

INICIO / GUÍAS /

# Una guía completa de Flexbox



Chris Coyier en 8 de abril de 2013 (Actualizado en 9 de diciembre de 2022)

Nuestra guía completa para el diseño de CSS flexbox. Esta guía completa explica todo sobre flexbox, centrándose en todas las diferentes propiedades posibles para el elemento principal (el contenedor flexible) y los elementos secundarios (los elementos flexibles). También incluye historial, demostraciones, patrones y un gráfico de soporte del navegador.

## Tabla de contenido

- Part 1: ([#aa-introduction](#)) [Fondo](#) ([#aa-background](#))
- Part 2: ([#aa-introduction](#)) [Conceptos básicos y terminología](#) ([#aa-basics-and-terminology](#))
- Part 3: ([#aa-introduction](#)) [Propiedades de caja flexible](#) ([#aa-flexbox-properties](#))
- Part 4: ([#aa-introduction](#)) [Prefijo Flexbox](#) ([#aa-prefixing-flexbox](#))
- Part 5: ([#aa-introduction](#)) [Ejemplos](#) ([#aa-examples](#))
- Part 6: ([#aa-introduction](#)) [trucos de flexbox](#) ([#aa-flexbox-tricks](#))
- Part 7: ([#aa-introduction](#)) [Compatibilidad con navegador](#) ([#aa-browser-support](#))
- Part 8: ([#aa-introduction](#)) [Insectos](#) ([#aa-bugs](#))
- Part 9: ([#aa-introduction](#)) [Propiedades relacionadas](#) ([#aa-related-properties](#))
- Part 10: ([#aa-introduction](#)) [Más información](#) ([#aa-more-information](#))
- Part 11: [Más fuentes](#) ([#aa-more-sources](#))

[\(#aa-get-the-poster\)](#) ¡Consigue el cartel!



¿Referencia mucho esta guía? ¡Aquí hay una imagen de alta resolución que puede imprimir!

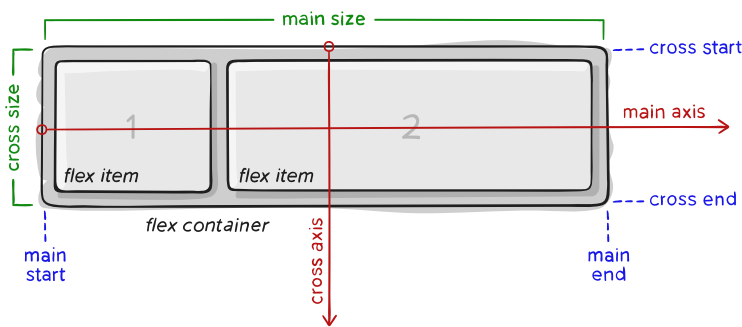
DESCARGAR GRATIS ([HTTPS://CSS-TRICKS.COM/WP-CONTENT/UPLOADS/2022/02/CSS-FLEXBOX-POSTER.PNG](https://css-tricks.com/wp-content/uploads/2022/02/css-flexbox-poster.png))

## 🔗 (#aa-background) Fondo

## 🔗 (#aa-basics-and-terminology) Conceptos básicos y terminología

Dado que flexbox es un módulo completo y no una sola propiedad, implica muchas cosas, incluido su conjunto completo de propiedades. Algunos de ellos están destinados a establecerse en el contenedor (elemento principal, conocido como "contenedor flexible"), mientras que otros están destinados a establecerse en los elementos secundarios (dichos "elementos flexibles").

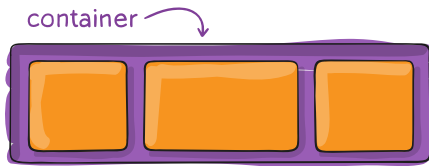
Si el diseño "regular" se basa en direcciones de flujo en bloque y en línea, el diseño flexible se basa en "direcciones de flujo flexible". Eche un vistazo a esta figura de la especificación, que explica la idea principal detrás del diseño flexible.



Los elementos se distribuirán siguiendo el eje (main axis de main-start a main-end) o la cruz (de cross-start a cross-end).

- **eje principal** : el eje principal de un contenedor flexible es el eje principal a lo largo del cual se disponen los elementos flexibles. Cuidado, no es necesariamente horizontal; Depende de la flex-direction propiedad (ver más abajo).
- **inicio principal | main-end** : los elementos flexibles se colocan dentro del contenedor comenzando desde el inicio principal y yendo al extremo principal.
- **tamaño principal** : el ancho o la altura de un elemento flexible, lo que esté en la dimensión principal, es el tamaño principal del elemento. La propiedad de tamaño principal del elemento flexible es la propiedad 'ancho' o 'alto', cualquiera que esté en la dimensión principal.
- **eje transversal** : el eje perpendicular al eje principal se denomina eje transversal. Su dirección depende de la dirección del eje principal.
- **inicio cruzado | extremo cruzado** : las líneas flexibles se llenan con artículos y se colocan en el contenedor comenzando en el lado de inicio cruzado del contenedor flexible y yendo hacia el lado del extremo cruzado.
- **tamaño cruzado** : el ancho o la altura de un elemento flexible, lo que esté en la dimensión cruzada, es el tamaño cruzado del elemento. La propiedad de tamaño cruzado es cualquiera de 'ancho' o 'alto' que esté en la dimensión cruzada.

