Cajas flexibles

Las cajas flexibles, o flexbox, es un nuevo modo de disposición en CSS3.

El uso de flexbox asegura que los elementos se comporten de manera predecible cuando el diseño de la página debe acomodar diferentes tamaños de pantalla y diferentes dispositivos de visualización.

Para muchas aplicaciones, el modelo de caja flexible proporciona una mejora sobre el modelo de bloque en el sentido de que no utiliza floats, ni tampoco los márgenes del contenedor flexible se colapsan con los márgenes de su contenido.

Flexbox se compone de contenedores y elementos flexibles.

Se declara un contenedor flexible estableciendo la propiedad de visualización de un elemento a tipo flex (renderizándose como un bloque) o a liníne-flex (representándose como elemento en linea).

Dentro de un contenedor flexible hay uno o más elementos flexibles.

Nota: todo lo que se encuentre fuera de un contenedor flexible y dentro de un elemento flexible se procesa como de costumbre. Flexbox define cómo se colocan los elementos flexibles dentro de un contenedor flexible.

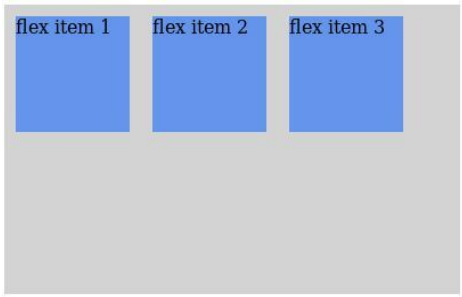
Los elementos flexibles se colocan dentro de un contenedor flexible a lo largo de una línea flexible. Por defecto, sólo hay una línea flexible por contenedor flexible.

Display:flex

El siguiente ejemplo muestra tres elementos flexibles. Están posicionados de forma predeterminada: a lo largo de la línea horizontal de flexión, de izquierda a derecha:

```
.flex-container {
    display: -webkit-flex;
    display: flex;
    width: 400px;
    height: 250px;
    background-color: lightgrey;
}



.flex-item {
    background-color: cornflowerblue;
    width: 100px;
    height: 100px;
    margin: 10px;
}
```

```
<div class="flex-container">
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
```



flex-direction

También es posible cambiar la dirección de la línea de flexión.

Si ajustamos la propiedad de direction a rtl (de derecha a izquierda), el texto se dibuja de derecha a izquierda y también la línea de flexión cambia de dirección, lo que cambiará el diseño de página.
La propiedad de direction de flex especifica la dirección de los elementos flexibles dentro del contenedor de flexión. El valor predeterminado de la dirección de flexión es fila (de izquierda a derecha, de arriba a abajo).

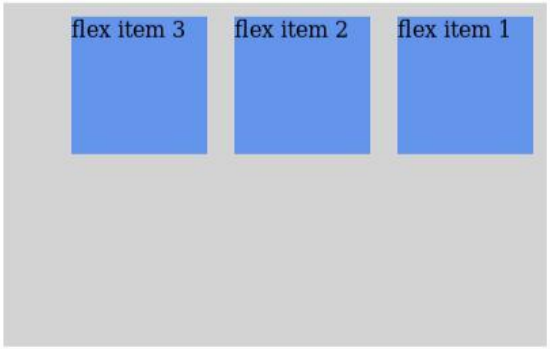
Los otros valores son los siguientes:

- row-reverse - Si el modo de escritura (dirección) se deja a la derecha, los elementos de flexión se colocarán de derecha a izquierda
- column - Si el sistema de escritura es horizontal, los elementos de flexión serán dispuestos verticalmente
- column-reverse - Igual que la columna, pero invertida

El siguiente ejemplo muestra el resultado del uso del valor de fila inversa:

```
.flex-container {
    display: -webkit-flex;
    display: flex;
    -webkit-flex-direction: row-
reverse;
    flex-direction: row-reverse;
    width: 400px;
    height: 250px;
    background-color: lightgrey;
}

.flex-item {
    background-color: cornflowerblue;
    width: 100px;
    height: 100px;
    margin: 10px;
}
```

