# Cómo combinar cuadrículas Flexbox y CSS para diseños eficientes

https://getflywheel.com/layout/combine-flexbox-and-css-grids-for-layouts-how-to/

[](https://getflywheel.com/layout/author/abbeyfitzgerald/)

[Abbey Fitzgerald](https://getflywheel.com/layout/author/abbeyfitzgerald/)- 15 de diciembre de 2018

En el pasado, las propiedades flotantes de CSS eran uno de los principales métodos para organizar elementos en un sitio web. Y si alguna vez ha trabajado de esa manera, sabe que no siempre es ideal para diseños complejos. Afortunadamente, en la era moderna del diseño web, la alineación de elementos se ha simplificado mucho gracias a las cuadrículas Flexbox y CSS.

Cuando apareció Flexbox, facilitó mucho la alineación y desde entonces ha sido ampliamente adoptado. [Los diseños de cuadrícula CSS](https://getflywheel.com/layout/css-grid-layouts-how-to/) también han creado mucho entusiasmo en la comunidad de diseño web. Hace un tiempo, echamos un vistazo a cómo crear un [diseño de cuadrícula CSS](https://getflywheel.com/layout/css-grid-layouts-how-to/) básico . Aunque no se ha adoptado de forma generalizada, los navegadores están empezando a admitirlo. Cuando sea totalmente compatible, esto tendrá un gran impacto en los diseños. El soporte del navegador está aumentando todo el tiempo; asegúrese de consultar [Can I use](_blank)? para obtener la información más actualizada.

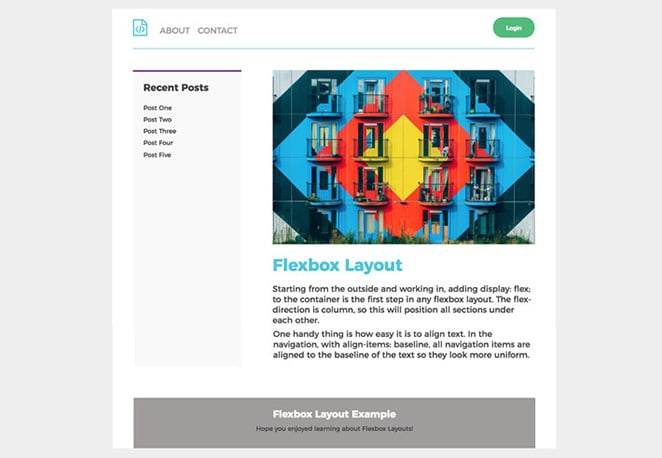
Ahora puede que se pregunte qué sigue; después de todo, Flexbox y CSS Grid Layouts parecen lograr resultados similares. Sin embargo, no es un debate entre Flexbox y Grid, sino más bien aprender a usarlos juntos. Cuanto más he jugado con Grid y Flexbox, he descubierto que no tienes que elegir solo uno u otro. En un futuro próximo, cuando los diseños de cuadrícula CSS tengan compatibilidad total con el navegador, los diseñadores podrán aprovechar las ventajas combinadas de cada uno y crear los diseños más eficientes e interesantes.

## Prueba de diseños de cuadrícula básicos de Flexbox y CSS

Para determinar si Flexbox o CSS Grid funcionan mejor para su flujo de trabajo de desarrollo, crear un diseño estándar que solo use uno u otro es una buena manera de ver cómo funcionan y si hay ventajas de uno sobre el otro. Comenzaremos con un tipo de diseño muy simple y familiar con un encabezado, barra lateral, contenido principal y pie de página. Un diseño simple como este es una forma rápida de colocar los distintos elementos.

## Cómo crear un diseño con Flexbox

Recientemente, cubrí el tema de la creación de un [diseño de tarjeta Flexbox](https://getflywheel.com/layout/how-to-use-flexbox-to-create-a-modern-card-design-layout/) . Esa publicación detalla cómo funciona Flexbox junto con información específica de CSS, por lo que si eres un principiante en Flexbox, te ayudará a familiarizarte con su funcionamiento.



