



Creciendo y menguando

Curso de Fundamentos

Introducción

Veamos un poco más de cerca lo que realmente sucedió cuando colocaste **flex**:
1 esos elementos flexibles en la última lección.

Resumen de la lección

Esta sección contiene una descripción general de los temas que aprenderá en esta lección.

- Aprenderá las 3 propiedades que se definen mediante la **flex** abreviatura y cómo usarlas individualmente.

La abreviatura flex

La **flex** declaración es en realidad una abreviatura de tres propiedades que puedes configurar en un elemento flexible. Estas propiedades afectan el tamaño de los elementos flexibles dentro de su contenedor. Ya has visto algunas propiedades abreviadas antes, pero aún no las hemos definido oficialmente.

“Las propiedades abreviadas son propiedades CSS que permiten establecer los valores de varias otras propiedades CSS simultáneamente. Con una propiedad abreviada, puede escribir hojas de estilo más concisas (y, a menudo, más legibles), lo que le permitirá ahorrar tiempo y energía.”

Fuente: [Propiedades de Shorthand en MDN](#)

En este caso, `flex` es en realidad una abreviatura de `flex-grow`, `flex-shrink` y `flex-basis`.

```
div {  
  flex: 1;  
}
```

En la captura de pantalla anterior, `flex: 1` equivale a: `flex-grow: 1`, `flex-shrink: 1`, `flex-basis: 0`.

Muy a menudo, se ve la abreviatura `flex` definida con *un* solo valor. En ese caso, ese valor se aplica a `flex-grow`. Por lo tanto, cuando colocamos `flex: 1` a nuestros `divs`, en realidad estábamos especificando una abreviatura de `flex: 1 1 0`.

Cultivo flexible

`flex-grow` espera un único número como valor, y ese número se utiliza como el "factor de crecimiento" del elemento flexible. Cuando aplicamos esto `flex: 1` a cada `div` dentro de nuestro contenedor, le estábamos diciendo a cada `div` que creciera la misma cantidad. El resultado de esto es que cada `div` termina teniendo exactamente el mismo tamaño. Si, en cambio, agregamos `flex: 2` solo a uno de los `divs`, entonces ese `div` crecería hasta el doble del tamaño de los otros.

En el siguiente ejemplo, la `flex` abreviatura tiene valores para `flex-shrink` y `flex-basis` especificados con sus valores predeterminados.

HTML

CSS

Result

EDIT ON

LIVE

```
<div class="flex-container">
  <div class="one"></div>
  <div class="two"></div>
  <div class="three"></div>
</div>
```

Resources1x0.5x0.25xRerun

Flexión-encogimiento

`flex-shrink` es similar a `flex-grow`, pero establece el "factor de encogimiento" de un elemento flexible. `flex-shrink` solo se termina aplicando si el tamaño de todos los elementos flexibles es mayor que su contenedor principal. Por ejemplo, si nuestros 3 divs de arriba tuvieran una declaración de ancho como: `width: 100px`, y `.flex-container` fueran más pequeños que `300px`, nuestros divs tendrían que encogerse para caber.

El factor de contracción predeterminado es `flex-shrink: 1`, lo que significa que todos los elementos se encogerán de manera uniforme. Si no *desea* que un elemento se encoja, puede especificar `flex-shrink: 0`. También puede especificar números más altos para que ciertos elementos se encojan a un ritmo mayor que el normal.

Here's an example. Note that we've also changed the `flex-basis` for reasons that will be explained shortly. If you shrink your browser window you'll notice that `.two` never gets smaller than the given width of 250px, even though the `flex-grow` rule would otherwise specify that each element should be equally sized.

HTML

CSS

Result

EDIT ON

LIVE

```
<div class="flex-container">
  <div class="one"></div>
  <div class="two"></div>
  <div class="three"></div>
</div>
```

Resources1x0.5x0.25xRerun

An important implication to notice here is that when you specify `flex-grow` or `flex-shrink`, flex items do not necessarily respect your given values for `width`. In the above example, all 3 divs are given a width of 250px, but when their parent is big enough, they grow to fill it. Likewise, when the parent is too small, the default behavior is for them to shrink to fit. This is not a bug, but it could be confusing behavior if you aren't expecting it.

