Alineación

Curso de Fundamentos

Introducción

Hasta ahora, todo lo que hemos hecho con flexbox ha utilizado la regla `flex: 1` de todos los elementos flexibles, lo que hace que los elementos crezcan o se encojan de manera uniforme para llenar todo el espacio disponible. Sin embargo, muy a menudo, este no es el efecto deseado. Flex también es muy útil para organizar elementos que tienen un tamaño específico.

Resumen de la lección

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

- Aprenderá a alinear elementos dentro de un contenedor flexible tanto vertical como horizontalmente.

Alineación

Veamos un ejemplo.

HTML CSS Result EDIT ON

LIVE

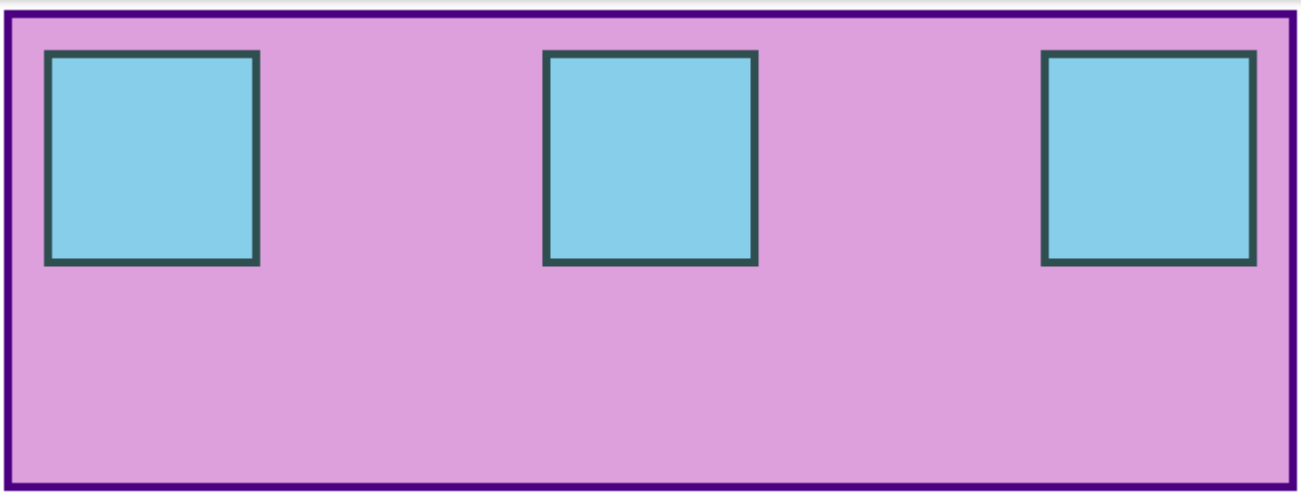
```
<div class="container">
  <div class="item"></div>
  <div class="item"></div>
  <div class="item"></div>
</div>
```

Resources 1× 0.5× 0.25× Rerun

A estas alturas ya deberías poder predecir lo que sucederá si te pones `flex: 1` el dispositivo `.item`. ¡Pruébalo antes de continuar!

`flex: 1` Al sumar, `.item` cada uno de los elementos crece hasta llenar el espacio disponible, pero ¿qué pasa si queremos que mantengan el mismo ancho, pero que se distribuyan de forma diferente dentro del contenedor? ¡Podemos hacerlo!

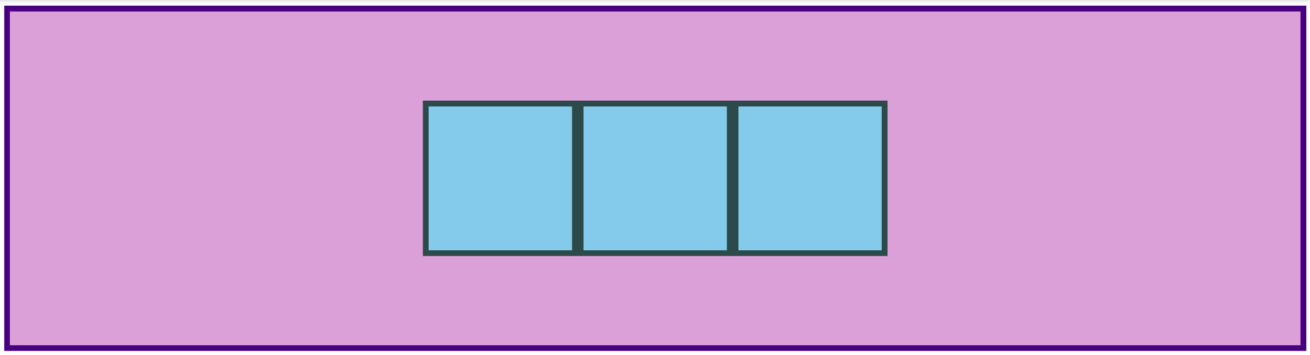
Quitar `flex: 1` de `.item` y agregar `justify-content: space-between` a `.container`. Al hacerlo, debería obtener algo como esto:



`justify-content` alinea los elementos a lo largo del **eje principal**. Hay algunos valores que puedes usar aquí. Aprenderás el resto en las tareas de lectura, pero por ahora

intenta cambiarlo a `center`, lo que debería centrar los cuadros a lo largo del eje principal.