## 🔗 (#aa-flexbox-properties) Propiedades de caja flexible



## (#aa-properties-for-the-parentflex-container) Propiedades para el padre (contenedor flexible)

### (#aa-display) mostrar

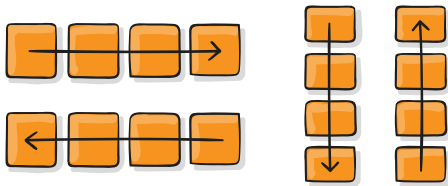
Esto define un contenedor flexible; en línea o bloque dependiendo del valor dado. Permite un contexto flexible para todos sus hijos directos.

```
.container {
  display: flex; /* or inline-flex */
}
```

CSS

Tenga en cuenta que las columnas CSS no tienen efecto en un contenedor flexible.

### (#aa-flex-direction) dirección de flexión



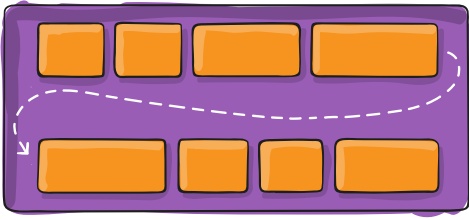
Esto establece el eje principal, definiendo así la dirección en que se colocan los artículos flexibles en el contenedor flexible. Flexbox es (aparte del envoltorio opcional) un concepto de diseño de una sola dirección. Piense en los elementos flexibles como dispuestos principalmente en filas horizontales o columnas verticales.

```
.container {
  flex-direction: row | row-reverse | column | column-reverse;
}
```

CSS

- row(predeterminado): de izquierda a derecha en ltr; de derecha a izquierda enrtl
- row-reverse: de derecha a izquierda en ltr; de izquierda a derecha enrtl
- column: igual que rowpero de arriba a abajo
- column-reverse: igual que row-reversepero de abajo hacia arriba

### (#aa-flex-wrap) envoltura flexible



De forma predeterminada, todos los elementos flexibles intentarán encajar en una línea. Puede cambiar eso y permitir que los elementos se ajusten según sea necesario con esta propiedad.

```
.container {  
  flex-wrap: nowrap | wrap | wrap-reverse;  
}
```

CSS

- **nowrap**(predeterminado): todos los elementos flexibles estarán en una línea
- **wrap**: los elementos flexibles se ajustarán a varias líneas, de arriba a abajo.
- **wrap-reverse**: los elementos flexibles se ajustarán a varias líneas de abajo hacia arriba.

Hay algunas [demostraciones visuales de flex-wrap](https://css-tricks.com/almanac/properties/f/flex-wrap/) (<https://css-tricks.com/almanac/properties/f/flex-wrap/>).

### [\(#aa-flex-flow\)](#) **flujo flexible**

Esta es una forma abreviada de las propiedades `flex-direction` y `flex-wrap`, que juntas definen los ejes principal y transversal del contenedor flexible. El valor predeterminado es `row nowrap`.

```
.container {  
  flex-flow: column wrap;  
}
```

CSS

### [\(#aa-justify-content\)](#) **justificar-contenido**

**flex-start****flex-end****center****space-between****space-around****space-evenly**

Esto define la alineación a lo largo del eje principal. Ayuda a distribuir el espacio libre adicional sobrante cuando todos los elementos flexibles de una línea son inflexibles o son flexibles pero han alcanzado su tamaño máximo. También ejerce cierto control sobre la alineación de los elementos cuando desbordan la línea.

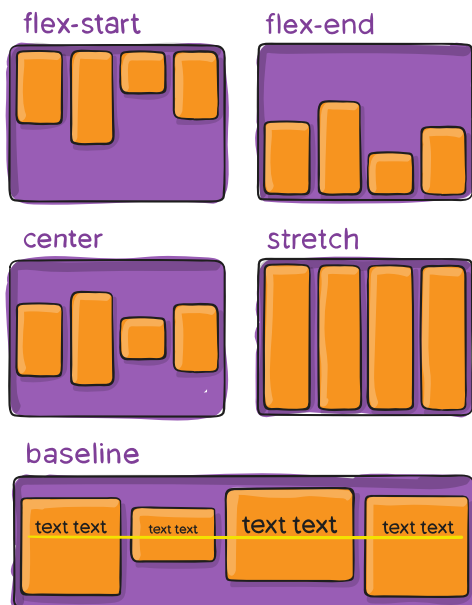
```
.container {
  justify-content: flex-start | flex-end | center | space-between | space-around | space-evenly | start | end | left | right ... + safe | unsafe;
}
```

- **flex-start**(predeterminado): los artículos se empaquetan hacia el inicio de la dirección flexible.
- **flex-end**: los artículos se embalan hacia el final de la dirección de flexión.
- **start**: los artículos se empaquetan hacia el inicio de la *writing-mode* dirección.
- **end**: los artículos se embalan hacia el final de la *writing-mode* dirección.
- **left**: los artículos se empaquetan hacia el borde izquierdo del contenedor, a menos que eso no tenga sentido con *flex-direction*, entonces se comporta como **start**.
- **right**: los artículos se empaquetan hacia el borde derecho del contenedor, a menos que eso no tenga sentido con *flex-direction*, entonces se comporta como **end**.
- **center**: los elementos se centran a lo largo de la línea
- **space-between**: los artículos se distribuyen uniformemente en la línea; el primer elemento está en la línea de inicio, el último elemento en la línea final
- **space-around**: los artículos se distribuyen uniformemente en la línea con el mismo espacio alrededor de ellos. Tenga en cuenta que visualmente los espacios no son iguales, ya que todos los elementos tienen el mismo espacio en ambos lados. El primer elemento tendrá una unidad de espacio contra el borde del contenedor, pero dos unidades de espacio entre el siguiente elemento porque ese siguiente elemento tiene su propio espacio que se aplica.
- **space-evenly**: los elementos se distribuyen de modo que el espacio entre dos elementos cualesquiera (y el espacio hasta los bordes) sea igual.

