Nombre: Rodrigo Mena Serna

Asignatura: Diseño de Interfaces Web

Segundo curso de Desarrollo de Aplicaciones Web

Nombre de la práctica: Documentación Flexbox

Contenido

Flexbox

Flexbox es un método de diseño unidimensional para organizar elementos en filas o columnas. Los elementos se flexionan (expanden) para llenar espacio adicional o se encogen para caber en espacios más pequeños.

¿Por qué Flexbox?

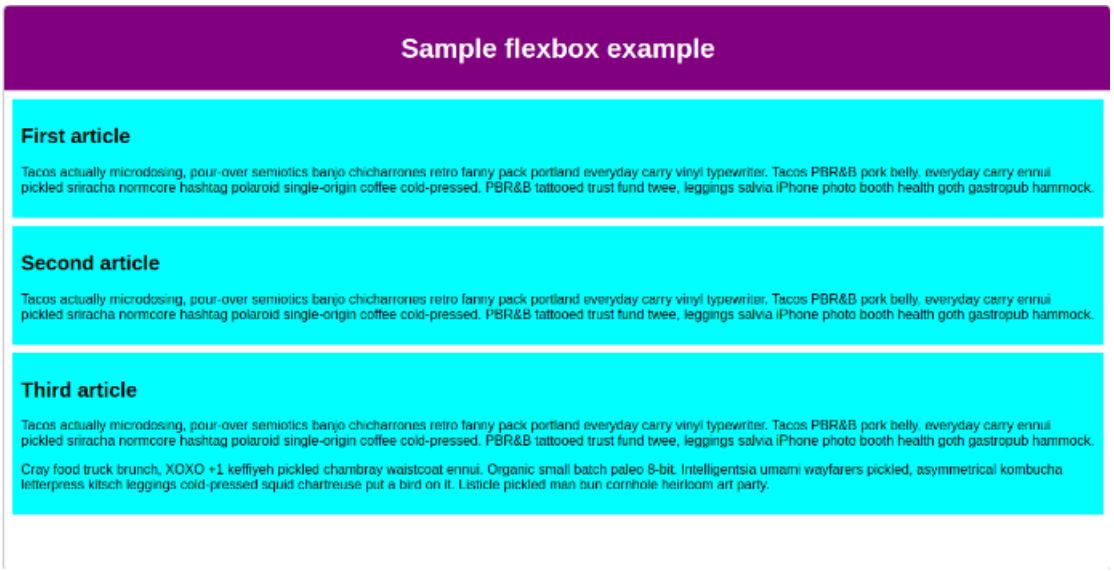
Durante mucho tiempo, las únicas herramientas confiables compatibles con varios navegadores disponibles para crear diseños CSS eran características como floats y positioning. Estos funcionan, pero de alguna manera también son limitantes y frustrantes.

Los siguientes diseños de diseños simples o imposibles de lograr con tales herramientas de cualquier forma conveniente y flexible:

- Centrar verticalmente un bloque de contenido dentro de su elemento principal.
- Hacer que todos los elementos secundarios de un contendor ocupen la misma cantidad de ancho/alta disponible, independientemente de cuánto ancho/alto haya disponible.
- Hacer que todas las columnas en un diseño de varias columnas adopten la misma altura incluso si contienen una cantidad diferente de contenido.

Ejemplo sencillo

Tenemos un elemento <header> con un encabezado de nivel superior dentro y un elemento <section> que contiene tres <article>. Vamos a usarlos para crear un diseño de tres columnas bastante estándar.