Para este tutorial, esto es lo que crearemos:

Vea esto en [Codepen](https://codepen.io/abbeyjfitzgerald/pen/jwqrZY) .

Para este diseño básico, las principales tareas de Flexbox incluyen:

* Crear encabezado y pie de página de ancho completo
* Coloque la barra lateral junto al área de contenido principal
* Tamaño correcto de la barra lateral y el área de contenido principal
* Posicionamiento de elementos de navegación

### Estructura HTML básica

|  |
| --- |
| <div class="container">        <header>          <nav>            <ul>              <li></li>              <li></li>              <li></li>            </ul>          </nav>          <button></button>      </header>        <div class="wrapper">          <aside class="sidebar">              <h3></h3>          </aside>          <section class="main">              <h2></h2>              <p></p>          </section>      </div><!-- /wrapper -->        <footer>          <h3></h3>          <p></p>      </footer>  </div><! -- /container --> |

### Cómo utilizar la pantalla flexible

Estilo de encabezado

Comenzando desde el exterior y trabajando hacia adentro, agregar display: flex;al contenedor es el primer paso en cualquier diseño de Flexbox. La dirección de flexión se establece en columna, por lo que esto colocará todas las secciones una debajo de la otra.

|  |
| --- |
| .container {      display: flex;      flex-direction: column;  } |

Crear un encabezado de ancho completo fue bastante automático con el display: flex;(los encabezados son un elemento de nivel de bloque por defecto). Debido a esta declaración, permitirá una fácil ubicación de los elementos de navegación.

Hay un logotipo y dos elementos de menú en la navegación a la izquierda con un botón de inicio de sesión a la derecha. La navegación está en el encabezado, por lo que al tener justify-content: space-between;la navegación y el botón, se espaciarán automáticamente.

Una cosa útil es lo fácil que es alinear el texto. En la navegación, con align-items: baseline;, todos los elementos de navegación se alinean con la línea de base del texto para que se vean más uniformes.



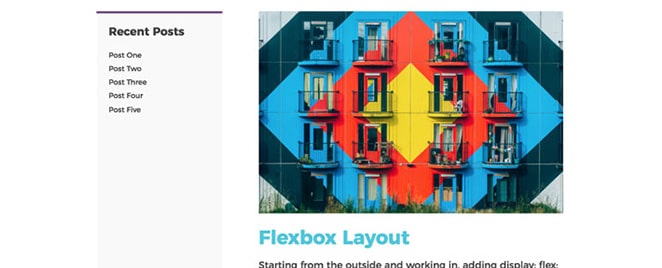
|  |
| --- |
| header{      padding: 15px;      margin-bottom: 40px;      display: flex;      justify-content: space-between;  }    header nav ul {      display: flex;      align-items: baseline;      list-style-type: none;  } |

Estilo de contenido de la página

A continuación, está la barra lateral y las áreas de contenido principal con un contenedor que incluye los dos. El div con una clase de .wrappertambién necesita el display: flex;pero la dirección de flexión es diferente a la anterior. Debido a que la barra lateral y las áreas de contenido están una al lado de la otra en lugar de apiladas, la dirección flexible es

A continuación, está la barra lateral y las áreas de contenido principal con un contenedor que incluye los dos. El div con una clase de .wrapper también necesita la pantalla: flex; pero la dirección de flexión es diferente a la anterior. Debido a que la barra lateral y las áreas de contenido están una al lado de la otra en lugar de apiladas, la dirección flexible es la fila, que es lo opuesto a lo que se hizo en el contenedor anterior.

|  |
| --- |
| .wrapper {      display: flex;      flex-direction: row;  } |



El tamaño de la sección principal y la barra lateral es muy importante ya que aquí vive información más prominente. El contenido principal debe ser tres veces el tamaño de la barra lateral, lo cual es bastante fácil de hacer con Flexbox.

