos de manera consistente.

```
/* Comentarios para documentación */
/* Estilos globales */


/* Nomenclatura consistente y descriptiva */
```

Buenas prácticas

Evitar selectores demasiado específicos: Evita el uso de selectores demasiado específicos y anidados excesivamente. Esto puede complicar el mantenimiento y la comprensión del código. Opta por selectores más generales y utiliza la cascada de CSS para aplicar estilos específicos según sea necesario.

Mantener la especificidad bajo control: Controla la especificidad de tus estilos para evitar conflictos y dificultades en la aplicación de estilos. Evita el uso excesivo de selectores de ID y trata de utilizar clases y selectores de tipo en su lugar.

```
/* Evitar selectores demasiado específicos */

/* Mantener la especificidad bajo control */
```

Buenas prácticas

Reutilizar estilos: Busca oportunidades para reutilizar estilos en lugar de duplicar código. Utiliza clases y selectores comunes para aplicar estilos similares a varios elementos en lugar de escribir reglas separadas para cada uno.

Agrupar estilos relacionados: Agrupa estilos relacionados o similares para una mayor legibilidad y organización. Esto facilita la identificación de estilos relacionados y su mantenimiento posterior.

```
/* Reutilización de estilos */


/* Agrupar estilos relacionados */
```

Buenas prácticas

Mantener el archivo CSS ordenado: Organiza tu archivo CSS de manera lógica y estructurada. Podes dividirlo en secciones basadas en componentes, características o páginas específicas. Además, mantén una estructura de carpeta ordenada para tus archivos CSS si tienes un proyecto más grande.

Utilizar herramientas de construcción y preprocesadores: Considera el uso de herramientas de construcción, como Gulp o Grunt, y preprocesadores de CSS, como Sass o Less, para optimizar y organizar tu flujo de trabajo de desarrollo CSS. Estas herramientas te permiten automatizar tareas repetitivas y facilitan la escritura de código más eficiente y modular.

```
/* Mantener el archivo CSS ordenado */

/* Estilos específicos para una página o componente */
```

- Manejo de assets e imágenes. Conociendo rutas absolutas y relativas.

Buenas prácticas

Cuando trabajas con assets e imágenes en una página web, es importante comprender el manejo de rutas absolutas y relativas para acceder y mostrar correctamente los archivos.

Manejo de assets e imágenes.

Rutas absolutas

Las rutas absolutas **especifican la ubicación completa** del archivo desde la raíz del sistema de archivos o el dominio.

Esta ruta comienza desde la raíz del dominio o la estructura de archivos y se utiliza para acceder al archivo desde cualquier ubicación en el sitio web.

Las rutas absolutas son útiles cuando necesitas hacer referencia a recursos que se encuentran en ubicaciones fijas y permanentes en el servidor, como imágenes, archivos CSS o scripts.

```
<img src="https://abc.com" alt="abc">

<img src="https://www.examp.com" alt="exam">
```

Manejo de assets e imágenes.

Rutas relativas

Las rutas relativas se definen en relación con la ubicación del archivo HTML actual. En lugar de especificar la ruta completa desde la raíz, las rutas relativas utilizan referencias relativas a la ubicación actual.

Rutas relativas a la misma carpeta: Por ejemplo, si tenes una imagen llamada logo.jpg en la misma carpeta que tu archivo HTML, podes hacer referencia a ella utilizando logo.jpg o ./logo.jpg.

Rutas relativas a una carpeta padre: Si tenes una estructura de carpetas y necesitas hacer referencia a un archivo que se encuentra en una carpeta padre, podes utilizar ../ para navegar hacia arriba en la estructura de carpetas utilizando ../img/pato.jpg.

```
<img src="logo.png" alt="Logo">
<img src="../img/pato.jpg" alt="Pato">
```

Manejo de assets e imágenes.

Es importante considerar el contexto y la ubicación del archivo HTML al especificar rutas relativas. Esto te permite acceder a los assets e imágenes de forma correcta independientemente de la ubicación del archivo HTML. Asegúrate de utilizar las rutas adecuadas en los atributos src o href de tus elementos <img>, <link>, <script>, etc.

Recorda que las rutas absolutas y relativas son conceptos relativos al servidor o al sistema de archivos en el que se encuentra tu sitio web. Por lo tanto, es fundamental comprender la estructura de tu proyecto y utilizar las rutas adecuadas para acceder a los assets e imágenes de manera correcta y consistente en tu página web.

```
<img src="logo.png" alt="Logo">
<img src="../img/pato.jpg" alt="Pato">


<img src="https://abc.com" alt="abc">
<img src="https://examp.com" alt="exam">
```

LIVE CODING

Ejemplo en vivo

Estilización de la Tipografía:

A partir de lo aprendido, usando las propiedades de CSS relacionadas con la tipografía, crear una página web simple utilizando HTML y CSS para aplicar estilos de fuente a elementos de texto específicos.

El objetivo es practicar el uso de propiedades de CSS relacionadas con la tipografía.

Podes crear la sesiones sobre como crear una receta, o un día de verano, etc.