Tenga en cuenta que el soporte del navegador para estos valores está matizado. Por ejemplo, `space-between` nunca obtuve soporte de algunas versiones de Edge, e `inicio/fin/izquierda/derecha` aún no están en Chrome. MDN [tiene gráficos detallados](https://developer.mozilla.org/en-US/docs/Web/CSS/justify-content) (<https://developer.mozilla.org/en-US/docs/Web/CSS/justify-content>). Los valores más seguros son `flex-start`, `flex-end` y `center`.

También hay dos palabras clave adicionales que puede combinar con estos valores: `safety` y `unsafe`. El uso de `safety` garantiza que, independientemente de cómo haga este tipo de posicionamiento, no puede empujar un elemento de manera que se represente fuera de la pantalla (por ejemplo, fuera de la parte superior) de tal manera que el contenido no se pueda desplazar también (llamado "pérdida de datos").

## 🔗 (#aa-align-items) alinear elementos



Esto define el comportamiento predeterminado de cómo se distribuyen los elementos flexibles a lo largo del **eje transversal** en la línea actual. Piense en ello como la `justify-content` versión para el eje transversal (perpendicular al eje principal).

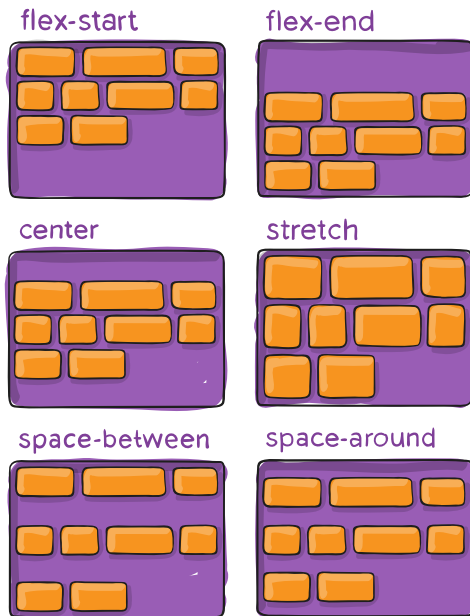
```
.container {
  align-items: stretch | flex-start | flex-end | center | baseline | first baseline | last baseline | start | end | self-start | self-end + ... sa
}
```

- **stretch**(predeterminado): estirar para llenar el contenedor (aún respeta el ancho mínimo/ancho máximo)
- **flex-start**// **start**: **self-start** los elementos se colocan al principio del eje transversal. La diferencia entre estos es sutil, y se trata de respetar las `flex-direction` reglas o las `writing-mode` reglas.
- **flex-end**// **end**: **self-end** los elementos se colocan al final del eje transversal. La diferencia nuevamente es sutil y se trata de respetar `flex-direction` reglas contra `writing-mode` reglas.
- **center**: los elementos están centrados en el eje transversal
- **baseline**: los elementos están alineados como sus líneas base se alinean

Las palabras clave modificadoras `safety` y `unsafe` se pueden usar junto con el resto de estas palabras clave (aunque tenga en cuenta la [compatibilidad del navegador](#)

(<https://developer.mozilla.org/en-US/docs/Web/CSS/align-items>), y tratan de ayudarlo a evitar la alineación de elementos de modo que el contenido se vuelva inaccesible.

## 🔗 (#aa-align-content) alinear-contenido



Esto alinea las líneas de un contenedor flexible cuando hay espacio adicional en el eje transversal, de forma similar a como justify-content se alinean los elementos individuales dentro del eje principal.

Hey!

**Nota:** Esta propiedad solo tiene efecto en contenedores flexibles de varias líneas, donde flex-wrap se establece en wrap o wrap-reverse. Un contenedor flexible de una sola línea (es decir, donde flex-wrap se establece en su valor predeterminado, no-wrap) no reflejará align-content.

```
.container {
  align-content: flex-start | flex-end | center | space-between | space-around | space-evenly | stretch | start | end | baseline | first baseline
}
```

CSS

- normal(predeterminado): los artículos se empaquetan en su posición predeterminada como si no se hubiera establecido ningún valor.
- flex-start/ start: artículos embalados al inicio del contenedor. El (más compatible) flex-start honra el flex-direction mientras que start honra la writing-mode dirección.
- flex-end/ end: artículos embalados hasta el final del contenedor. El (más soporte) flex-end honra el flex-direction mientras que el final honra la writing-mode dirección.
- center: elementos centrados en el contenedor
- space-between: artículos distribuidos uniformemente; la primera línea está al principio del contenedor mientras que la última está al final
- space-around: artículos distribuidos uniformemente con el mismo espacio alrededor de cada línea
- space-evenly: los elementos se distribuyen uniformemente con el mismo espacio a su alrededor
- stretch: las líneas se estiran para ocupar el espacio restante

Las palabras clave modificadoras safe y unsafe pueden usarse junto con el resto de estas palabras clave (aunque tenga en cuenta la [compatibilidad del navegador](#))

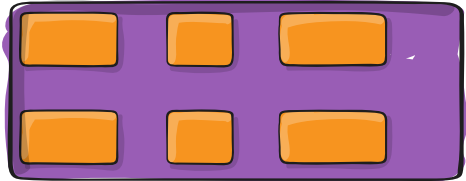
(<https://developer.mozilla.org/en-US/docs/Web/CSS/align-items>), y tratan de ayudarlo a evitar la alineación de elementos de modo que el contenido se vuelva inaccesible.

