Полное руководство по CSS Flexbox

Содержание:

Свойства Родителя (flex container)

display

Это свойство определяет flex-контейнер. Это свойство применимо ко всем прямым потомкам (элементам) этого контейнера.

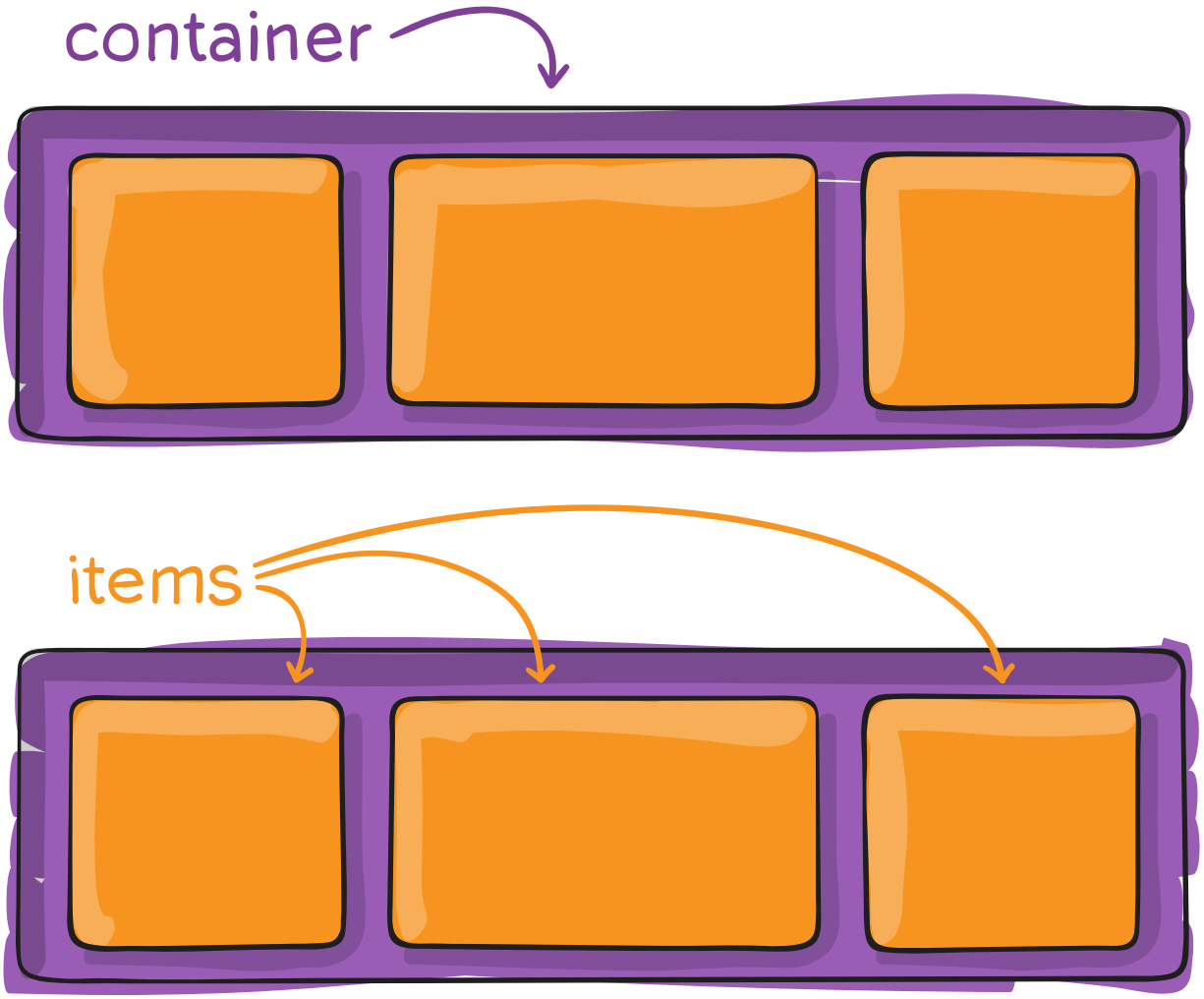
После применения свойства `display` со значением `flex` к основному родительскому контейнеру дочерние элементы располагаются в один ряд слева на право при этом занимают всю высоту родительского контейнера. То есть родительский элемент становится **flex-контейнером**, а дочерние элементы находящееся в нем — **flex-елементами**.

