Guía de ejercicios 3 - Alineando contenido con Flexbox

¡Hola! Te damos la bienvenida a esta nueva guía de ejercicios.

¡Recuerda! La guía es un **complemento** de la sesión sincrónica. Si no fuiste a clases, te recomendamos revisar la grabación antes de revisar la guía.

¿En qué consiste esta guía?

Esta guía te ayudará a profundizar los conocimientos vistos en la clase y te entregará las herramientas necesarias para resolver el desafío, si bien no todos los aprendizajes entregados en esta guía son necesarios para resolver el desafío, los contenidos cubiertos son muy útiles en el día a día trabajando como desarrollador web.

En esta guía hablaremos del modelo de caja y revisaremos las técnicas vistas en clases para poner elementos uno al lado del otro. Para lograrlo, repasaremos algunos conceptos básicos como el de propiedad display y el valor flexbox.

Finalmente, profundizaremos en HTML semántico, el cual nos permite hacer código más entendible para los navegadores, así como también para otros desarrolladores del mismo sitio.

Para poder desarrollar esta guía con más facilidad, deberás tener en cuenta los siguientes aspectos:

- Conocer el concepto de etiqueta padre/hija e identificar visualmente un elemento a partir de esta referencia.
- Agregar una hoja de estilo a un html.
- Utilizar selectores de etiqueta y de clase.
- Utilizar selectores para seleccionar elementos descendientes de un elemento.
- Cambiar las propiedades de margin, border, padding, width y height de un elemento.
- Modificar la fuente, tamaño y alineación de un texto.
- Utilizar divs para dividir la página en secciones (o subsecciones).
- Agregar imágenes de fondo y ajustarlas utilizando CSS.
- Si tienes dudas puedes volver a revisar la guía de la unidad 2 - CSS.

¡Vamos con todo!

Tabla de contenidos

¡Comencemos!

Repaso Modelo de cajas

El modelo de cajas es uno de los conceptos más importantes detrás de CSS, y responde a la pregunta de cuánto mide realmente cada elemento.

Imagen 1. Modelo de cajas.
Fuente: Desafío Latam.

Los elementos básicos del modelo de caja los cubrimos en la unidad anterior, pero en esta ocasión hablaremos de aquellas propiedades que nos permiten poner una caja al lado de otra.

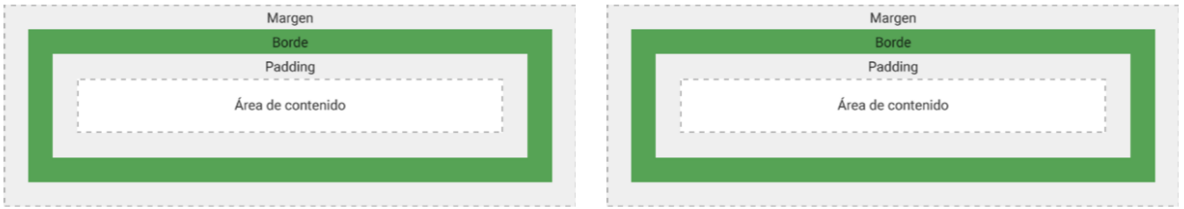

Imagen 2. Múltiples cajas.
Fuente: Desafío Latam.

Propiedad Display (inline, block, inline-block)

La propiedad display es la propiedad de CSS más importante para controlar la disposición de los elementos.

Cada elemento HTML tiene un valor de visualización predeterminado según el tipo de elemento que sea, siendo para la mayoría de estos **block** o **inline**.

Elementos block

Un elemento de bloque siempre comienza en una nueva línea y ocupa todo el ancho disponible (se extiende desde izquierda hacia la derecha todo lo que pueda). Por lo tanto, **si la propiedad tiene asignada el valor block, sin cambiar ninguna otra propiedad, no podremos poner un elemento al lado de otro.**

Algunos ejemplos son:

- <**div**>.
- <**h1**> - <**h6**>.
- <**p**>.
- <**form**>.
- <**header**>.
- <**footer**>.
- <**section**>.