## 🔗 (#aa-gap-row-gap-column-gap) espacio, espacio entre filas, espacio entre columnas

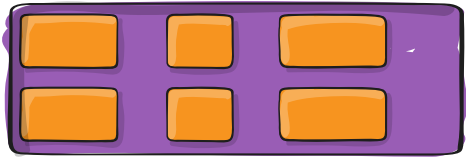
gap: 10px



gap: 30px



gap: 10px 30px



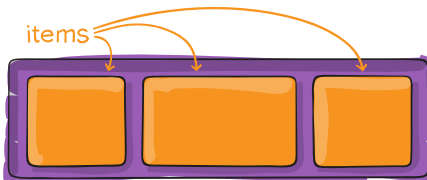
La gap propiedad (<https://css-tricks.com/almanac/properties/g/gap/>) controla explícitamente el espacio entre elementos flexibles. Se aplica ese espacio *solo entre elementos que* no están en los bordes exteriores.

```
.container {
  display: flex;
  ...
  gap: 10px;
  gap: 10px 20px; /* row-gap column gap */
  row-gap: 10px;
  column-gap: 20px;
}
```

CSS

El comportamiento podría considerarse como un canalón *mínimo*, como si el canalón fuera más grande de alguna manera (debido a algo como `justify-content: space-between;`), entonces la brecha solo tendrá efecto si ese espacio termina siendo más pequeño.

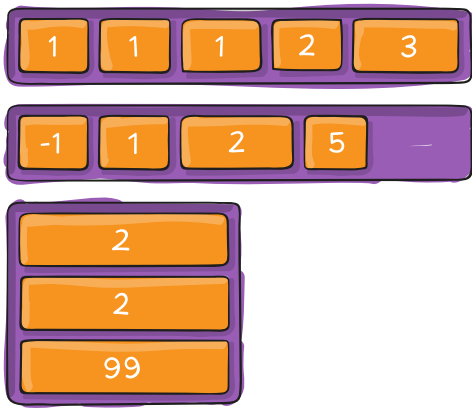
No es exclusivamente para flexbox, también gap funciona en cuadrícula y diseño de varias columnas.



## 🔗 (#aa-properties-for-the-childrenflex-items) Propiedades para los niños (artículos flexibles)

### 🔗 (#aa-order) ordenar





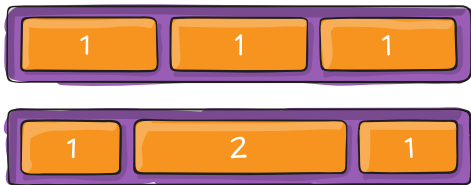
De forma predeterminada, los elementos flexibles se presentan en el orden de origen. Sin embargo, la `order` propiedad controla el orden en que aparecen en el contenedor flexible.

```
.item {
  order: 5; /* default is 0 */
}
```

CSS

Los artículos con el mismo `order` orden de origen vuelven.

### 🔗 (#aa-flex-grow) crecimiento flexible



Esto define la capacidad de crecimiento de un elemento flexible si es necesario. Acepta un valor sin unidades que sirve como proporción. Determina qué cantidad de espacio disponible dentro del contenedor flexible debe ocupar el artículo.

Si todos los elementos se han `flex-grow` establecido en 1, el espacio restante en el contenedor se distribuirá por igual a todos los niños. Si uno de los hijos tiene un valor de 2, ese hijo ocuparía el doble de espacio que cualquiera de los otros (o lo intentará, al menos).

```
.item {
  flex-grow: 4; /* default 0 */
}
```

CSS

Los números negativos no son válidos.

### 🔗 (#aa-flex-shrink) flexión-encogimiento

Esto define la capacidad de que un elemento flexible se encoja si es necesario.

```
.item {
  flex-shrink: 3; /* default 1 */
}
```

CSS

}

Los números negativos no son válidos.

### 🔗 (#aa-flex-basis) base flexible

Esto define el tamaño predeterminado de un elemento antes de que se distribuya el espacio restante. Puede ser una longitud (por ejemplo, 20 %, 5 rem, etc.) o una palabra clave. La autopalabra clave significa "mirar mi propiedad de ancho o alto" (lo cual fue hecho temporalmente por la `main-size` palabra clave hasta que quedó en desuso). La palabra clave significa "dimensionarlo en función del contenido del elemento". Esta palabra clave aún no es compatible, `content` por lo que es difícil probarla y saber qué hacen sus hermanos `.max-content`, `min-content`, `fit-content`.

```
.item {
  flex-basis: 100px; /* default auto */
}
```

CSS

Si se establece en `0`, el espacio adicional alrededor del contenido no se tiene en cuenta. Si se establece en `auto`, el espacio adicional se distribuye en función de su `flex-grow` valor. [Vea este gráfico.](http://www.w3.org/TR/css3-flexbox/images/rel-vs-abs-flex.svg) (<http://www.w3.org/TR/css3-flexbox/images/rel-vs-abs-flex.svg>)