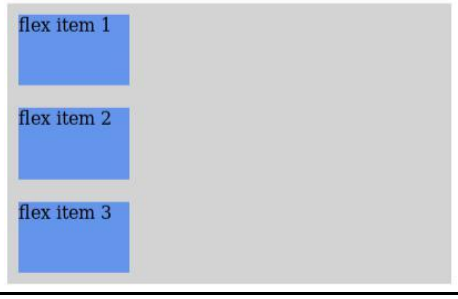
```
<div class="flex-container">
 <div class="flex-item">flex item
1</div>
  <div class="flex-item">flex
item2</div>
  <div class="flex-item">flex item
3</div>
</div>
```



El siguiente ejemplo muestra el resultado de usar el valor column:

```
.flex-container {
    display: -webkit-flex;
    display: flex;
    -webkit-flex-direction: column;
    flex-direction: column;
    width: 400px;
    height: 250px;
    background-color: lightgrey;
}

.flex-item {
    background-color: cornflowerblue;
    width: 100px;
    height: 100px;
    margin: 10px;
}
```

```
<div class="flex-container">
  <div class="flex-item">flex item
1</div>
  <div class="flex-item">flex item
2</div>
  <div class="flex-item">flex item

3</div>
</div>
```



Si usamos flex-direction: column-reverse; los elementos se dispondrían empezando por el tercero.

Justify-content

La propiedad Justify-content alinea horizontalmente los elementos del contenedor flexible cuando los elementos no utilizan todo el espacio disponible en el eje principal.

Los valores posibles son los siguientes:

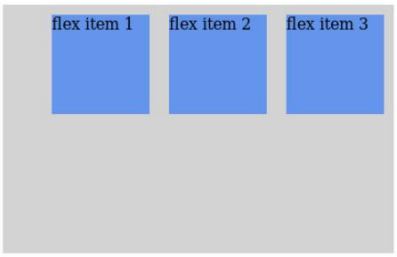
- Flex-start - Valor predeterminado. Los artículos se colocan al principio del contenedor
- Flex-end - Los artículos se colocan en el extremo del contenedor
- Center - Los objetos se colocan en el centro del contenedor
- Space-between - Los elementos se colocan con espacio entre las líneas
- Space-around - Los elementos se colocan con espacio antes, entre y después de las líneas

Ejemplos al utilizar los diferentes valores soobre estos elementos con estas reglas:

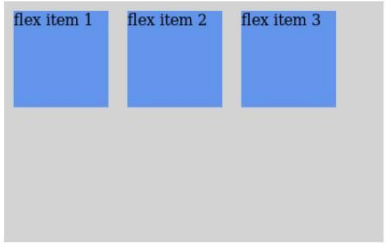
```
.flex-container {
    display: -webkit-flex;
    display: flex;
    width: 400px;
    height: 250px;
    background-color: lightgrey;
}
.flex-item {
    background-color: cornflowerblue;
    width: 100px;
    margin: 10px;
}
```

```
<div class="flex-container">
  <div class="flex-item">flex item 1</div>
  <div class="flex-item">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
```

justify-content: flex-end;

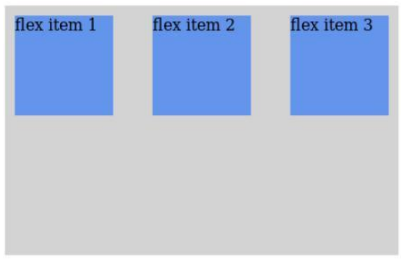


justify-content: flex-start;



justify-content: center;

justify-content: space-between;



justify-content: space-around;

align-items

La propiedad align-items alinea verticalmente los elementos del contenedor flexible cuando los elementos no utilizan todo el espacio disponible en el eje transversal.