Para cambiar la ubicación de los elementos a lo largo del **eje transversal**, utilice `align-items`. Intente colocar los cuadros en el centro del contenedor agregando `align-items: center` a `.container`. El resultado deseado se ve así:



Debido a que `justify-content` y `align-items` se basan en el eje principal y transversal de su contenedor, su comportamiento cambia cuando cambia la dirección flexible de un contenedor flexible. Por ejemplo, cuando cambia `flex-direction` a `column`, `justify-content` se alinea verticalmente y `align-items` se alinea horizontalmente. Sin embargo, el comportamiento más común es el predeterminado, es decir, `justify-content` alinea los elementos horizontalmente (porque el eje principal está predeterminado en horizontal) y `align-items` los alinea verticalmente. Uno de los mayores problemas que tienen los principiantes con flexbox es la confusión cuando este comportamiento cambia.

Brecha

Una característica muy útil de flex es la `gap` propiedad. La configuración `gap` en un contenedor flexible agrega un espacio específico entre los elementos flexibles, de manera similar a agregar un margen a los elementos mismos. `gap` es una propiedad *nueva*, por lo que aún no aparece en muchos recursos, pero funciona de manera confiable en todos los navegadores modernos, por lo que es segura de usar y muy útil. Al agregarla `gap: 8px` al ejemplo centrado anterior, se obtiene el resultado que se muestra a continuación.

HTML CSS Result EDIT ON

LIVE

```
<div class="container">
  <div class="item"></div>
  <div class="item"></div>
  <div class="item"></div>
</div>
```

Resources 1× 0.5× 0.25× Rerun

Hay más cosas que aprender en la siguiente lectura, pero en este punto seguramente podrá ver lo inmensamente útil que es flexbox. ¡Con solo las propiedades que ya cubrimos, ya podría armar algunos diseños impresionantes!

Take your time going through the reading. There will be some review of the items we've already covered here, but it goes into more depth and touches on a few things that haven't been mentioned yet. Don't stress too much about trying to memorize every little detail yet; just code along with the examples and do your best to internalize everything that is *possible* with flexbox. You'll have to reach for these resources again once you get to the practice exercises, but that's perfectly acceptable. The more you use this stuff the better it will stick in your mind... and you will be using it *constantly*. Have fun!

Assignment

1. This beautiful Interactive Guide to Flexbox covers everything you need to know. It will help reinforce concepts we've already touched on with some really fun and creative examples. Spend some time here, some of it should be review at this point, but the foundations here are important!
2. Typical use cases of Flexbox is an MDN article that covers some more practical tips. Don't skip the interactive sections! Playing around with this stuff is how you learn it!
3. The CSS Tricks "Guide to Flexbox" is a classic. The images and examples are super helpful. It would be a good idea to review parts 1-3 and part 5

(don't worry about the media query parts, we will cover them later in the course) and then bookmark it as a great cheat sheet for future reference (keep it handy for the practice exercises).

4. Do the exercises in our CSS exercises repository's `foundations/flex` directory (remember that the instructions are in the README) in the order:

   - `01-flex-center`
   - `02-flex-header`
   - `03-flex-header-2`
   - `04-flex-information`
   - `05-flex-modal`
   - `06-flex-layout`
   - `07-flex-layout-2`

   Note: Solutions for these exercises can be found in the `solution` folder of each exercise.

Knowledge check

The following questions are an opportunity to reflect on key topics in this lesson. If you can't answer a question, click on it to review the material, but keep in mind you are not expected to memorize or master this knowledge.

- What is the difference between `justify-content` and `align-items`?
- How do you use flexbox to completely center a div inside a flex container?
- What's the difference between `justify-content: space-between` and `justify-content: space-around`?

Additional resources

This section contains helpful links to related content. It isn't required, so consider it supplemental.

- Flexbox Froggy is a funny little game for practicing moving things around with flexbox.

- Flexbox Zombies is another gamified take on flexbox. Free, but requires an account.
- The Basic Concepts of Flexbox article on MDN is another good starting point. There are helpful examples and interactive sections.
- Aligning Items in a Flex Container goes into more depth on the topic of axes and `align-items` vs `justify-content`.
- Este tutorial de Flexbox de freecodecamp es otro recurso decente.
- Flexbox Crash Course es un buen recurso de Traversy Media.
- Para ver demostraciones más interactivas, prueba Scrim en la `justify-content` propiedad y Scrim en la `align-items` propiedad. Ten en cuenta que para ver estos Scrims es necesario iniciar sesión en Scrimba.

THE ODIN PROJECT

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