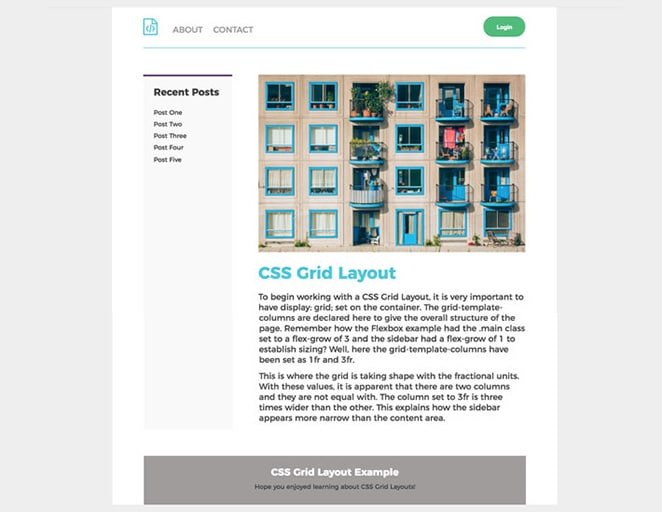
|  |
| --- |
| .main {      flex: 3;      margin-right: 60px;  }    .sidebar {     flex: 1;  } |

Para este fragmento de código, utilicé taquigrafía. Los valores de flex son para la propiedad flex-grow. Flex-grow es poderoso porque esto es cuánto crecerán los artículos en relación con el resto de los artículos flexibles dentro del mismo contenedor.

En general, Flexbox fue bastante eficiente en la creación de este diseño simple. Fue especialmente útil tener control sobre el estilo de los elementos de la lista y el espacio entre la navegación y el botón.

## Cómo crear un diseño con diseños de cuadrícula CSS

Para probar la eficiencia, el siguiente paso es crear el mismo diseño básico con CSS Grid. Los elementos de la página son todos iguales y se colocarán de la misma manera que en el ejemplo de Flexbox.



Vea esto en [Codepen](https://codepen.io/abbeyjfitzgerald/pen/qjZNpz) .

### Áreas de plantilla de cuadrícula

Una cosa útil con CSS Grid es la capacidad de especificar áreas de plantilla, lo que puede hacer que la definición de diseños sea muy intuitiva. Al adoptar este enfoque, las áreas de la cuadrícula se pueden nombrar y hacer referencia a elementos de posición. Para este diseño básico, hay cuatro elementos que necesitaremos nombrar:

* encabezamiento
* contenido principal
* barra lateral
* pie de página

### Estructura HTML básica

|  |
| --- |
| <div class="container">        <header>          <nav>            <ul>              <li></li>              <li></li>              <li></li>            </ul>          </nav>          <button></button>      </header>        <aside class="sidebar">          <h3></h3>          <ul>              <li></li>              <li></li>      <li></li>      <li></li>      <li></li>            </ul>      </aside>        <section class="main">          <h2></h2>          <p></p>           <p> </p>    </section>        <footer>          <h3></h3>          <p></p>      </footer>    </div> |

Definiremos estas áreas en nuestro contenedor de cuadrícula en orden, como dibujarlas. También los espaciaré para facilitar la lectura.

|  |
| --- |
| grid-template-areas:          "header header"          "sidebar main"          "footer footer"; |

¿Tenga en cuenta cómo la barra lateral se enumeró antes de principal? Cambiarlos también haría que el orden cambiara en la página. Actualmente, la barra lateral está a la izquierda y el contenido principal a la derecha, pero puede cambiarlo fácilmente si es necesario.

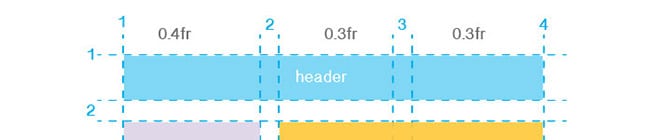
Una cosa a tener en cuenta: estos nombres deben estar "conectados" al estilo. Solo porque se han declarado áreas de plantilla de cuadrícula, no sabemos a dónde pertenece realmente el encabezado. En el bloque de encabezado, grid-area: header;debe agregarse.

