

FrontEndI

Módulo 3. Estructuración avanzada

Sesión 4. Cajas flexibles

Flexbox utiliza una estructura de filas y columnas. Es soportado por las versiones comerciales de todos los navegadores web desde 2015.

Un contenedor Flex posee dos ejes: el eje principal (main axis) y el eje transversal (cross axis).



La propiedad Flex-direction permite definir el main axis del contenedor:

`flex-direction: row;` // Los items se disponen en el eje x, de izquierda a derecha. Es el valor por defecto.

`flex-direction: row-reverse;` // Los items se disponen en el eje x, de derecha a izquierda.

`flex-direction: column;` // Los items se disponen en el eje y, de arriba a abajo.

`flex-direction: column-reverse;` // Los items se disponen en el eje y, de abajo a arriba.

Para alinear los elementos a través del main axis se usa la propiedad `justify-content`, y para alinearlos a través del cross axis se utiliza `align-items`:

`justify-content: flex-start;` // Valor por defecto. Los items se ubican al inicio del main axis.

`justify-content: flex-end;` // Los items se ubican al final del main axis.

`justify-content: center;` // Los items se alinean en el centro del main axis.

`justify-content: space-between;` // Los items se distribuyen de manera uniforme.

`justify-content: space-around;` // Los items se distribuyen de manera uniforme, dejando un margen al inicio y un margen al final.

`align-items: stretch;` // Valor por defecto. Los items ocupan todo el espacio en el cross axis.

`align-items: flex-start;` // Los items se alinean al inicio del cross axis.

`align-items: flex-end;` // Los items se alinean al final del cross axis.

`align-items: center;` // Los items se alinean al centro del cross axis.

En casos de contenedores de una sola línea se establece la propiedad `flex-flow` con el valor `no-wrap` y se emplea la propiedad `align-items`. En el caso de contenedores multilínea se establece la propiedad `flex-flow` con los valores `wrap` o `wrap-reverse` y se utiliza `align-content`. `Align-content` admite los siguientes valores:

`align-content: flex-start;`

`align-content: flex-end;`

`align-content: center;`

`align-content: stretch;`

`align-content: space-between;`

`align-content: space-around;`

Estructura básica de Flexbox

Flexbox propone una estructura basada en el uso de un contenedor padre (`flex-container`) y sus elementos hijos (`flex-items`).

```
.contenedor-padre {  
    display: flex; // También puede adoptar el valor inline-flex.  
    flex-wrap: wrap; // Para que el flex-container respete el ancho de los  
flex-items.  
}
```

Items

`order: 1;` // Controla el orden en el que aparecen los items, independiente del orden establecido en el HTML. El valor por defecto es cero.

`Flex-grow: 0,75;` // Establece cuánto puede crecer un elemento si tiene espacio libre.

align-self: flex-end / flex-start, etc // Permite alinear el item sobre el cross axis, independiente del valor establecido por align-items.

Sincrónico

justify-content: Cómo se alinean los elementos en el eje principal

Box-sizing: border-box; // Generalmente se le aplica a todo el documento.

Flex-box soluciona problemas del position. Normalmente no se requieren usar en forma conjunta.

Es mala práctica usar las dos si se quiere usar solo una de ellas; solamente se aplicaría la que está más abajo.

Sticky y fixed tienen problemas de compatibilidad. Se recomienda que en proyectos grandes no se use, aunque normalmente se presentan más problemas con fixed.

Recomendación

Se recomienda usar el normalize? No lo recomienda.

Pone en blanco la página; sin márgenes o paddings predeterminados, etc.

Una forma de 'normalizar' (revisar):

```
* {  
  margin: 0;  
  padding: 0;  
  border: 0;  
  font-size: 100%;  
  font: inherit;  
  vertical-align: baseline;  
}
```

Para que las medidas relativas funcionen mejor, lo ideal es trabajar siempre con contenedores.

Recomendación: Utilizar medidas relativas, especialmente porcentajes.

Absolute se usa muy poco en pocos elementos; no se debe plantear el layout de todo un sitio con absolute.

Flex-box (revisar):

display: flex;

Casos:

```
justify-content: flexend;  
justify-content: center;  
justify-content: space-around;  
justify-content: space-between;
```

align-items: flex-end; // Ubica en el eje secundario

Flex-box es ideal para armar layout
Flex-box es responsive.

Ej: para centrar un elemento, se combinan las dos propiedades, align-items y justify-content.

Ejercicios css:

<https://mastery.games/flexboxzombies/>
<https://flexboxfroggy.com/#es>
<http://www.flexboxdefense.com/>
<https://flukeout.github.io/>

Sesión 14. Diseño adaptativo.

Julio 1 de 2021.

Viewport

Etiqueta viewport:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

Medidas relativas

Son aquellas que tienen en cuenta el contexto en el que se encuentran. Si el contexto cambia, estas medidas cambiarán con él.

- **Porcentajes:** relacionada con la medida del elemento padre en el mismo eje. No se recomienda usar porcentajes para el ancho de un elemento.

- **rem y em.** em toma como referencia la propiedad font-size del elemento padre. rem se calcula siempre con base en el font-size del elemento de referencia inicial, que es el <html>, cuyo font-size es 16px. Se recomienda usar rem en lugar de em.

- **vw y vh.** Medidas expresadas como porcentaje de las dimensiones del viewport.

Media Queries

Son un conjunto de reglas de CSS que permiten cambiar los estilos de los elementos en función de las características del dispositivo que esté visualizando el sitio.

```
@media (min-width: 460px){
```

```
  body {  
    background: red;  
  }
```

```
}
```

Orientación:

```
@media (max-width: 460px) and (orientation: landscape){  
  body {
```

```
        background: blue;
    }
}
```

Estrategia de diseño: Mobile first. Recomendado.

Determinar de manera general las reglas CSS para pequeñas pantallas para luego, a través de media queries, ir aclarando el comportamiento en viewports más grandes.

```
body {
    background: red;
}
```

```
@media (min-width: 460px){
/* Tablets */
}
```

```
@media (min-width: 768px){
/* Laptop */
}
```

Breakpoints: Puntos de quiebre a partir de los cuales cambian las reglas de estilo.

Los más utilizados:

0 – 480px
481 – 768px
769 – 1279px
1280 -

Sincrónico

Analizar página de Nike.

Alinear botones a la izquierda para facilitar el uso con el pulgar

Para que no se vea alguna de las imágenes:

display: none
visibility:
width: 0%

Figma:

<https://www.figma.com/file/wSyAvMIFnSM7eE3fufsm3I/Clase-14?node-id=0%3A1>

Código de referencia:

<https://github.com/juan351/heroes>

Sesión 15

Sincrónico

Pensar primero en los teléfonos móviles: "Mobile first".

Ver la página de Puma.

Se utiliza JavaScript para mostrar un número alto de elementos en una galería. Algunos de los elementos quedan extra-canvas.

Flex-wrap: Permite que los elementos fluyen a la siguiente línea. (Revisar).

Figma:

<https://www.figma.com/file/31NtnGFVE8XyUbfA8Esktw/Petshop?node-id=0%3A1>

Pendiente:

- Completar clase 9 (PetShop)
- Completar Clase 11 (Pizza)
- Clase 12 (PetShop, flex)
- Clase 14 (Héroes, v2, flex, responsive)
- Clase 15 (Petshop, responsive)

visibility: hidden; // Oculta el elemento, pero sigue ocupando espacio

display: none; // Oculta el elemento y deja de ocupar espacio

Github Agustina:

<https://github.com/agustinagarciaey/TpPetShop>