Также при определении flex-контейнера появляется такое понятие как ось — главная и поперечная. Данные оси отвечают за то как и в каком порядке будут располагаться flex-элементы внутри flex-контейнера. По умолчанию flex-элементы располагаются вдоль главной горизонтальной оси слева на право.

При этом контейнер со свойством `flex` является блочным элементом — то есть блок растягивается на всю ширину страницы. Но существует второе свойство `inline-flex` — при его использовании flex-контейнер становится строчным элементом и его ширина будет равняться ширине находящихся в нем элементам.

Открыть в CodePen

```
.container {
  display: flex; /* или inline-flex - ширина основного контейнера по содержимому */
}
```

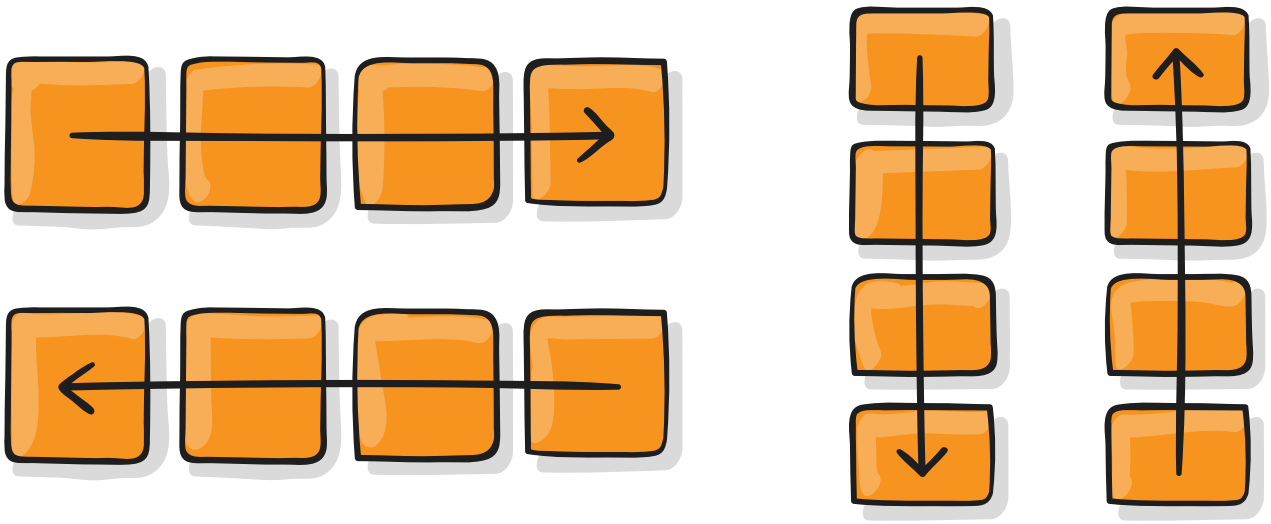


flex-direction

Это свойство отвечает за то как и в каком порядке будут располагаться flex элементы внутри flex контейнера.

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

Copy



`row` — то есть в главной оси flex-елементы располагаются в горизонтальном порядке слева на право, а в поперечной — в вертикальном порядке снизу в верх. `row-reverse` — изменяет направление основной оси слева на право; `column` — главная ось меняется местами, элементы располагаются сверху в низ; `column-reverse` — разворачивает главную ось, элементы располагаются с низу в верх;

Таким образом flex-элементы располагаются всегда вдоль главной оси и мы можем управлять лишь положением главной оси. Поперечная же ось поворачивается вместе с основной в зависимости от того какое положение для основной оси мы задали. То есть поперечная ось будет всегда направлена слева в низ или с верху на право.