|  |
| --- |
| header{      grid-area: header;      padding: 20px 0;      display: grid;      grid-template-columns: 1fr 1fr;  } |

La estructura HTML es la misma que en el ejemplo de Flexbox, pero el CSS es bastante diferente para crear el diseño de cuadrícula.

|  |
| --- |
| .container{      max-width: 900px;      background-color: #fff;      margin: 0 auto;      padding: 0 60px;      display: grid;      grid-template-columns: 1fr 3fr;      grid-template-areas:          "header header"          "sidebar main"          "footer footer";      grid-gap: 50px;  } |

Para comenzar a trabajar con un diseño de cuadrícula CSS, es muy importante haber display: grid;configurado el contenedor. Las columnas de plantilla de cuadrícula se declaran aquí para dar la estructura general de la página. ¿Recuerda cómo el ejemplo de Flexbox tenía la .mainclase configurada en un flex-grow de 3 y la barra lateral tenía un flex-grow de 1 para establecer el tamaño? Aquí, las columnas de plantilla de cuadrícula se han establecido como 1fr y 3fr. Aquí es donde va tomando forma la cuadrícula con las unidades fraccionarias. Con estos valores, es evidente que hay dos columnas y no tienen el mismo ancho. La columna configurada en 3fr es tres veces más ancha que la otra. Esto explica cómo la barra lateral parece más estrecha que el área de contenido.



A continuación, las unidades fr utilizadas para el contenedor deben ajustarse para el encabezado. Las columnas de la plantilla de la cuadrícula se han ajustado a 1fr y 1fr. De esa manera, hay dos columnas de igual tamaño y los elementos de navegación y el botón encajarán.

|  |
| --- |
| header{      grid-area: header;      display: grid;      grid-template-columns: 1fr 1fr;  } |



Para colocar el botón, solo necesitamos usar justify-self y configurarlo para que finalice.

|  |
| --- |
| header button {      justify-self: end;  } |

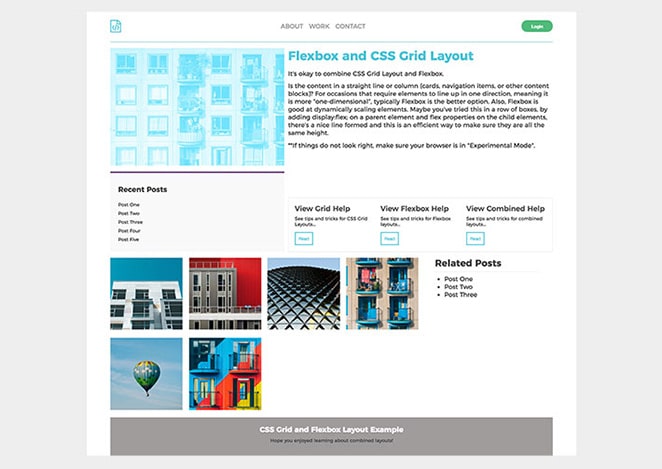
La navegación se coloca donde debe estar:

|  |
| --- |
| header nav {      justify-self: start;  } |

El pie de página de ancho completo no necesita que se establezcan diferentes columnas, ya que el contenido está en el medio.

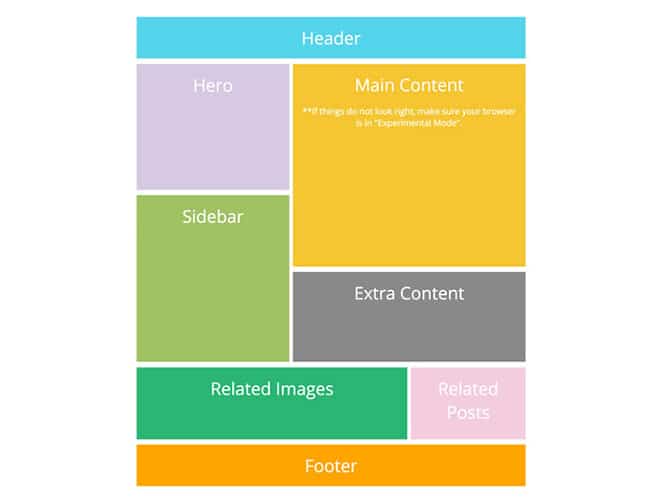
## Cómo crear un diseño con Flexbox y CSS Grids

Ahora que hemos visto lo que cada método puede hacer individualmente, es hora de crear algo más complejo combinando Flexbox y CSS Grid Layouts.