Elementos inline

Por su parte, los elementos inline no comienzan en una nueva línea y su ancho ocupa el mínimo espacio posible (solo lo necesario según su contenido).

Algunos ejemplos son:

- <**span**>.
- <**a**>.
- <**em**>.
- <**strong**>.

Inline-block

Es un híbrido de valor display entre el block y el inline.

Inline-block se comporta como inline, donde puedes poner un elemento al lado de otro, pero a la vez se comporta como block, dado que puedes establecer un ancho y alto.

Utilizando inline-block podemos:

- Construir un menú.
- Construir una galería de imágenes.
- Construir una sección de características.

La mayoría de estas actividades tienen la misma estructura: un contenedor padre, donde hay varios elementos con la propiedad display: inline-block.

Construyendo un menú con la propiedad display

Si queremos construir un menú, creamos la siguiente estructura utilizando html:

```
<menu>
  <ul>
    <li>
      <a href=""> 1 </a>
    </li>
    <li>
      <a href=""> 2 </a>
    </li>
    <li>
      <a href=""> 3 </a>
    </li>
  </ul>
</menu>
```

Aquí los elementos hermanos son los list items (li), donde los links son hijos de estos, por lo que aplicaremos display:inline-block. Esta acción la podemos hacer dentro de la etiqueta style en el head o una hoja de estilos nueva.

```
li {
  display: inline-block;
}
```

Cuidado

No debemos cambiar las propiedades de todas las etiquetas <li>. Es mejor cambiar solo las que estén dentro del menú para evitar futuros problemas con CSS.

```
menu li {
  display: inline-block;
}
```

El selector anterior se lee como todos los li que estén dentro de <menu>, es poco probable que vayamos a tener otro menú con li, pero si así fuera, los podríamos distinguir agregando una clase y haciendo referencia a la clase en lugar de la etiqueta.

El resto del menú depende de lo que queramos lograr, es frecuente cambiar el color de fondo, asignarle un alto a la caja y un line-height para centrar verticalmente el elemento.

Todas estas propiedades las podemos agregar en el <ul> o en <menu>. Las agregaremos sobre el menú.

```
menu {
  background-color: black;
  color: white;
  line-height: 50px;
}

menu li{
  display: inline-block;
  margin-right: 50px;
}
menu a {
  color: white;
  text-decoration: none;
}
```

Finalmente, eliminamos los margin y padding por defecto de los elementos para que el menú quede en la parte superior del sitio.

```
body, menu {
 padding: 0px;
 margin: 0px;
}

menu ul{
 margin: 0px;
}
```

También es posible cambiar el color, o subrayado, e incluso pasar de minúsculas a mayúsculas el elemento apuntado con el mouse, esto lo lograremos utilizando la pseudo-selector :hover

¿Qué son las pseudo-selectores?

Son selectores que en lugar de apuntar a un elemento apuntan al estado de un elemento, como por ejemplo, en el caso de que el mouse esté encima de un elemento o no (:hover) o si el elemento fue visitado o no.
Esto lo podemos utilizar de la siguiente manera en nuestro menú.

```
menu a:hover{
```

```
  color: goldenrod;
}
```

Después de agregar lo anterior al CSS, abre la página en el navegador y pasa el mouse por encima de alguno de los links.

Existen dos tipos de pseudo-selectores, pseudo-clases y pseudo-elementos, aunque no es necesario memorizar esta distinción, podemos simplemente referirnos a ambos como pseudo-selectores.

Tipo de selector	Descripción	Ejemplos
Pseudo-clase	Se utiliza para seleccionar y aplicar estilos a elementos en función de su estado o relación con otros elementos en el documento.	:hover, :active, :focus
Pseudo-elemento	Se utiliza para seleccionar y aplicar estilos a partes específicas de un elemento.	::before, ::after, ::first-line

Tabla 01. Pseudoselectores.
Fuente: Desafío Latam.