flex-wrap

`nowrap` — значение по-умолчанию, элементы располагаются на одной строке и не переносятся. При большом количестве элементов появляется горизонтальная полоса прокрутка (`row` или `row-reverse`); `wrap` — данное значение разрешает перенос элементов на новую строку, элементы не ужимаются и имеют правильные отступы, которы были заданы ранее; `wrap-reverse` — используется крайне редко, данное свойство размещает элементы в обратном направлении;

justify-content

Данное свойство отвечает за распределение элементов вдоль главной оси. `flex-start` — свойство по умолчанию, означает то что все элементы будут прижиматься к началу главной оси; `flex-end` — все элементы прижимаются к концу главной оси; `center` — элементы располагаются по центру; `space-between` — располагает равномерно элементы вдоль главной оси, при этом первый элемент прижимается к началу, а последний ко концу главной оси. Расстояние между элементами становится одинаковым благодаря тому что браузер высчитывает его самостоятельно; `space-around` — также задает одинаковое расстояние между элементами, однако оно еще задает отступы для первого и последнего элемента. Данные отступы будут становить половине расстояния между остальными элементами.

align-items (выравнивание вдоль поперечной оси)

Отвечает за выравнивание элементов вдоль поперечной оси `stretch` — значение по умолчанию, растягивает элементы на всю ширину родительского контейнера; `flex-start` — элементы будут выравниваться по перечной оси и прижмутся к ее началу; `flex-end` — элементы прижмутся к концу поперечной оси; `center` — элементы расположатся по центру поперечной оси; `baseline` — выравнивает элементы по базовой линии.

align-content

Имеет точно такие же значения как и `justify-content` + `stretch`. Это свойство выравнивает элементы вдоль поперечной оси и работает только в том случае если у нас задано свойство `wrap` либо `wrap-reverse`. `stretch` — значение по умолчанию, растягивает элементы по всей высоте контейнера `flex-start` — размещает ряды к началу поперечной оси; `flex-end` — размещает ряды к концу поперечной оси; `space-between` — будет располагать наши элементы вдоль поперечной оси при этом первый ряд прижмется к началу поперечной оси, а последний к концу. Расстояние между рядами высчитывается браузером автоматически. `space-around` — наши элементы также будут располагаться вдоль поперечной оси, но для первого ряда задастся некий отступ с верху, а для последнего ряда будет задан отступ снизу и данный отступ будет равен половине отступу между нашими рядами.

Также необходимо знать что свойство `align-items` может влиять на расположение рядов, но может оно влиять только в том случае если свойству `align-content` задано значение по-умолчанию `stretch`.

Свойства дочерних элементов (flex items)

align-self

`stretch` — значение по умолчанию `flex-start` — размещает элемент в начале поперечной оси; `flex-end` — размещает элемент в конце поперечной оси; `center` — размещает элемент по центру поперечной оси; `baseline` — размещает элемент по базовой линии;

order

С помощью этого свойства мы можем изменять порядок элементов в потоке без изменения структуры html. Свойство `order` принимает в качестве значения порядковой номер элемента — это целое положительное или отрицательное число. По-умолчанию каждый flex элемент имеет значение раное нулю.

flex-basis

Свойство которые отвечает за базовые размеры flex элементов. Данное свойство может принимать значение в любых единицах измерения — px, %, em. `auto` — значение по умолчанию, при даном значение каждое значение берется из свойств `width` или `height`. В том случае если свойство `flex-basis` принимает какое-либо значение `width` и `height` будут проигнорированны — так как `flex-basis` имеет больший приоритет чем данное свойство. Самое главное отличие `flex-basis` являет то что при изменении главной оси — этой свойство реагирует на данное поведение. То есть если главная ось направлена слева на право то свойство `flex-basis` отвечает за ширину, однако если мы изменим направление главной оси на снизу в верх, то данное свойство будет отвечать за высоту flex элементов.

flex-grow

Отвечает данное свойство за растягивание flex элементов. По умолчанию данное свойство имеет значение равное нулю, но если мы зададим данному свойству значение отличимое от нуля то ширина flex элемента будет увеличивать и при этом он будет занимать все место свободное место внутри flex контейнера