### 🔗 (#aa-flex) doblar

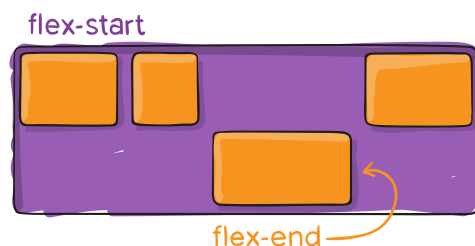
Esta es la abreviatura de `flex-grow`, `flex-shrink` y `flex-basis` combinado. Los parámetros segundo y tercero (`flex-shrink` y `flex-basis`) son opcionales. El valor predeterminado es `0 1 auto`, pero si lo configura con un solo valor numérico, como `flex: 5`, eso cambia `flex-basis` a `0%`, por lo que es como configurar `flex-grow: 5`; `flex-shrink: 1`; `flex-basis: 0%`.

```
.item {
  flex: none | [ <'flex-grow'> <'flex-shrink'>? || <'flex-basis'> ]
}
```

CSS

**Se recomienda utilizar esta propiedad abreviada en** lugar de establecer las propiedades individuales. La taquigrafía establece los otros valores de forma inteligente.

### 🔗 (#aa-align-self) alinearse



Esto permite anular la alineación predeterminada (o la especificada por `align-items`) para elementos flexibles individuales.

Consulte la [align-items](#) explicación para comprender los valores disponibles.

```
.item {
  align-self: auto | flex-start | flex-end | center | baseline | stretch;
}
```

CSS

Tenga en cuenta que float, clear y vertical-align no tienen efecto en un elemento flexible.

## ▼ #aa-prefixing-flexbox) Prefijo Flexbox

Flexbox requiere algunos prefijos de proveedores para admitir la mayoría de los navegadores posibles. No solo incluye propiedades antepuestas con el prefijo del proveedor, sino que en realidad hay nombres de propiedades y valores completamente diferentes. Esto se debe a que la especificación Flexbox ha cambiado con el tiempo, creando versiones "antiguas", "interpoladas" y "nuevas" (<https://css-tricks.com/old-flexbox-and-new-flexbox/>).

Quizás la mejor manera de manejar esto es escribir en la sintaxis nueva (y final) y ejecutar su CSS a través de Autoprefixer (<https://css-tricks.com/autoprefixer/>), que maneja muy bien las fallas.

Alternativamente, aquí hay un Sass `@mixin` para ayudar con algunos de los prefijos, que también le da una idea de qué tipo de cosas se deben hacer:

```
@mixin flexbox() {
  display: -webkit-box;
  display: -moz-box;
  display: -ms-flexbox;
  display: -webkit-flex;
  display: flex;
}

@mixin flex($values) {
  -webkit-box-flex: $values;
  -moz-box-flex: $values;
  -webkit-flex: $values;
  -ms-flex: $values;
  flex: $values;
}

@mixin order($val) {
  -webkit-box-ordinal-group: $val;
  -moz-box-ordinal-group: $val;
  -ms-flex-order: $val;
  -webkit-order: $val;
  order: $val;
}

.wrapper {
  @include flexbox();
}

.item {
  @include flex(1 200px);
  @include order(2);
}
```

SCSS

## ▼ #aa-examples) Ejemplos

Empecemos con un ejemplo muy muy sencillo, resolviendo un problema casi diario: el centrado perfecto. No podría ser más simple si usas flexbox.

```
.parent {
  display: flex;
  height: 300px; /* Or whatever */
}

.child {
  width: 100px; /* Or whatever */
  height: 100px; /* Or whatever */
  margin: auto; /* Magic! */
}
```

Esto se basa en el hecho de que un margen establecido auto en un contenedor flexible absorbe espacio adicional. Por lo tanto, establecer un margen de auto hará que el elemento quede perfectamente centrado en ambos ejes.

Ahora usemos algunas propiedades más. Considere una lista de 6 elementos, todos con dimensiones fijas, pero que pueden ajustarse automáticamente. Queremos que se distribuyan uniformemente en el eje horizontal para que cuando cambiemos el tamaño del navegador, todo se escale bien y sin consultas de medios.

```
.flex-container {
  /* We first create a flex layout context */
  display: flex;

  /* Then we define the flow direction
   and if we allow the items to wrap
   * Remember this is the same as:
   * flex-direction: row;
   * flex-wrap: wrap;
   */
  flex-flow: row wrap;

  /* Then we define how is distributed the remaining space */
  justify-content: space-around;
}
```

Hecho. Todo lo demás es solo una preocupación de estilo. A continuación se muestra un bolígrafo con este ejemplo. Asegúrese de ir a CodePen e intente cambiar el tamaño de sus ventanas para ver qué sucede.