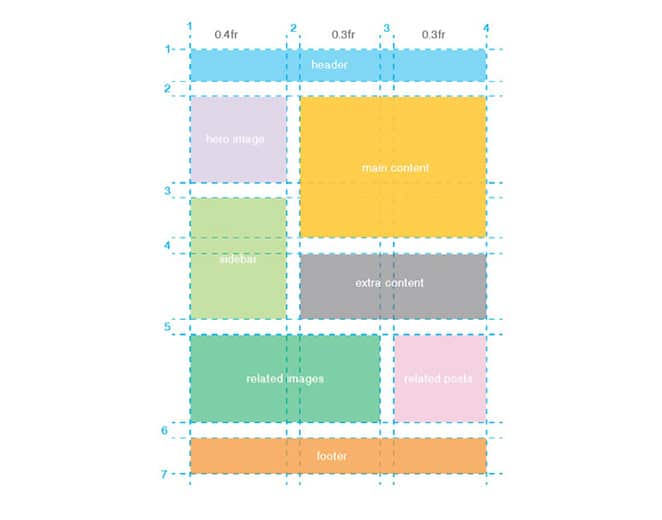
Vea esto en [Codepen](https://codepen.io/abbeyjfitzgerald/pen/RgRoKy) .

Aquí está el esquema básico para poner en marcha la cuadrícula:



Vea esto en [Codepen](https://codepen.io/abbeyjfitzgerald/pen/zzqqYo) .

¿Observa cómo el diseño se basa tanto en columnas como en filas? Este diseño necesita que las cosas se alineen y se comporten de manera consistente en ambas direcciones, por lo que el uso de CSS Grid es eficiente para el diseño general.



La planificación es clave con un diseño como este. Es una buena idea esbozarlo primero y ver cómo se acumulan las cosas, literalmente. Para iniciar el código, tener display: grid; es esencial; sin él, el uso de este tipo de diseño no funcionará. Una cosa a tener en cuenta aquí es que hay un espacio entre los bloques de contenido. Esto se logró con grid-column-gap y grid-row-gap.

|  |
| --- |
| .container {    display: grid;    grid-template-columns: 0.4fr 0.3fr 0.3fr;    grid-column-gap: 10px;    grid-row-gap: 15px;  } |

Las unidades fraccionales están de vuelta para este diseño y ahora se necesitan tres áreas. El primer valor de 0,4 fr es ligeramente más ancho que el segundo y el tercero, ambos de 0,3 fr.

### Diseño de columnas y filas

Aquí es donde es importante hacer referencia al diagrama desde el principio. Comenzando por la parte superior, así es como se coloca el encabezado. Abarca todas las columnas y una fila.

|  |
| --- |
| .header {    grid-column-start: 1;    grid-column-end: 4;    grid-row-start: 1;    grid-row-end: 2;    background-color: #d5c9e2;  } |

Si desea utilizar la abreviatura, los valores inicial y final están en la misma línea y separados por una barra. Se vería así:

|  |
| --- |
| .header {    grid-column: 1 / 4;    grid-row: 1 / 2;    background-color: #55d4eb;  } |

Para colocar todos los demás elementos, solo es necesario agregar los valores adecuados de la cuadrícula y la columna al CSS. En lugar de [revisarlos](https://codepen.io/abbeyjfitzgerald/pen/zzqqYo) aquí uno por uno, este ejemplo está en [Codepen](https://codepen.io/abbeyjfitzgerald/pen/zzqqYo) .

Una vez creado el diseño de cuadrícula, el siguiente paso es ajustar el contenido.