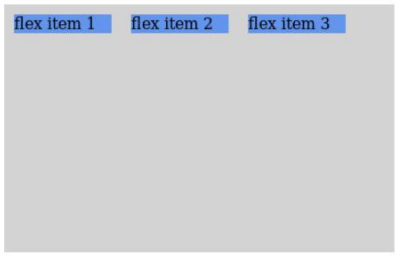
Los valores posibles son los siguientes:

- stretch - Valor predeterminado. Los artículos se estiran para ajustar el contenedor
- flex-start - Los elementos se colocan en la parte superior del contenedor
- flex-end - Los artículos se colocan en la parte inferior del contenedor
- center - Los artículos se colocan en el centro del contenedor (verticalmente)
- baseline - Los elementos se colocan en la línea de base del contenedor

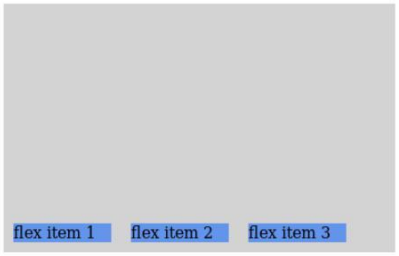
Ejemplos al utilizar los diferentes valores sobre los elementos del ejemplo anterior:

align-items: stretch;

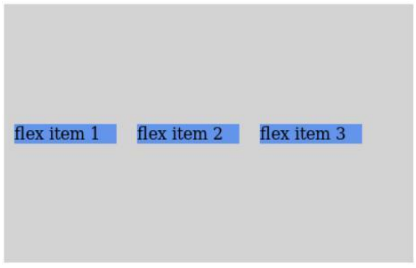
align-items: flex-start;



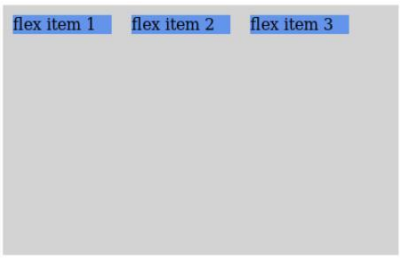
align-items: flex-end;



align-items: center;



align-items: baseline;



flex-wrap

La propiedad flex-wrap especifica si los elementos flexibles deben envolverse o no, si no hay suficiente espacio para ellos en una línea flex.

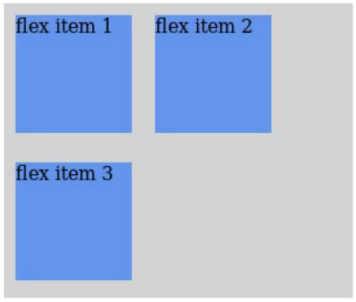
Los valores posibles son los siguientes:

- Nowrap - Valor predeterminado. Los elementos flexibles no se envolverán
- Wrap - Los artículos flexibles se envolverán si es necesario
- Wrap-reverse - Los artículos flexibles se envolverán, si es necesario, en orden inverso

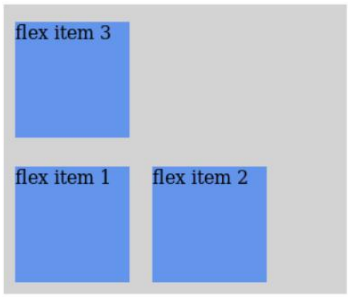
Ejemplos al utilizar los diferentes valores sobre los elementos del ejemplo anterior aplicando también un height de 100px a .flex-item:

flex-wrap: nowrap;

flex-wrap: wrap;



flex-wrap: wrap-reverse;



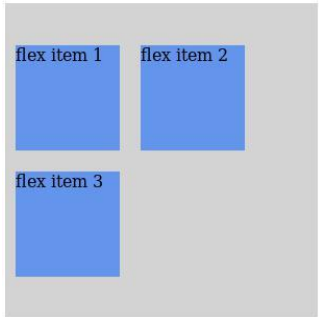
align-content

La propiedad align-content modifica el comportamiento de la propiedad flex-wrap. Es similar a align-items, pero en lugar de alinear elementos flexibles, alinea las líneas flexibles.