Embedded Pen Here

Probemos otra cosa. Imagine que tenemos un elemento de navegación alineado a la derecha en la parte superior de nuestro sitio web, pero queremos que esté centrado en pantallas medianas y de una sola columna en dispositivos pequeños. Suficientemente fácil.

```
/* Large */
.navigation {
  display: flex;
  flex-flow: row wrap;
  /* This aligns items to the end line on main-axis */
  justify-content: flex-end;
}

/* Medium screens */
@media all and (max-width: 800px) {
  .navigation {
    /* When on medium sized screens, we center it by evenly distributing empty space around items */
    justify-content: space-around;
  }
}
```

```
}

/* Small screens */
@media all and (max-width: 500px) {
  .navigation {
    /* On small screens, we are no longer using row direction but column */
    flex-direction: column;
  }
}
```

Embedded Pen Here

¡Intentemos algo aún mejor jugando con la flexibilidad de los elementos flexibles! ¿Qué pasa con un diseño de 3 columnas para dispositivos móviles con encabezado y pie de página de ancho completo? E independiente del orden de origen.

```
.wrapper {
  display: flex;
  flex-flow: row wrap;
}

/* We tell all items to be 100% width, via flex-basis */
.wrapper > * {
  flex: 1 100%;
}

/* We rely on source order for mobile-first approach
 * in this case:
 * 1. header
 * 2. article
 * 3. aside 1
 * 4. aside 2
 * 5. footer
 */

/* Medium screens */
@media all and (min-width: 600px) {
  /* We tell both sidebars to share a row */
  .aside { flex: 1 auto; }
}

/* Large screens */
@media all and (min-width: 800px) {
  /* We invert order of first sidebar and main
   * And tell the main element to take twice as much width as the other two sidebars
   */
  .main { flex: 3 0px; }
  .aside-1 { order: 1; }
  .main { order: 2; }
  .aside-2 { order: 3; }
  .footer { order: 4; }
}
```

CSS

Embedded Pen Here

## ▼#aa-browser-support) Compatibilidad con navegador

Los datos de soporte de este navegador son de [Caniuse](http://caniuse.com/#feat=flexbox) (<http://caniuse.com/#feat=flexbox>), que tiene más detalles. Un número indica que el navegador admite la función en esa versión y en adelante.

### Escritorio

Chrome: 21\* Firefox: 28 IE: 11 Edge: 12 Safari: 6.1\*

### Móvil / Tableta

Android Chrome: 108 Android Firefox: 107 Android: 4.4 iOS Safari: 7.0-7.1\*

## ▼#aa-bugs) Insectos

Flexbox ciertamente no está exento de errores. La mejor colección de ellos que he visto es [Flexbugs](https://github.com/philipwalton/flexbugs) (<https://github.com/philipwalton/flexbugs>) de Philip Walton y Greg Whitworth . Es un lugar de código abierto para realizar un seguimiento de todos ellos, por lo que creo que es mejor vincular a eso.

## ▼#aa-related-properties) Propiedades relacionadas

Almanaque del 28 de julio de 2021

**alinear contenido** (<https://css-tricks.com/almanac/properties/a/align-content/>)

```
.element { align-content: space-around; } (https://css-tricks.com/almanac/properties/a/align-content/)
```



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/3  
4cr  
oss  
/)

34 Cruz (<https://css-tricks.com/author/34cross/>)

Almanaque el 28 de septiembre de 2022

**alinear elementos** (<https://css-tricks.com/almanac/properties/a/align-items/>)

```
.element { align-items: flex-start; } (https://css-tricks.com/almanac/properties/a/align-items/)
```



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/3  
4cr  
oss  
/)

34 Cruz (<https://css-tricks.com/author/34cross/>)

Almanaque del 27 de diciembre de 2018


**alinearse** (<https://css-tricks.com/almanac/properties/a/align-self/>)

```
.box { align-self: flex-end; } (https://css-tricks.com/almanac/properties/a/align-self/)
```

Almanaque del 30 de agosto de 2021

**espacio entre columnas** (<https://css-tricks.com/almanac/properties/c/column-gap/>)

```
.example { column-gap: 25px; } (https://css-tricks.com/almanac/properties/c/column-gap/)
```



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/3  
4cr  
oss  
/)

34 Cruz (<https://css-tricks.com/author/34cross/>)




(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/g  
eof  
fg  
rah  
am  
/)

geoff graham (<https://css-tricks.com/author/geoffgraham/>)

Almanaque del 15 de octubre de 2021

**mostrar (<https://css-tricks.com/almanac/properties/d/display/>)**

`.element { display: inline-block; } (https://css-tricks.com/almanac/properties/d/display/)`




(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/s  
ara  
co  
pe/  
)

Sara Cope (<https://css-tricks.com/author/saracope/>)

Almanaque el 22 de septiembre de 2022

**brecha (<https://css-tricks.com/almanac/properties/g/gap/>)**

`.element { gap: 20px 30px; } (https://css-tricks.com/almanac/properties/g/gap/)`




(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/s  
eye  
di/  
)

Mojtaba Seyedi (<https://css-tricks.com/author/seyedi/>)

Almanaque del 2 de marzo de 2021

**justificar-elementos (<https://css-tricks.com/almanac/properties/j/justify-items/>)**

`.element { justify-items: center; } (https://css-tricks.com/almanac/properties/j/justify-items/)`




(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/  
/)

mohit khare (<https://css-tricks.com/author/mohitkhare/>)

Almanaque del 30 de septiembre de 2022

**doblar (<https://css-tricks.com/almanac/properties/f/flex/>)**

`.element { flex: 1 1 100px; } (https://css-tricks.com/almanac/properties/f/flex/)`



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/s  
/)

Sara Cope (<https://css-tricks.com/author/saracope/>)

mo  
hit  
kh  
are  
)

ara  
co  
pe/  
)

Almanaque del 30 de septiembre de 2022

### base flexible (<https://css-tricks.com/almanac/properties/f/flex-basis/>)

```
.element { flex-basis: 100px; } (https://css-tricks.com/almanac/properties/f/flex-basis/)
```