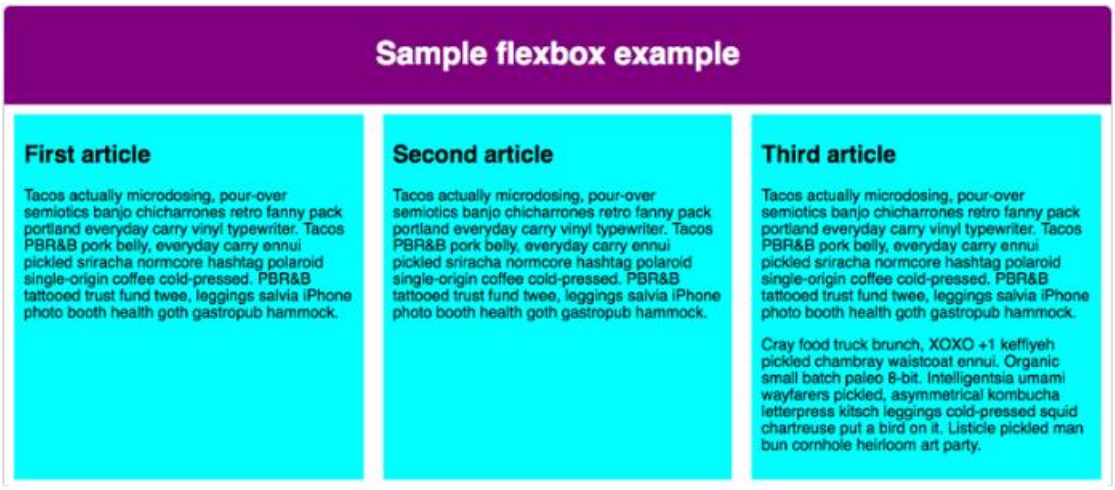


Especificar qué elementos distribuir como cajas flexibles

Para empezar, debemos seleccionar qué elementos se distribuirán como cajas flexibles. Para hacer esto, establecemos un valor especial de display en el elemento principal de los elementos que desea afectar. En este caso, queremos diseñar los elementos <article>, por lo que estableceremos esto en <section>:

```
section {
  display: flex;
}
```

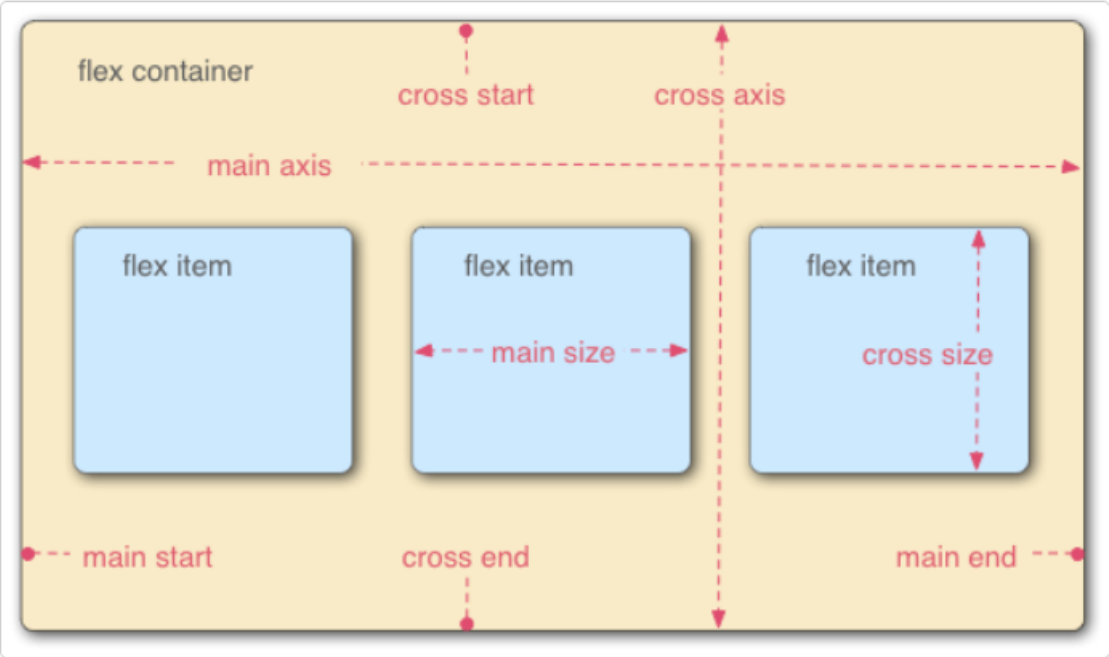
Esto hace que el elemento <section> se convierta en un contenedor flexible y sus elementos secundarios se conviertan en elementos flexibles. El resultado de esto debería ser algo así:



Para ser claros, reiteremos lo que está sucediendo aquí. El elemento al que le hemos dado un display valor flex actúa como un elemento de nivel de bloque en términos de cómo interactúa con el resto de la página, pero sus elementos secundarios se presentan como elementos flexibles. La siguiente sección explicará con más detalle lo que esto significa. Tenga en cuenta también que puede usar display valor de inline-flex si desea diseñar los elementos secundarios de un elemento como elementos flexibles, pero hacer que ese elemento se comporte como un elemento en línea.

El modelo flexible

Cuando los elementos se disponen como elementos flexibles, se disponen a lo largo de dos ejes:



- El eje principal es el eje que corre en la dirección en la que están dispuestos los elementos flexibles (por ejemplo, como una fila en la página o una columna en la página). El inicio y el final de este eje se denominan inicio principal y fin principal.
- El eje transversal es el eje que corre perpendicular a la dirección en la que se colocan los elementos flexibles. El inicio y el final de este eje se denominan inicio y final transversales.
- El elemento principal que se ha display: flexestablecido en él (<section>en nuestro ejemplo) se llama contenedor flexible.
- Los elementos dispuestos como cajas flexibles dentro del contenedor flexible se denominan elementos flexibles (los <article>elementos de nuestro ejemplo).

¿Columnas o filas?

Flexbox proporciona una propiedad denominada flex-directionque especifica en qué dirección corre el eje principal (en qué dirección se disponen los hijos de flexbox). De forma predeterminada, está configurado en row, lo que hace que se coloquen en una fila en la dirección en la que funciona el idioma predeterminado de su navegador (de izquierda a derecha, en el caso de un navegador en inglés).

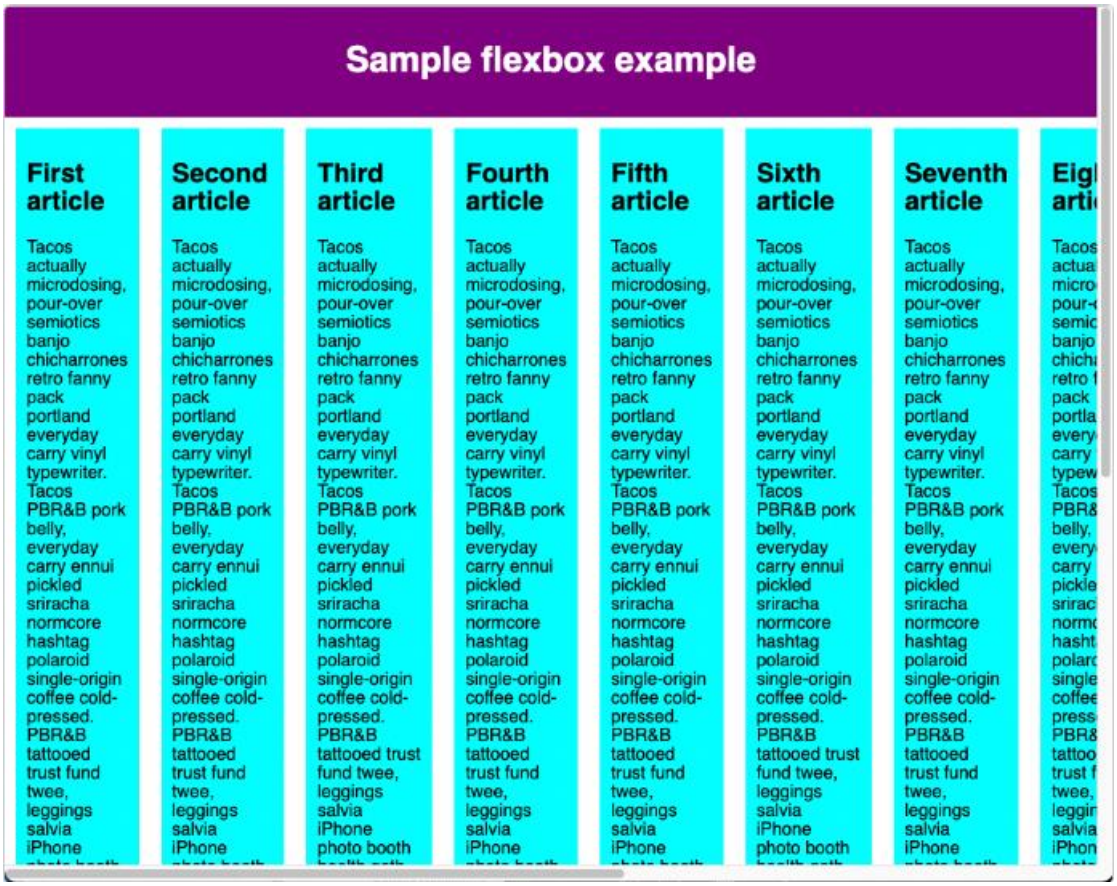
Intente agregar la siguiente declaración a su <section>:

```
flex-direction: column;
```