Tiempo: 15 minutos

LIVE CODING

Ejemplo en vivo

Pasos:

- *Crea un archivo HTML llamado index.html y un archivo CSS llamado styles.css.*
- *En el archivo HTML, crea la estructura básica de una página web con un encabezado (<header>), un párrafo (<p>), una lista desordenada (<ul>) con al menos tres elementos de lista (<li>), y un pie de página (<footer>).*
- *Dentro del <header>, agrega un título (<h1>) y un subtítulo (<h2>).*
- *En el archivo CSS (styles.css), enlaza las fuentes de Google Fonts para usar en tu página web. Por ejemplo, puedes utilizar la fuente "Roboto" para el texto principal y la fuente "Open Sans" para el encabezado.*

LIVE CODING

Ejemplo en vivo

Pasos:

- *Personaliza el color de fuente, el espaciado entre líneas (line-height), y otros estilos de fuente según tus preferencias.*
- *Agrega cualquier otro estilo de fuente adicional que desees, como el uso de colores, tamaños de fuente y márgenes.*
- *Asegúrate de enlazar tu archivo CSS en el archivo HTML utilizando la etiqueta <link> en la sección <head>.*
- *Visualiza tu página web en un navegador y ajusta los estilos de fuente según sea necesario para lograr el aspecto deseado.*
- *Agrega las sesiones que sean necesarias.*

Ejercicio

Aplicando Flexbox

Aplicando Flexbox

Breve descipción

Consigna: ✍️

Crea una página web con un menú de navegación horizontal utilizando Flexbox.

El objetivo es aplicar las propiedades de Flexbox para alinear los elementos del menú horizontalmente.

Tiempo🕐: 20 minutos

Aplicando Flexbox

Breve descipción

Paso a paso: ⚙️

1. Crea un archivo HTML llamado index.html y un archivo CSS llamado styles.css.
2. En el archivo HTML, crea la estructura básica de la página web. Debe incluir un encabezado (<header>) y un contenedor (<nav>) para el menú de navegación.
3. Dentro del contenedor de navegación (<nav>), agrega una lista desordenada (<ul>) con al menos cinco elementos de lista (<li>). Cada elemento de lista debe contener un enlace (<a>) que represente un elemento del menú (por ejemplo, Inicio, Acerca de, Servicios, Portafolio, Contacto).
4. En el archivo CSS (styles.css), aplica estilos básicos al cuerpo (<body>) de la página, como una fuente de Google Fonts y una paleta de colores.

Aplicando Flexbox

Breve descipción

Paso a paso: ⚙️

5. Utiliza Flexbox para crear un menú de navegación horizontal. Aplica las propiedades display: flex y justify-content para alinear los elementos del menú horizontalmente.
6. Agrega espaciado y margen entre los elementos del menú para mejorar la apariencia.
7. Personaliza el diseño del menú según tus preferencias, como el tamaño de fuente, el color de fuente y los efectos de hover.
8. Visualiza tu página web en un navegador de escritorio para asegurarte de que el menú sea atractivo y funcional en una pantalla grande.
9. Publica tu proyecto en un servidor web o plataforma de alojamiento para que otros puedan verlo en línea si lo deseas.

Solución

```
l>

s">


rset="UTF-8">

e="viewport" content="width=device-width, initial-scale=1.0">

="stylesheet" href="styles.css">

nú de Navegación</title>




ogo</h1>


ul>

    <li><a href="#">Inicio</a></li>

    <li><a href="#">Acerca de</a></li>

    <li><a href="#">Servicios</a></li>

    <li><a href="#">Portafolio</a></li>

    <li><a href="#">Contacto</a></li>

/ul>

>
```

```
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
}

header {
    background-color: #333;
    color: #fff;
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 10px 20px;
}

nav ul {
    list-style-type: none;
    display: flex;
}

nav ul li {
    margin-right: 20px;
}
```

```
nav ul li:last-child {
    margin-right: 0;
}

nav a {
    text-decoration: none;
    color: #fff;
    font-weight: bold;
    transition: color 0.3s ea
}

nav a:hover {
    color: #f00;
}
```

¿Alguna consulta?

RESUMEN

¿Qué logramos en esta clase?

- ✓ *Comprender el uso del modelo de cajas*
- ✓ *Aprender a aplicar estilos de fuentes y sombras*
- ✓ *Incorporar flexbox a nuestro CSS*
- ✓ *Aplicar buenas prácticas para construir una hoja de estilos*

#WorkingTime

Continuemos ejercitando

¡Antes de cerrar la clase! Te invitamos a: 👇👇👇

1. Repasar nuevamente la grabación de esta clase
2. Revisar el material compartido en la plataforma de Moodle (lo que se vio en clase y algún ejercicio adicional)
   a. Material 1: Lección 3: Aplicación de estilos y responsividad página: 10-24
   b. Material 2: Video de la lección N°3
3. Traer al próximo encuentro, todas tus dudas y consultas para verlas antes de iniciar nuevo tema.

¡Muchas Gracias!

Nos vemos en la próxima clase 👋🏻

English Always | alkemy