(ht  
tps  
://c  
ss-  
tri  
cks  
.co 34 Cruz (<https://css-tricks.com/author/34cross/>)  
m/  
aut  
ho  
r/3  
4cr  
oss  
)

Almanaque del 4 de agosto de 2021

### dirección de flexión (<https://css-tricks.com/almanac/properties/f/flex-direction/>)

```
.element { flex-direction: column-reverse; } (https://css-tricks.com/almanac/properties/f/flex-direction/)
```



(ht  
tps  
://c  
ss-  
tri  
cks (<https://css-tricks.com/author/>)  
.co  
m/  
aut  
ho  
r/)

Almanaque del 4 de agosto de 2021

### flujo flexible (<https://css-tricks.com/almanac/properties/f/flex-flow/>)

```
.element { flex-flow: row wrap; } (https://css-tricks.com/almanac/properties/f/flex-flow/)
```



(ht  
tps  
://c  
ss-  
tri  
cks (<https://css-tricks.com/author/>)  
.co  
m/  
aut  
ho  
r/)

Almanaque del 4 de agosto de 2021

### crecimiento flexible (<https://css-tricks.com/almanac/properties/f/flex-grow/>)

```
.flex-item { flex-grow: 2; } (https://css-tricks.com/almanac/properties/f/flex-grow/)
```



(ht  
tps  
://c  
ss-  
tri  
cks  
.co chris coyier (<https://css-tricks.com/author/chriscoyier/>)  
m/  
aut  
ho  
r/c  
hri  
sco  
yie  
r/)

Almanaque del 4 de agosto de 2021

### flexión-encogimiento (<https://css-tricks.com/almanac/properties/f/flex-shrink/>)

```
.element { flex-shrink: 2; } (https://css-tricks.com/almanac/properties/f/flex-shrink/)
```

Almanaque del 30 de septiembre de 2022

### envoltura flexible (<https://css-tricks.com/almanac/properties/f/flex-wrap/>)

```
.example { flex-wrap: wrap; } (https://css-tricks.com/almanac/properties/f/flex-wrap/)
```





(ht  
tps  
://c  
ss-  
tri  
cks  
(<https://css-tricks.com/author/>)  
.co  
m/  
aut  
ho  
r/)



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/c  
hri  
sco  
yie  
r/)  
chris coyier (<https://css-tricks.com/author/chriscoyier/>)

Almanaque el 22 de septiembre de 2022

**justificar-contenido** (<https://css-tricks.com/almanac/properties/j/justify-content/>)

```
.element { justify-content: center; } (https://css-tricks.com/almanac/properties/j/justify-content/)
```



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/3  
4cr  
oss  
/)  
34 Cruz (<https://css-tricks.com/author/34cross/>)



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/g  
eof  
fg  
rah  
am  
/)  
geoff graham (<https://css-tricks.com/author/geoffgraham/>)

Almanaque del 30 de agosto de 2021

**espacio entre filas** (<https://css-tricks.com/almanac/properties/r/row-gap/>)

```
.element { row-gap: 2rem; } (https://css-tricks.com/almanac/properties/r/row-gap/)
```



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/g  
eof  
eof  
geoff graham (<https://css-tricks.com/author/geoffgraham/>)

fg  
rah  
am  
/)

[#aa-more-information](#) **Más información**

- **Módulo de diseño de caja flexible CSS Nivel 1** (<https://www.w3.org/TR/css-flexbox-1/>) (W3C)
- **Una hoja de trucos de CSS Flexbox** ([https://www.digitalocean.com/community/cheatsheets/css-flexbox?utm\\_medium=content\\_acq&utm\\_source=css-tricks&utm\\_campaign=&utm\\_content=awareness\\_best sellers](https://www.digitalocean.com/community/cheatsheets/css-flexbox?utm_medium=content_acq&utm_source=css-tricks&utm_campaign=&utm_content=awareness_best sellers)) (DigitalOcean)
- **Centrando cosas en CSS con Flexbox** ([https://www.digitalocean.com/community/tutorials/css-centering-using-flexbox?utm\\_medium=content\\_acq&utm\\_source=css-tricks&utm\\_campaign=&utm\\_content=awareness\\_best sellers](https://www.digitalocean.com/community/tutorials/css-centering-using-flexbox?utm_medium=content_acq&utm_source=css-tricks&utm_campaign=&utm_content=awareness_best sellers)) (DigitalOcean)

Artículo del 26 de septiembre de 2013

**Resuelto por flexbox** (<https://css-tricks.com/solved-flexbox/>)



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/c  
hri  
sco  
yie  
r/)

chris coyier (<https://css-tricks.com/author/chriscoyier/>)

Artículo del 25 de noviembre de 2013

**Hoja de referencia de Flexbox** (<https://css-tricks.com/flexbox-cheat-sheet/>)



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/c  
hri  
sco  
yie  
r/)

chris coyier (<https://css-tricks.com/author/chriscoyier/>)

Artículo del 23 de diciembre de 2012

**Sumérgete en Flexbox** (<https://css-tricks.com/dive-into-flexbox/>)



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/c  
hri  
sco  
yie  
r/)

chris coyier (<https://css-tricks.com/author/chriscoyier/>)

Artículo del 23 de octubre de 2018

**Casos de uso para Flexbox** (<https://css-tricks.com/use-cases-for-flexbox/>)



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/c  
hri  
sco  
yie  
r/)

chris coyier (<https://css-tricks.com/author/chriscoyier/>)