Verá que esto vuelve a colocar los elementos en un diseño de columna, como estaban antes de que agregáramos cualquier CSS.

Wrapping

Un problema que surge cuando tiene un ancho o una altura fijos en su diseño es que, eventualmente, sus elementos secundarios flexbox desbordarán su contenedor, rompiendo el diseño.



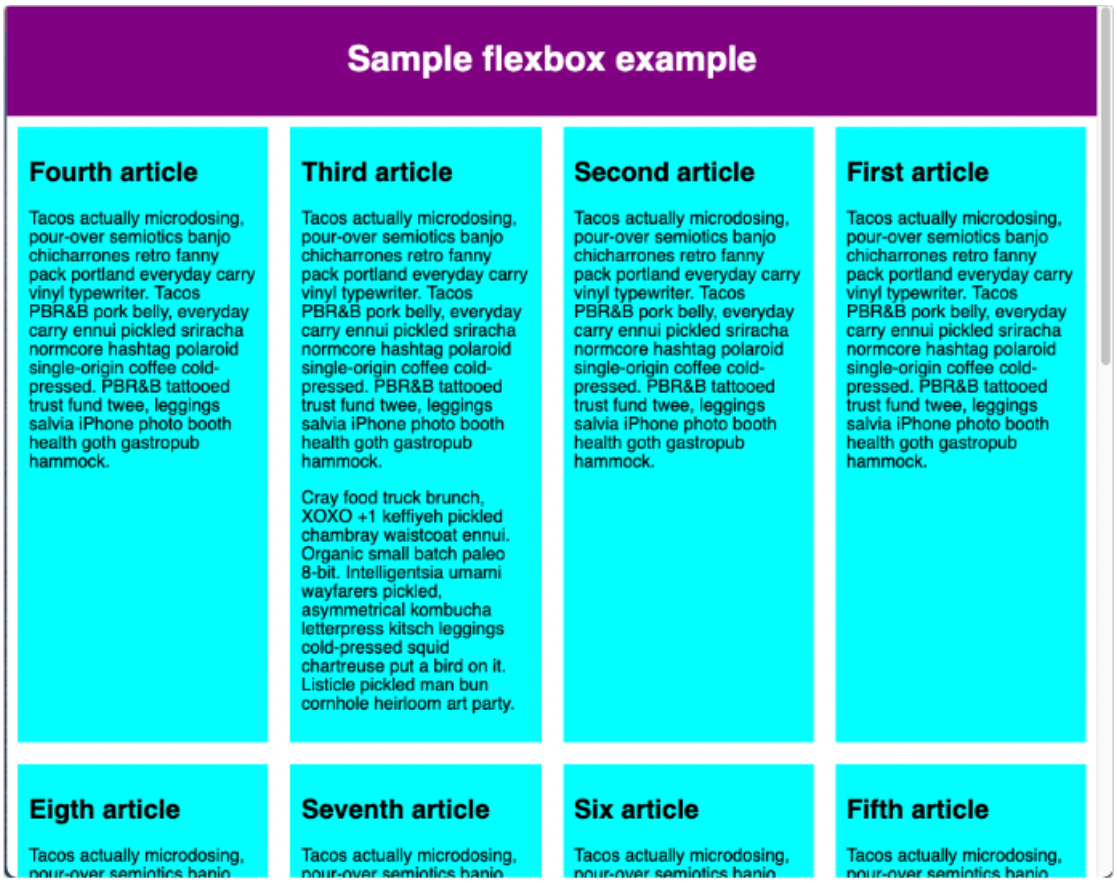
Aquí vemos que los niños de hecho están saliendo de su contenedor. Una forma de solucionar esto es agregar la siguiente declaración a su <section>:

```
flex-wrap: wrap;
```