Los valores posibles son los siguientes:

- stretch - Valor predeterminado. Las líneas se extienden para ocupar el espacio restante
- flex-start - Las líneas están empacadas hacia el inicio del contenedor flexible
- flex-end - Las líneas se empaquetan hacia el extremo del contenedor flexible
- center - Las líneas están empaquetadas hacia el centro del contenedor flexible
- space-between - Las líneas están uniformemente distribuidas en el contenedor flexible
- space-around - Las líneas están distribuidas uniformemente en el contenedor flexible, con espacios de tamaño medio en cada extremo

El siguiente ejemplo muestra el resultado de usar el valor del centro:



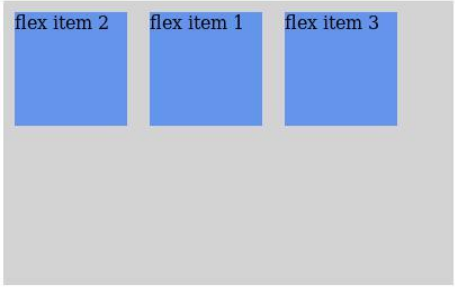
Propiedades de los elementos flexibles.

Orden

La propiedad order especifica el orden de un elemento flexible en relación con el resto de los elementos flexibles dentro del mismo contenedor:

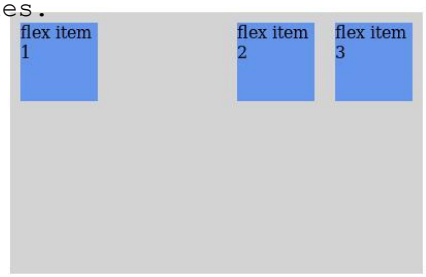
```
.flex-container {
    display: -webkit-flex;
    display: flex;
    width: 400px;
    height: 250px;
    background-color: lightgrey;
}

.flex-item {
    background-color: cornflowerblue;
    width: 100px;
    height: 100px;
    margin: 10px;
}

.first {
    -webkit-order: -1;
    order: -1;
}
```

```
<div class="flex-container">
  <div class="flex-item">flex item 1</div>
  <div class="flex-item first">flex item 2</div>
  <div class="flex-item">flex item 3</div>
</div>
```



Margen

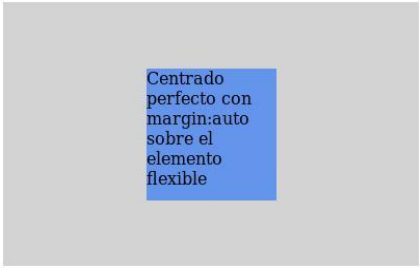
Estableciendo margin: auto; absorberá el espacio extra. Se puede utilizar para empujar elementos flexibles en diferentes posiciones.



En el siguiente ejemplo establecemos margin-right: auto; para el primer elemento flex. Esto hará que todo el espacio extra sea absorbido a la derecha de ese elemento:

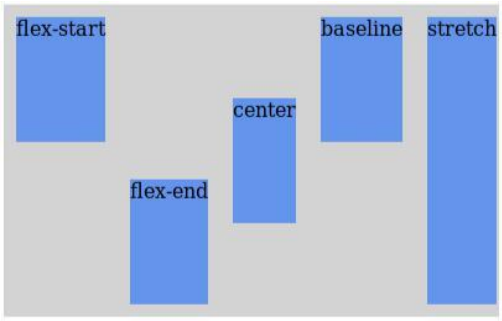
Centrado perfecto

En el siguiente ejemplo resolveremos un problema casi diario: centrado perfecto.



Es muy fácil con flexbox. margin: auto; hará que el elemento quede perfectamente centrado en ambos ejes:

Alineado



La propiedad align-self de los elementos flexibles anula la propiedad align-items del contenedor flex para ese elemento. Tiene los mismos valores posibles que la propiedad align-items.

El siguiente ejemplo establece diferentes valores de align-self para cada elemento flexible:

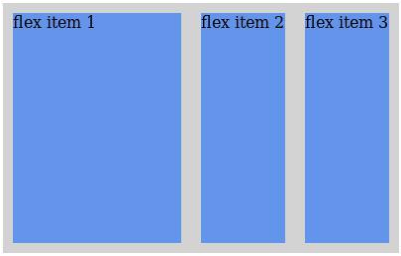
Propiedad flex

La propiedad flex especifica la longitud del elemento flex, en relación con el resto de los elementos flexibles dentro del mismo contenedor.