Un uso de pseudo-elemento sería:

HTML:

```
<a href="#">Ejemplo de enlace</a>
```

CSS:

```
a::first-letter {
  font-size: 1.5em;
  font-weight: bold;
  color: red;
}
```

En este ejemplo, se aplican los estilos especificados al primer carácter del enlace. En particular, se establece el tamaño de fuente en 1.5 em, el peso de la fuente en negrita y el color del texto en rojo. Es importante tener en cuenta que el pseudoelemento::first-letter solo se puede utilizar con elementos que contienen texto, como <p>, <h1>, <a>, entre otros.

Actividad 1

Intenta replicar el ejercicio anterior de memoria, para ello, utiliza el autocompletado de vscode para apoyarte y Google u otro navegador para buscar las propiedades y los valores a agregar.

El problema de los inline-block

Cuando la suma de los anchos de los elementos mide el 100% es posible que el elemento que esté más a la derecha quede debajo. Esto se debe a que los espacios entre las etiquetas afectan el tamaño, lo que se puede resolver haciendo que la suma de un poco menor que 100%, por ejemplo 99.5% o removiendo el espacio en blanco.

Actividad 2

- Crear un nuevo archivo html y agregar lo siguiente al body.

```
<p> Lorem ipsum dolor sit, amet consectetur adipisicing elit. Cum deleniti explicabo non id voluptate, voluptatem animi enim soluta dolorum illo aliquam labore sed rerum odio, iusto, minus delectus nemo! Perspiciatis! </p>
<p> Lorem ipsum dolor sit, amet consectetur adipisicing elit. Cum deleniti explicabo non id voluptate, voluptatem animi enim soluta dolorum illo aliquam labore sed rerum odio, iusto, minus delectus nemo! Perspiciatis! </p>
```

- Agregar el siguiente CSS, puede ser en el head o un archivo externo.

```
p{
  display: inline-block;
  width: 50%;
}
```

- Al abrir la página web en el navegador debería aparecer un texto bajo el otro.
- Cambiar el width por 49% y abrir el navegador nuevamente.
    - Deberían quedar uno al lado del otro.
- Volver a poner 50%, remover todos los espacios en blanco en el html entre las etiquetas <p>
- Al abrir el navegador deberíamos poder ver los párrafos uno al lado del otro.

Flexbox

Flexbox (flexible layout) es un sistema que proporciona un modelo de cajas flexible para el posicionamiento de elementos en una página web. En palabras simples **nos permite poner de forma sencilla una caja al lado de otra caja.**

Un caso básico de uso sería:

```
<div style="display:flex">
  <p> Item 1 </p>
  <p> Item 2 </p>
  <p> Item 3 </p>
</div>
```

Imagen 3. Código.
Fuente: Desafío Latam.

Lo que nos daría como resultado:

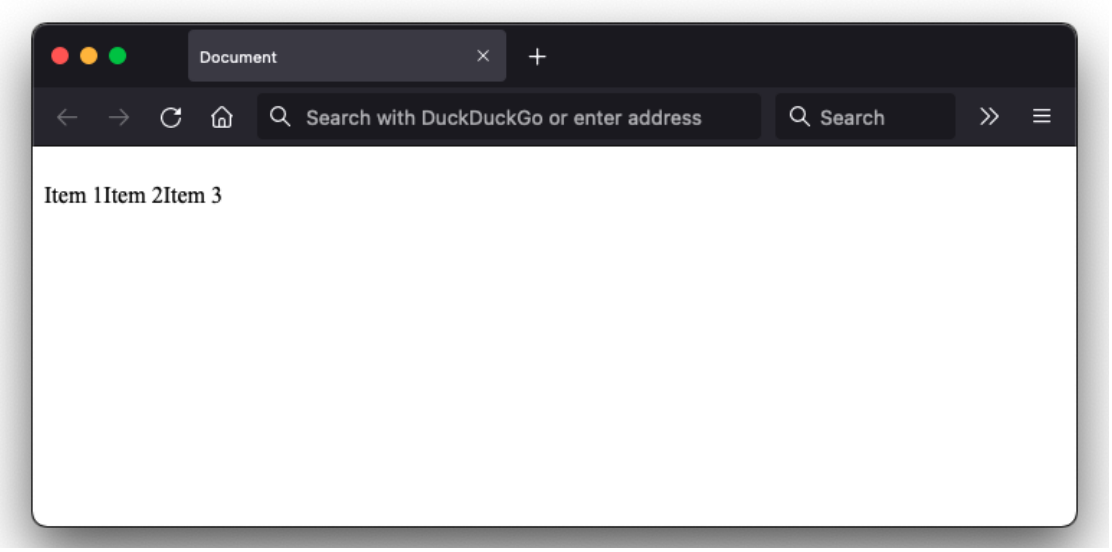

Imagen 4. Screenshot de ejemplo básico de flex.
Fuente: Desafío Latam.

Un detalle importante que podemos observar en el código html, es que la propiedad display con el valor flexbox se aplica al contenedor padre.

Alineando el contenido

La propiedad justify-content nos permite alinear el contenido dentro de un contenedor flex. Por ejemplo, probemos:

```
<div style="display:flex; justify-content:center">
  <p> Item 1 </p>
  <p> Item 2 </p>
  <p> Item 3 </p>
</div>
```

Imagen 5. Código.
Fuente: Desafío Latam.

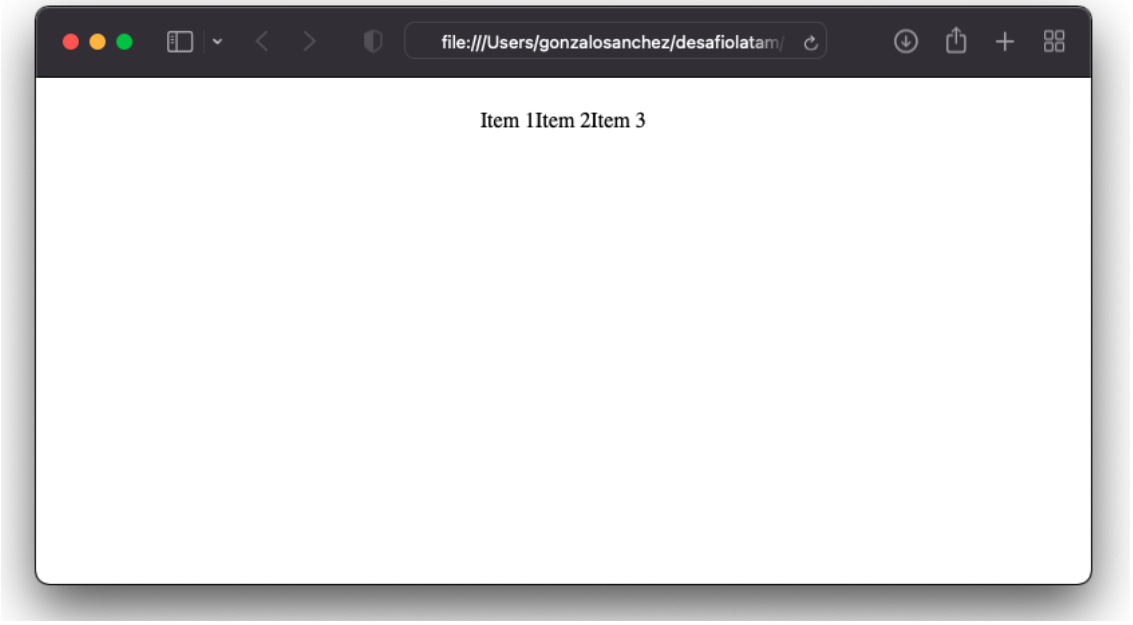

Imagen 6. Flex + justify-content.
Fuente: Desafío Latam.