Además, agregue la siguiente declaración a su <article>:

```
flex: 200px;
```

Prueba esto ahora. Verás que el diseño se ve mucho mejor con esto incluido:



Ahora tenemos varias filas. Cada fila tiene tantos elementos secundarios flexbox como sea sensato. Cualquier desbordamiento se mueve a la siguiente línea. La flex: 200pxdeclaración establecida en los artículos significa que cada uno tendrá al menos 200 px de ancho. Hablaremos de esta propiedad con más detalle más adelante. También puede notar que los últimos elementos secundarios en la última fila se hacen más anchos para que toda la fila aún se llene.

Pero hay más que podemos hacer aquí. En primer lugar, intente cambiar el valor de flex-direction de su propiedad a row-reverse. Ahora verá que todavía tiene su diseño de filas múltiples, pero comienza desde la esquina opuesta de la ventana del navegador y fluye en sentido inverso.

flex-flow shorthand

En este punto, vale la pena señalar que existe una forma abreviada de flex-directiony flex-wrap: flex-flow. Entonces, por ejemplo, puede reemplazar

```
flex-direction: row;
flex-wrap: wrap;
```

Con

```
flex-flow: row wrap;
```

Tamaño flexible de elementos flexibles

```
article {
  flex: 1;
}
```

Este es un valor de proporción sin unidades que dicta cuánto espacio disponible a lo largo del eje principal ocupará cada elemento flexible en comparación con otros elementos flexibles. En este caso, estamos dando a cada <article>elemento el mismo valor (un valor de 1), lo que significa que todos ocuparán la misma cantidad de espacio libre que queda después de que se hayan establecido propiedades como padding y margin. Este valor se comparte proporcionalmente entre los elementos flexibles: dar a cada elemento flexible un valor de 400000 tendría exactamente el mismo efecto.