Navegación

Flexbox es perfecto para colocar los elementos del encabezado. El ejemplo de diseño básico tocó esto con justify-content: space-between. El ejemplo de la cuadrícula necesitaba tener justify-self: start; sobre la navegación y justificarse: fin; para que el botón coloque cosas, pero Flexbox facilitó el espaciado para la navegación.

|  |
| --- |
| .header {    grid-column: 1 / 4;    grid-row: 1 / 2;    color: #9f9c9c;    text-transform: uppercase;    border-bottom: 2px solid #b0e0ea;    padding: 20px 0;    display: flex;    justify-content: space-between;    align-items: center;  } |



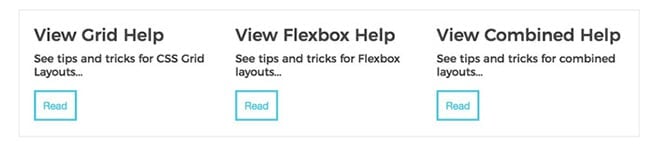
Este mismo formato se seguirá aquí. El logotipo, los elementos del menú y el botón utilizaron el contenido de justificación de Flexbox para el espaciado.

Cuadrícula de contenido de columna

Para ocasiones que requieren que los elementos se alineen en una dirección, lo que significa que es más "unidimensional", normalmente Flexbox es la mejor opción. Además, Flexbox es bueno para escalar elementos dinámicamente. Tal vez haya intentado esto en una fila de cuadros agregando display:flex;un elemento principal y propiedades flexibles en los elementos secundarios. Con esa técnica, se forma una buena línea y es una forma eficiente de asegurarse de que todos los elementos tengan la misma altura.

Contenido de fila con texto y botones

En la sección "contenido extra", se han agregado tres áreas con texto y botones. Flexbox facilita mantener el ancho establecido para tres columnas.



|  |
| --- |
| .extra {    grid-column: 2 / 4;    grid-row: 4 / 5;    padding: 1rem;    display: flex;    flex-wrap: wrap;    border: 1px solid #ececec;    justify-content: space-between;  } |

### Una excepción de Flexbox

Sí, dije que Flexbox es mejor para diseños, cuadrículas o columnas unidimensionales, pero si lees la [publicación Cómo usar Flexbox para crear una publicación de diseño de tarjeta CSS moderna](https://getflywheel.com/layout/how-to-use-flexbox-to-create-a-modern-card-design-layout/) , hubo un truco de Flexbox de "última fila" que se demostró para mantener el equilibrio de filas y columnas, incluso sin un número par de tarjetas.



|  |
| --- |
| .related-images {    grid-column: 1 / 3;    grid-row: 5 / 6;    display: grid;    grid-template-columns: repeat(4,1fr);    grid-gap: 1rem;  }    .related-images .icon {      background-color: white;      flex: 1 1 150px;  } |

### Resumen del enfoque de diseño

Básicamente, el enfoque que tomé aquí fue utilizar CSS Grid Layout para el diseño general (y cualquier cosa que no fuera lineal en el diseño). Dentro de las áreas de contenido de la cuadrícula, se utilizó Flexbox para ordenar y ajustar el estilo dentro de las áreas de la cuadrícula.

### Recursos

Hay tantos recursos excelentes para Flexbox y CSS Grid Layouts, demasiados para mencionarlos. Aquí hay algunos que lo ayudarán a comenzar en la dirección correcta e inspirarán sus diseños.

* [Hoja de referencia de alineación de cajas](https://rachelandrew.co.uk/css/cheatsheets/box-alignment)
* [Laboratorio de diseños de Jen Simmons](http://labs.jensimmons.com/)
* [Una guía completa de Grid](https://css-tricks.com/snippets/css/complete-guide-grid/)
* [Una guía completa de Flexbox](https://css-tricks.com/snippets/css/a-guide-to-flexbox/)

Con suerte, estos ejercicios le dieron una mejor idea de cómo crear diseños con CSS Grid Layouts y Flexbox. ¿Qué opinas de estas nuevas tecnologías? ¿Cómo han ayudado a su proceso de desarrollo?