Justify-content tiene varios valores posibles, la siguiente ilustración muestra las diferencias.

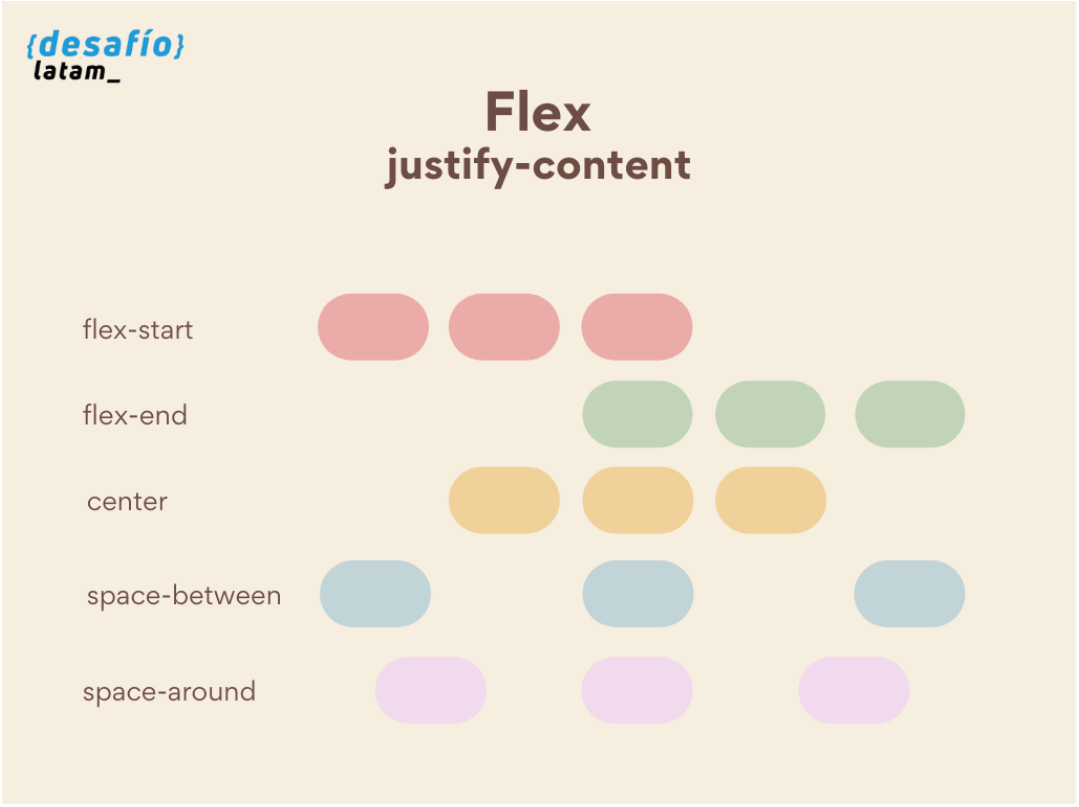

Imagen 7. Propiedad Justify-content.
Fuente: Desafío Latam.

Actividad 3: Alineando contenido con Flex

En el ejemplo anterior, donde se tiene un contenedor con display:flex y 3 párrafos:

- Prueba cambiando la propiedad justify-content del contenedor flex utilizando los valores de la ilustración, observa los resultados.
- También puedes usar el inspector de elementos para probar los cambios.

Actividad 4: Creando una galería de imágenes (parte 1)

Utiliza lo aprendido para crear una galería de imágenes, para lograrlo, tendrás que crear una nueva carpeta y el archivo index.html con una base de HTML.