Ahora agregue la siguiente regla debajo de la anterior:

```
article:nth-of-type(3) {
  flex: 2;
}
```

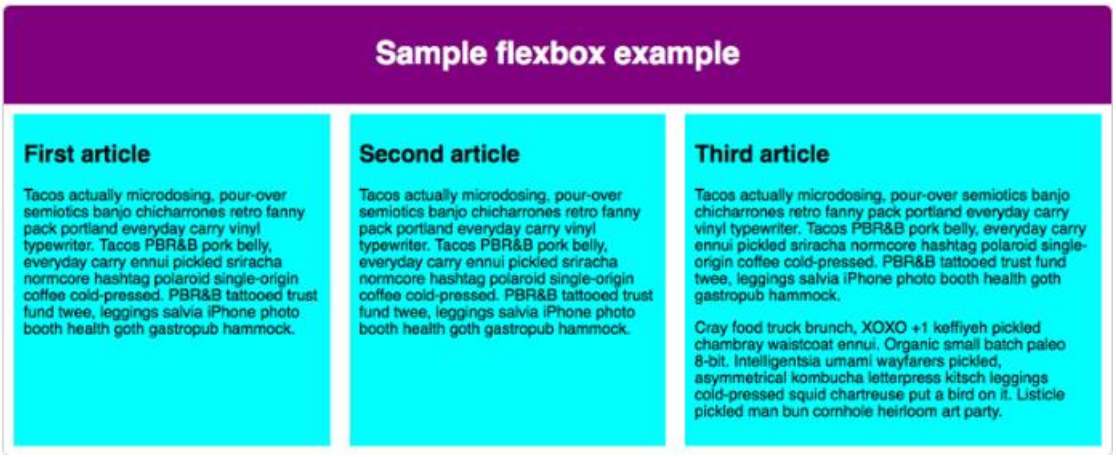
Ahora, cuando actualice, verá que el tercero <article>ocupa el doble del ancho disponible que los otros dos. Ahora hay cuatro unidades de proporción disponibles en total (ya que 1 + 1 + 2 = 4). Los dos primeros elementos flexibles tienen una unidad cada uno, por lo que cada uno ocupa 1/4 del espacio disponible. El tercero tiene dos unidades, por lo que ocupa 2/4 del espacio disponible (o la mitad).

También puede especificar un valor de tamaño mínimo dentro del valor flexible. Intente actualizar sus reglas de artículos existentes de esta manera:

```
article {
  flex: 1 200px;
}

article:nth-of-type(3) {
  flex: 2 200px;
}
```

Esto básicamente dice: "Cada elemento flexible recibirá primero 200 px del espacio disponible. Después de eso, el resto del espacio disponible se compartirá de acuerdo con las unidades de proporción". Intente actualizar y verá una diferencia en cómo se comparte el espacio.



El valor real de flexbox se puede ver en su flexibilidad/capacidad de respuesta. Si cambia el tamaño de la ventana del navegador o agrega otro <article>elemento, el diseño continúa funcionando correctamente.

flex: shorthand versus longhand

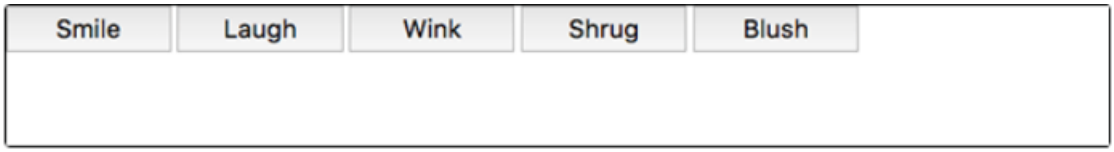
flex es una propiedad abreviada que puede especificar hasta tres valores diferentes:

- El valor de proporción sin unidades que discutimos anteriormente. Esto se puede especificar por separado usando la flex-grow propiedad longhand.
- Un segundo valor de proporción sin unidades, flex-shrink, que entra en juego cuando los elementos flexibles desbordan su contenedor. Este valor especifica cuánto se reducirá un elemento para evitar el desbordamiento. Esta es una característica bastante avanzada de flexbox y no la cubriremos más en este artículo.
- El valor de tamaño mínimo que discutimos anteriormente. Esto se puede especificar por separado usando el flex-basis valor manual.

Recomendamos no usar las propiedades flex manuales a menos que realmente tenga que hacerlo (por ejemplo, para anular algo establecido previamente). Conducen a que se escriba una gran cantidad de código adicional y pueden ser algo confusos.