Отступы margin во flex элементах

По сравнении с блочной моделью внешние отступы между элементами не схлопываются а складываются. При помощи свойства `margin` мы можем отцентрировать флекс элементы по горизонтали или вертикали задав ему значение `auto`. Важно понимать что свойство `margin:auto` ломает выравнивание элементов по главной и поперечной оси и таким образом такие свойства как `justify-conten, align-items, aling-self` не будут работать. В первую очередь в родительском контейнере определяются элементы у которых задан `margin:auto` и все свободное место во flex контейнере отдается данным элементам и распределяется поровну

Управление размерами во flex элементах

flex-basis

Данное свойство отвечает за базовый размер Flex элемента. Это свойство можно задать в двух единицах измерения: px, %, em и др. Также данное свойство имеет по умолчанию значение `auto`. При данном значение базовый размер Flex элемента берется из свойств `with` или `height`. Однако если в свойстве flex-basis`будут применятся какие-либо значения с единицами измерения, то свойство`with`или`heigh` будут проигнорированы, потому что данное свойство имеет больший приоритет.

При изменении направления главной оси свойство `flex-basis` реагирует на данное поведение. То есть если расположение flex элементов направлены слева на право то данное свойство отвечает за ширину данного элемента. Однако если изменить направление главной оси сверху в низ, то данное свойство будет отвечать за высоту flex элемента.

flex-grow

Это свойство отвечает за растягивание flex элементов. По умолчанию данное свойство имеет значение равное нулю. Но если мы зададим этому свойству положительное значение не равное нулю то при этом flex элемент будет увеличиваться и занимать все свободное место. Если данное свойство задать для нескольких flex элементов в одном flex контейнере то эти элементы начнут делить свободное место между собой. Например если первому элементу задать значение 1 а последнему 2, то последний элемент займет в два раза больше свободного пространства чем первый элемент.

Также необходимо учитывать что при задании свойства `flex-basis` или `flex-grow` Flex элемента может быть больше базового размера, однако меньше его он не будет, однако это актуально только когда flex контейнеру задано свойство `flex-wrap` со значение `wrap` или `wrap-reverse`.

flex-shrink

Задает коэффициент сжатия элементов во flex контейнере относительно друг-друга. По умолчанию значение данного свойства равно 1.

Расчет flex элементов

Первое что мы должны выполнить при расчете размеров дочерних flex элементов это вычислить количество свободного места.

Расчет flex-grow

Открыть в CodePen

Если базовый размер составляет `150px` (flex-basis: 150px), а общее количество элементов 5, то общая ширина которую занимают элементы будет составлять `750px` (150px x 5). Значит свободное место будет составлять `450px` (1200px — 750px). Общая формула расчета свободного места будет выглядеть так:
Свободное место = Ширина flex контейнера — Сумма базовых flex элементов
1200 — 750 = 450

После этого мы должны рассчитать `часть свободного места` по следующей формуле:
Часть свободного места = Свободное месть / Сума `flex-grow` всех flex элементов
450 / (1 + 5) = 75

И наконец мы можем рассчитать размер flex элемента по следующей формуле:
Размер flex элемента = Базовый размер + (Часть свободного места x flex-grow)

Расчет flex-shrink

Открыть в CodePen

Первым делом высчитываем отрицательное значение:
Отрицательное значение = Ширина flex-контейнера — Сумма базовых размеров flex-элементов
1200 — (300 x 5) = 300 (без знака минус)

Далее высчитываем сумму базовых размеров:
Сумма размеров = (Базовый размер 1 x flex-shrink-1) + (Базовый размер n x flex-shrink-n)
(300 x 3) + (300 x 2) = 1500

Затем мы вычисляем коэффициент сжатия:
Коэффициент сжатия = (Базовый размер x flex-shrink-n) / Сумма размеров
(300 x 3) / 1500 = 0.6
(300 x 2) / 1500 = 0.4

Размер элемента вычесляем по формуле:
Размер элемента = Базовый размер — (Коэффициент сжатия * Отрицательное значение)
300 — (0.6 x 300) = 80
300 — (0.4 x 300) = 120

Оставьте комментарий

Для отправки комментария вам необходимо авторизоваться.

Категории

Верстка (2)

Инструменты (1)

Плагины (2)

Поддержка (5)

Производительность (3)

Разработка темы (5)