Luego, puedes utilizar la siguiente estructura como base para el ejercicio:

```
<div style="display:flex">
```

```
<!-- Primer elemento de la galería -->
 <div>
   <img>
   <p>
</div>
<!-- Segundo elemento de la galería -->
 <div>
   <img>
   <p>
</div>
<!-- Tercer elemento de la galería -->
 <div>
   <img>
   <p>
</div>
</div>
```

Utiliza justify content para alinear el contenido de la galería, según estimes como mejor quedan los resultados.

Observaciones:

- Los flex-items también pueden ser un div u otro tipo de elemento. El primer div es el flex-container y sus elementos hijos son los flex-items.
- Puedes mover el CSS en un archivo externo o en el header.
- Si las imágenes son muy grandes puedes utilizar CSS para ajustarlas, puedes partir probando con agregar max-width: 100% sobre la imagen.

Actividad 5: Creando una galería de imágenes (parte 2)

Sobre el ejercicio de la actividad 4:

- Agrega 9 elementos más a la galería de imágenes y observa como se ve la página, deberían quedar a lo largo de una única fila, y los últimos elementos no podrán verse a menos que hagas scroll horizontal.
- Utiliza sobre el flex-container la propiedad flex-wrap: wrap y abre la página de nuevo, ahora deberías poder ver la galería de imágenes bien.

Actividad 6: Creando un menú con flex

- Crea una nueva carpeta llamada primer-menu y el archivo index.html con la base de HTML.
- Copia dentro del body las siguientes etiquetas

```
<ul>
  <li>
    <a href="#"> Inicio </a>
  </li>
  <li>
    <a href="#"> Pag 2 </a>
  </li>
  <li>
    <a href="#"> Pag 3 </a>
  </li>
</ul>
```

- Dentro del Head o un archivo de CSS externo agrega:

```
ul {display: flex}
```

- Utiliza CSS para eliminar los márgenes de forma que quede en la parte superior del sitio.
- Agrega un color de fondo al menú.
- Puedes también cambiar el color de los links, el text-decoration y el color.

HTML Semántico

Hasta el momento, para dividir nuestra página en distintas secciones hemos ocupado la etiqueta div, por ejemplo:

```
<div class="menu"> <!-- Menu --> </div>
<div class="header"> <!-- Contenido introductorio--> </div>
<div class="main"> <!-- Contenido principal --> </div>
<div class="footer"> <!-- Footer e información de contacto --> </div>
```

Ahora, aprenderemos que existen etiquetas específicas que podemos utilizar para este propósito; estas cumplen el mismo rol visual que el div, pero a su vez comunican su propósito a través del nombre.

Por ejemplo, en lugar de ahora utilizaremos la etiqueta <menu> y en lugar de utilizaremos <header>. Este tipo de etiquetas se llaman etiquetas semánticas.

Principales etiquetas semánticas

	<menu>
	<header>
	<main>
	<footer>
	<section>

Tabla 2. Etiquetas semánticas.
Fuente: Desafío Latam.

La semántica es el estudio de los significados. En el caso del ** **, este solo divide contenido, pero no se le atribuye ninguna carga de significado específica, en cambio, <menu> deja claramente establecido en el nombre que construiremos a continuación.

Ventajas de las etiquetas semánticas

- Comunicar correctamente a otros desarrolladores la organización del contenido en nuestro sitio web.
- Ayudar a que los motores de búsqueda en la web puedan indexar correctamente nuestros sitios impactando en su posicionamiento.

Estructura de un sitio con HTML semántico

No existe una estructura única de sitio web, hay varios arquetipos de estructura dependiendo del propósito, pero un sitio web estático probablemente tendrá una estructura como la siguiente:

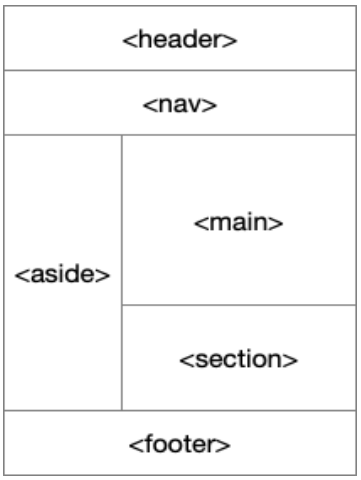

Imagen 8. Layout genérico de dos columnas.
Fuente: Desafío Latam.