Alineación horizontal y vertical

En este momento, verá una barra de menú horizontal con algunos botones atascados en la esquina superior izquierda.



Ahora, agregue lo siguiente al final del CSS del ejemplo:

```
div {
  display: flex;
  align-items: center;
  justify-content: space-around;
}
```

Actualice la página y verá que los botones ahora están bien centrados horizontal y verticalmente. Hemos hecho esto a través de dos nuevas propiedades.

align-items controla dónde se ubican los elementos flexibles en el eje transversal.

- De forma predeterminada, el valor es stretch, que estira todos los elementos flexibles para llenar el padre en la dirección del eje transversal. Si el padre no tiene una altura fija en la dirección del eje transversal, todos los elementos flexibles serán tan altos como el elemento flexible más alto. Así es como nuestro primer ejemplo tenía columnas de igual altura por defecto.
- El valor center que usamos en nuestro código anterior hace que los elementos mantengan sus dimensiones intrínsecas, pero estén centrados a lo largo del eje transversal. Esta es la razón por la que los botones de nuestro ejemplo actual están centrados verticalmente.
- También puede tener valores como flex-starty flex-end, que alinearán todos los elementos al principio y al final del eje transversal, respectivamente. Ver align-items para los detalles completos.

Puede anular el align-items comportamiento de elementos flexibles individuales aplicándoles la propiedad align-self. Por ejemplo, intente agregar lo siguiente a su CSS:

```
button:first-child {
  align-self: flex-end;
}
```

justify-content controla dónde se ubican los elementos flexibles en el eje principal.

- El valor predeterminado es flex-start, lo que hace que todos los elementos se sitúen al comienzo del eje principal.
- Puedes usar flex-end para hacer que se sienten al final.
- Center es también un valor para justify-content. Hará que los elementos flexibles se asienten en el centro del eje principal.
- El valor que hemos usado anteriormente, space-around, es útil: distribuye todos los elementos de manera uniforme a lo largo del eje principal con un poco de espacio en cada extremo.
- Hay otro valor, space-between, que es muy similar a space-around excepto que no deja ningún espacio en ninguno de los extremos.

La propiedad justify-items se ignora en los diseños de flexbox.

Ordering flex ítems

Flexbox también tiene una función para cambiar el orden de diseño de los elementos flexibles sin afectar el orden de origen. Esta es otra cosa que es imposible de hacer con los métodos de diseño tradicionales.

El código para esto es simple. Intente agregar el siguiente CSS a su código de ejemplo de la barra de botones:

```
button:first-child {
  order: 1;
}
```

Actualice y verá que el botón "Sonrisa" se ha movido al final del eje principal. Hablemos de cómo funciona esto con un poco más de detalle:

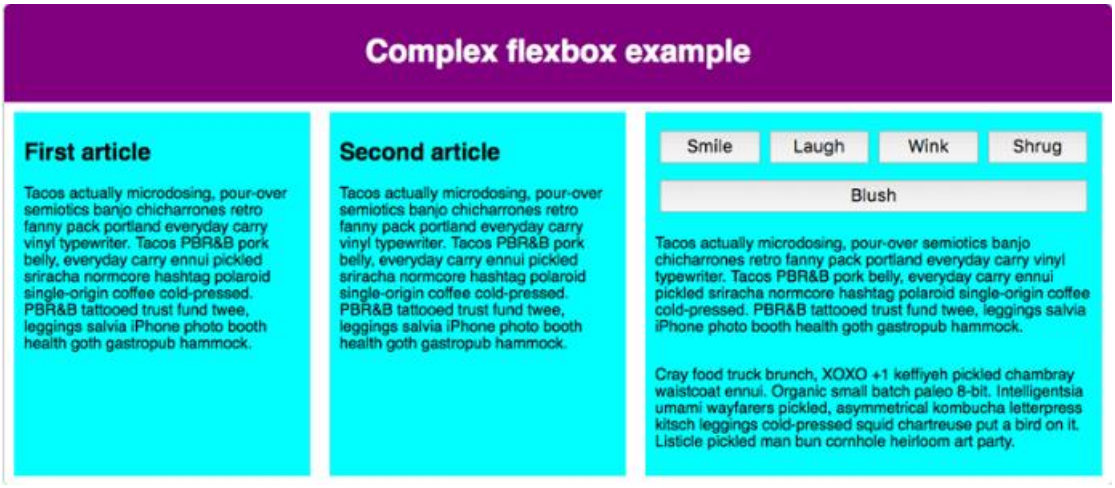
- De forma predeterminada, todos los elementos flexibles tienen un order valor de 0.
- Los artículos flexibles con valores de pedido especificados más altos aparecerán más tarde en el orden de visualización que los artículos con valores de pedido más bajos.
- Los artículos flexibles con el mismo valor de pedido aparecerán en su pedido de origen. Entonces, si tiene cuatro elementos cuyos valores de orden se han establecido como 2, 1, 1 y 0 respectivamente, su orden de visualización sería 4, 2, 3 y luego 1.
- El tercer artículo aparece después del segundo porque tiene el mismo valor de pedido y está después en el pedido de origen.

Puede establecer valores de orden negativos para que los elementos aparezcan antes que los elementos cuyo valor es 0. Por ejemplo, puede hacer que el botón "Blush" aparezca al comienzo del eje principal utilizando la siguiente regla:

```
button:last-child {
  order: -1;
}
```

Cajas flexibles anidadas

Es posible crear algunos diseños bastante complejos con flexbox. Está perfectamente bien configurar un elemento flexible para que también sea un contenedor flexible, de modo que sus elementos secundarios también se presenten como cajas flexibles.



El HTML para esto es bastante simple. Tenemos un elemento <section> que contiene tres <article>s. El tercero <article> contiene tres s, y el primero contiene cinco <button>s:

```
section - article
          article
          article - div - button
                    div   button
                    div   button
                          button
                          button
```

Veamos el código que hemos usado para el diseño.

En primer lugar, configuramos los elementos secundarios de <section> para que se presenten como cajas flexibles.

```
section {
  display: flex;
}
```

A continuación, establecemos algunos valores de flexión en los propios <article> s. Tome nota especial de la segunda regla aquí: estamos configurando la tercera <article> para que sus

elementos secundarios también se presenten como elementos flexibles, pero esta vez los colocaremos como una columna.

```
article {
  flex: 1 200px;
}

article:nth-of-type(3) {
  flex: 3 200px;
  display: flex;
  flex-flow: column;
}
```

A continuación, seleccionamos el primero . Primero usamos flex: 1 100px; para darle efectivamente una altura mínima de 100 px, luego configuramos sus elementos secundarios (los elementos <button>) para que también se presenten como elementos flexibles. Aquí los colocamos en una fila envolvente y los alineamos en el centro del espacio disponible como hicimos con el ejemplo de botón individual que vimos anteriormente.

```
article:nth-of-type(3) div:first-child {
  flex: 1 100px;
  display: flex;
  flex-flow: row wrap;
  align-items: center;
  justify-content: space-around;
}
```

Finalmente, establecemos un tamaño en el botón. Esta vez dándole un valor flexible de 1 auto. Esto tiene un efecto muy interesante, que verá si intenta cambiar el tamaño del ancho de la ventana del navegador. Los botones ocuparán tanto espacio como puedan. En una línea caben tantos como sea cómodo; más allá de eso, bajarán a una nueva línea.

```
button {
  flex: 1 auto;
  margin: 5px;
  font-size: 18px;
  line-height: 1.5;
}
```

Bibliografía:

https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Flexbox#nested_flex_boxes