Artículo del 14 de febrero de 2019

**¡Rápido! ¿Cuál es la diferencia entre Flexbox y Grid? (<https://css-tricks.com/quick-whats-the-difference-between-flexbox-and-grid/>)**

(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/c  
hri  
sco  
yie  
r/)

**chris coyier** (<https://css-tricks.com/author/chriscoyier/>)

Artículo del 23 de febrero de 2022

**¿CSS Grid reemplaza a Flexbox? (<https://css-tricks.com/css-grid-replace-flexbox/>)**

(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/r  
obi  
nre  
ndl  
e/)

**petirrojo** (<https://css-tricks.com/author/robinrendle/>)

Artículo del 25 de junio de 2020

**Grid para diseño, flexbox para componentes (<https://css-tricks.com/grid-for-layout-flexbox-for-components/>)**

(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/r  
obi  
nre  
ndl  
e/)

**petirrojo** (<https://css-tricks.com/author/robinrendle/>)

Artículo del 13 de abril de 2016

**¿Debo usar Grid o Flexbox? (<https://css-tricks.com/use-grid-flexbox/>)**

(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/r  
obi  
nre  
ndl  
e/)

**petirrojo** (<https://css-tricks.com/author/robinrendle/>)

Artículo del 13 de agosto de 2016

**No lo piense demasiado (Flexbox) Rejillas (<https://css-tricks.com/dont-overthink-flexbox-grids/>)**

(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/c  
hri  
sco  
yie  
r/)

**chris coyier** (<https://css-tricks.com/author/chriscoyier/>)

Artículo del 24 de noviembre de 2021

**Creación de diseños multidireccionales (<https://css-tricks.com/building-multi-directional-layouts/>)**

(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/a  
hmad  
alfy/)

**Ahmad El Alfy** (<https://css-tricks.com/author/ahmadalfy/>)

yie  
r/)

alf  
y/)

Artículo del 6 de enero de 2020  
**Cómo funcionan los márgenes automáticos en Flexbox**  
(<https://css-tricks.com/how-auto-margins-work-in-flexbox/>)



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/c  
hri  
sco  
yie  
r/)

chris coyier (<https://css-tricks.com/author/chriscoyier/>)

Artículo del 10 de abril de 2017  
**`flex-grow` es raro. ¿O es eso?** (<https://css-tricks.com/flex-grow-is-weird/>)



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/  
m  
ma  
tuz  
o/)

manuel matuzovic (<https://css-tricks.com/author/mmatuzo/>)

Artículo del 18 de octubre de 2022  
**Comprender flex-grow, flex-shrink y flex-base**  
(<https://css-tricks.com/understanding-flex-grow-flex-shrink-and-flex-basis/>)



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/r  
obi  
nre  
ndl  
e/)

petirrojo (<https://css-tricks.com/author/robinrendle/>)

Artículo del 18 de febrero de 2019  
**Colocación automática de cuadrícula compatible con IE10 con Flexbox** (<https://css-tricks.com/ie10-compatible-grid-auto-placement-with-flexbox/>)



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut  
ho  
r/b  
hol  
t/)

Brian Holt (<https://css-tricks.com/author/bholt/>)

Artículo del 13 de agosto de 2013  
**Flexbox “antiguo” y Flexbox “nuevo”** (<https://css-tricks.com/old-flexbox-and-new-flexbox/>)



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut

chris coyier (<https://css-tricks.com/author/chriscoyier/>)

Artículo del 15 de junio de 2013  
**Uso de Flexbox: mezcla de lo antiguo y lo nuevo para obtener la mejor compatibilidad con navegadores**  
(<https://css-tricks.com/using-flexbox/>)



(ht  
tps  
://c  
ss-  
tri  
cks  
.co  
m/  
aut

chris coyier (<https://css-tricks.com/author/chriscoyier/>)

ho  
r/c  
hri  
sco  
yie  
r/)

ho  
r/c  
hri  
sco  
yie  
r/)

## ▼#aa-more-sources) Más fuentes

- Módulo de diseño de caja flexible CSS Nivel 1 (<https://www.w3.org/TR/css-flexbox-1/>) (W3C)
- Una hoja de trucos de CSS Flexbox ([https://www.digitalocean.com/community/cheatsheets/css-flexbox?utm\\_medium=content\\_acq&utm\\_source=css-tricks&utm\\_campaign=&utm\\_content=awareness\\_best sellers](https://www.digitalocean.com/community/cheatsheets/css-flexbox?utm_medium=content_acq&utm_source=css-tricks&utm_campaign=&utm_content=awareness_best sellers)) (DigitalOcean)
- Centrando cosas en CSS con Flexbox ([https://www.digitalocean.com/community/tutorials/css-centering-using-flexbox?utm\\_medium=content\\_acq&utm\\_source=css-tricks&utm\\_campaign=&utm\\_content=awareness\\_best sellers](https://www.digitalocean.com/community/tutorials/css-centering-using-flexbox?utm_medium=content_acq&utm_source=css-tricks&utm_campaign=&utm_content=awareness_best sellers)) (DigitalOcean)
- Diseño CSS: Flexbox ([https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS\\_layout/Flexbox](https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Flexbox)) (MDN)
- Casos de uso para Flexbox (<https://www.smashingmagazine.com/2018/10/flexbox-use-cases/>) (Revista Smashing)