Flex-basis

`flex-basis` sets the initial size of a flex item, so any sort of `flex-grow`ing or `flex-shrink`ing starts from that baseline size. The shorthand value defaults to `flex-basis: 0%`. The reason we had to change it to `auto` in the `flex-shrink` example is that with the basis set to `0`, those items would ignore the item's width, and everything would shrink evenly. Using `auto` as a flex-basis tells the item to check for a width declaration (`width: 250px`).

Important note about flex-basis

There is a difference between the default value of `flex-basis` and the way the `flex` shorthand defines it if no `flex-basis` is given. The actual default value for `flex-basis` is `auto`, but when you specify `flex: 1` on an element, it interprets that as `flex: 1 1 0`. If you want to *only* adjust an item's `flex-grow` you can do so directly, without the shorthand. Or you can be more verbose and use the full 3 value shorthand `flex: 1 1 auto`, which is also equivalent to using `flex: auto`.

What is flex auto?

If you noticed, we mentioned a new flex shorthand `flex: auto` in the previous note. However we didn't fully introduce it. `flex: auto` is one of the shorthands of flex. When `auto` is defined as a flex keyword it is equivalent to the values of `flex-grow: 1`, `flex-shrink: 1` and `flex-basis: auto` or to `flex: 1 1 auto` using the flex shorthand. Note that `flex: auto` is not the default value when using the flex shorthand despite the name being "auto" which may be slightly confusing at first. You will encounter and learn more about `flex: auto` and its potential use-cases when reading through the assignment section.

In practice

In practice you will likely not be using complex values for `flex-grow`, `flex-shrink` or `flex-basis`. Generally, you're most likely to use declarations like `flex: 1;` to make divs grow evenly and `flex-shrink: 0` to keep certain divs from shrinking.

Es posible ser sofisticado y configurar diseños en los que algunas columnas se relacionan entre sí en una proporción específica, por lo que es útil saber que se pueden usar otros valores, pero son relativamente raros.

Asignación

1. Lea [la sección flex de W3C](#) para comprender los valores básicos de las abreviaturas flex más comunes.
2. [La documentación de MDN sobre flex](#) resume todos los valores abreviados de flex y también presenta alguna sintaxis nueva que no se ha tratado en el artículo anterior.

Comprobación de conocimientos

Las siguientes preguntas son una oportunidad para reflexionar sobre temas clave de esta lección. Si no puede responder una pregunta, haga clic en ella para revisar el material, pero tenga en cuenta que no se espera que memorice o domine este conocimiento.

- [¿Cuáles son los 3 valores definidos en la flex propiedad abreviada \(por ejemplo flex: 1 1 auto\)?](#)

- [¿Cuáles son los 3 valores definidos para la abreviatura flex flex:auto?](#)

Recursos adicionales

Esta sección contiene enlaces útiles a contenido relacionado. No es obligatoria, por lo que se la puede considerar complementaria.

- Un [vídeo que explora cómo funciona Flexbox y por qué](#).
- Para obtener una explicación y una demostración interactivas, consulte el [Scrim sobre la abreviatura flex](#). Para obtener una explicación alternativa, puede ver el [Scrim sobre el uso de flex-grow, flex-shrink y flex-basis](#). Tenga en cuenta que estos Scrim requieren iniciar sesión en Scrimba para poder verlos.

 [Mejorar en GitHub](#)

 [Informar un problema](#)

[Ver registro de cambios de la lección](#)



Ver curso

Marcar como completado



Próxima lección

¡Apóyanos!

El Proyecto Odin está financiado por la comunidad. ¡Únase a nosotros para ayudar a estudiantes de todo el mundo apoyando el Proyecto Odin!

Más información

Dona ahora



Educación en codificación de alta calidad mantenida por una comunidad de código abierto.



Sobre nosotros

[Acerca de](#)

[Equipo](#)

[Blog](#)

[Casos de éxito](#)

Apoyo

[Preguntas frecuentes](#)

[Contribuir](#)

[Contáctenos](#)

Guías

[Guías de la comunidad](#)

[Guías de instalación](#)

Legal

[Términos](#)

[Privacidad](#)

© 2025 El Proyecto Odin. Todos los derechos reservados.