En el siguiente ejemplo, el primer elemento flexible consumirá 2/4 del espacio libre y los otros dos elementos flexibles consumirán 1/4 del espacio libre cada uno:

```
.flex-container {
    display: -webkit-flex;
    display: flex;
    width: 400px;
    height: 250px;
    background-color: lightgrey;
}

.flex-item {
    background-color: cornflowerblue;
    margin: 10px;
}

.item1 {-webkit-flex: 2;flex: 2;}

.item2 {-webkit-flex: 1;flex: 1;}

.item3 {-webkit-flex: 1;flex: 1;}
```

```
<div class="flex-container">
  <div class="flex-item item1">flex
item 1</div>
  <div class="flex-item item2">flex
item 2</div>
  <div class="flex-item item3">flex
item 3</div>
</div>
```



Media queries (consultas de medios)

La regla @media, introducida en CSS2, permitió definir diferentes reglas de estilo para diferentes tipos de medios.

Ejemplos: Puede tener un conjunto de reglas de estilo para pantallas de computadora, una para impresoras, una para dispositivos portátiles, otra para dispositivos de tipo televisión, etc.

Desafortunadamente estos tipos de medios nunca recibieron mucha compatibilidad con dispositivos, aparte del tipo de medios de impresión. Las consultas de medios en CSS3 amplían la idea de los tipos de medios CSS2: En lugar de buscar un tipo de dispositivo, buscan la capacidad del dispositivo.

Las consultas de medios se pueden utilizar para comprobar muchas cosas, tales como:

- Anchura y altura de la ventana del dispositivo
- Anchura y altura del dispositivo
- Orientación (esta tableta / teléfono en apaisado o vertical?)
- Resolución de pantalla

El uso de consultas de medios es una técnica popular para entregar una hoja de estilos adaptada a tabletas y teléfonos.

Sintaxis

Una media query consiste en un tipo de medio que puede contener una o más expresiones, que se resuelven a true o false.

```
@media not|only tipo_de_medio and (expresiones)
    { Código CSS;
}
```

El resultado de la consulta es cierto si el tipo de medio especificado coincide con el tipo de dispositivo en el que se muestra el documento. Cuando una consulta de medios es verdadera, se aplican las reglas de estilo o de hoja de estilo correspondientes, siguiendo las reglas en cascada habituales.

A menos que utilice los operadores not o only, el tipo de medio es opcional y el tipo all se usará por defecto.

También puede tener diferentes hojas de estilo para diferentes soportes:

```
 <link rel="stylesheet" media="tipo_de_medio and|not|only
(expresiones)" href="para_impresoras.css">
```

Ejemplos:

Una forma de usar consultas de medios es tener una sección CSS alternativa dentro de su hoja de estilo.

El siguiente ejemplo cambia el color de fondo a lightgreen si la ventana es de 480 píxeles o más de ancho (si la vista es menor de 480 píxeles, el color de fondo será de color rosa):

```
@media screen and (min-width: 480px) {
    body {
```

```
        background-color: lightgreen;
    }
}
```

Cuando el ancho del navegador es entre 520 y 699px, se aplicará un icono de correo electrónico a cada enlace de correo electrónico:

```
@media screen and (max-width: 699px) and (min-width: 520px)
    { ul li a {
        padding-left: 30px;
        background: url(email-icon.png) left center no-repeat;
    }
}
```

Para anchos de navegador por encima de 1151px, volveremos a agregar el icono como lo usamos antes.

Aquí, no tenemos que escribir un bloque adicional de consulta de medios, solo podemos agregar una consulta de medios adicional a nuestra ya existente usando una coma (esto se comportará como un operador OR):

```
@media screen and (max-width: 699px) and (min-width: 520px), (min-width:
1151px) {
    ul li a {
        padding-left: 30px;
        background: url(email-icon.png) left center no-repeat;
    }
}
```