La etiqueta semántica no le da la posición en el sitio, para lograr que un elemento o conjunto de elementos esté al lado de otro, debemos utilizar CSS con las técnicas aprendidas hasta ahora y que revisaremos más adelante.

Determinando que tipo de etiqueta semántica utilizar

El siguiente diagrama te ayudará en caso de que no tengas claridad que tipo de etiqueta utilizar:

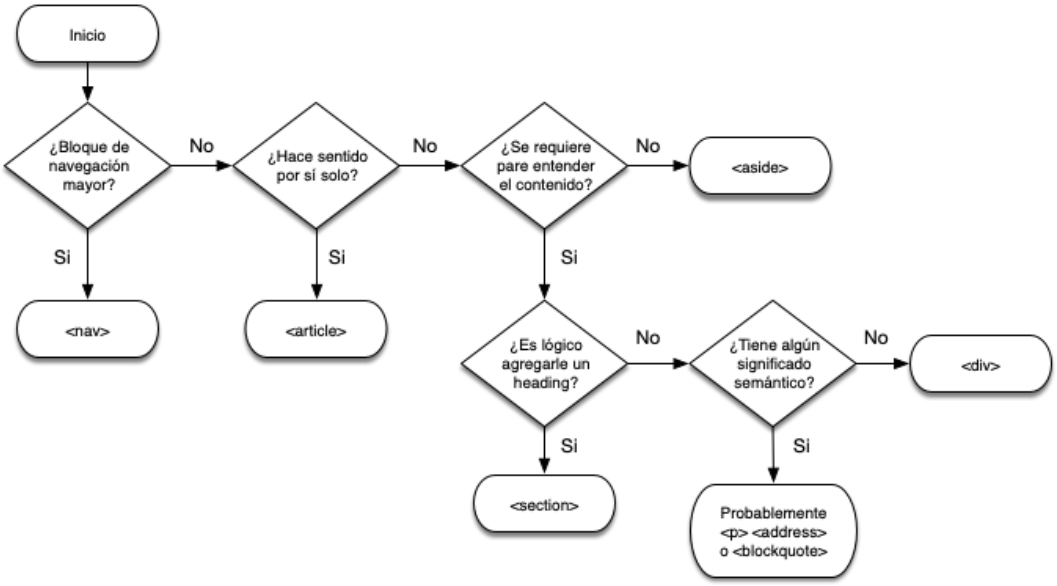

Imagen 9. Traducción del diagrama sectioning flowchart de html5doctor
Fuente: http://html5doctor.com/downloads/h5d-sectioning-flowchart.pdf

Antes de continuar:

- ¿Por qué se habla de etiquetas semánticas?
- ¿Por qué no es una etiqueta semántica?

Si pudiste responder con facilidad ambas preguntas, puedes seguir adelante, si alguna te presenta dificultad, asegúrate de aclarar los puntos que más te hayan costado.

Resumen de HTML Semántico

- La etiqueta se utiliza para crear divisiones dentro de una página web, pero no tiene semántica.
- HTML5 tiene etiquetas que cumplen la misma función de dividir que el ** **, pero que le atribuyen semántica al contenido, como **<nav>**, **<header>**, **<section>** y **<footer>**, entre otras.
- **<nav>** se usa para bloques de navegación.
- **<header>** se usa para realizar una introducción al sitio.
- **<main>** se utiliza para la sección principal del sitio.
- **<section>** se usa para identificar secciones independientes.
- **<footer>** para definir un pie de página.