**Флексбокс** — это *CSS*-механизм, созданный специально для быстрого и удобного распределения блоков, их выравнивания и распределения свободного места между ними. Свойство ***display*** со значением ***flex*** превращает элемент во флекс-контейнер, а все его вложенные элементы – во флекс-элементы.

В то время как *display: block/inline/inline-block* определяет поведение конкретного компонента, *display: flex* задает **контекст** для всех вложенных флекс-элементов – особые правила поведения внутри себя (это удобнее). Он делает блоки “резиновыми” – элементы могут сжиматься и растягиваться по заданным правилам, занимая нужное пространство, или автоматически выстраиваться в несколько строк или столбцов, занимая всё предоставленное место.

Уровень вложенности неограничен – дочерние элементы сами могут быть флекс-контейнерами. Настройки флекс-контейнеров более высокого уровня не наследуются.



При преобразовании элемента во флексбокс-контейнер, внутри него появляется две оси – основная и поперечная.

По умолчанию все флекс-элементы выстраиваются в одну строку.

Свойство ***flex-direction*** позволяет задать направление основной оси:

*flex-direction: row; // слева направо (по умолчанию)*

*flex-direction: column; // сверху вниз*

*flex-direction: row-reverse; // справа налево*

*flex-direction: column-reverse; // снизу вверх*

Свойство ***justify-content*** позволяет распределить свободное пространство на основной оси между флекс-элементами:

*justify-content: start; // сначала элементы, потом оставшееся пустое пространство*

*justify-content: end; // сначала пустое пространство, потом элементы*

*justify-content: center; // элементы по середине, пустое пространство поровну с двух сторон*

*// первый элемент прижат к началу контейнера, последний – к концу,*

*// а остальные расставлены между ними с одинаковыми промежутками*

*justify-content: space-between;*

*// все пустое пространство делится на промежутки. Элементы получают отступы по обе стороны,*

*// равные этому промежутку. Получается между элементами отступ в два промежутка,*

*// а перед первым и после последнего элемента отступ в один промежуток*

*justify-content: space-around;*

*justify-content: space-evenly; // все отступы равной величины*

По умолчанию все флекс-элементы растягиваются по поперечной оси до размера наибольшего элемента (в том числе изображения).

Свойство ***align-items*** позволяет задать логику выравнивания флекс-элементов по поперечной оси:

*align-items: stretch // элементы растянуты от края до края (по умолчанию)*

*align-items: start // элементы прижаты к началу*

*align-items: end // элементы прижаты к концу*

*align-items: center // элементы по середине*

*align-items: baseline // элементы выравниваются по базовой линии текста*

**Базовая линия текста** – воображаемая линия, проходящая по нижнему краю знаков шрифта первой строки.



Свойство ***flex-wrap*** позволяет переносить элементы флекс-контейнера, которые в него не влезают:

*flex-wrap: nowrap // элементы не переносятся (по умолчанию)*

*flex-wrap: wrap // элементы переносятся на следующую строку сверху вниз*

*flex-wrap: wrap-reverse // элементы переносятся на следующую строку снизу вверх*

Свойство ***row-gap*** позволяет задать расстояние между строками элементов флекс-контейнера. Свойство ***column-gap*** позволяет задать расстояние между столбцами элементов флекс-контейнера. Оба эти значения можно записать кратко через свойство ***gap***. Такие отступы являются фиксированными и не меняются в результате сжатия или растяжения. Его удобно использовать для выравнивания карточек.

*row-gap: 15px; // задать расстояние между строками*

*column-gap: 30px; // задать расстояние между столбцами*

*gap: 25px 15px; // задать расстояние между строками 25px и столбцами 15px*

*gap: 25px; // задать расстояние между строками и столбцами по 25px*

Свойство ***align-content*** позволяет задать логику выравнивания элементов флекс-контейнера по поперечной оси при переносе на новую строку. Для его работы должен быть включен ***flex-wrap*** и свободное пространство.

*// строки растянуты одинаково так, чтобы занять*

*// все доступное пространство родителя (по умолчанию)*

*align-content: stretch;*

*align-content: start; // все строки располагаются у начала родителя*

*align -content: end; // все строки располагаются у конца родителя*

*align -content: center; // все строки располагаются по центру родителя*

*// первая строка прижата к началу контейнера, последняя – к концу,*

*// а остальные расставлены между ними с одинаковыми промежутками*

*align -content: space-between;*

*// все пустое пространство делится на промежутки. Строки получают отступы по обе стороны,*

*// равные этому промежутку. Получается между строками отступ в два промежутка,*

*// а перед первой и после последней строки отступ в один промежуток*

*align -content: space-around;*

*align -content: space-evenly; // все отступы равной величины*

Свойство ***order*** позволяет задать вес элементу флекс-контейнера. Элементы располагаются в порядке увеличения веса. По умолчанию вес равен нулю. Следует использовать его только в случаях, когда отсутствует доступ в *HTML*.

Свойство ***align-self*** позволяет задать конкретному элементу флекс-контейнера логику выравнивания по поперечной оси. Может принимать те же значения, что и ***align-items***.

Свойство ***flex-grow*** флекс-элемента указывает, может ли он вырастать при наличии свободного места, игнорируя размер контента. По умолчанию значение равно нулю (расширяться нельзя).

Если всем флекс-элементам задать значения 1, то свободное пространство равномерно распределится между ними. Но стоит одному флекс-элементу задать значение 2, то он заберет в 2 раза больше свободного пространства, чем флекс-элементы со значением 1.

Например, с помощью этого свойства закрепим ***footer*** внизу страницы, независимо от высоты контента:

*body {*

*min-height: 100%; // задаем размер равный всей видимой части браузера*

*display: flex; // преобразуем во флекс-контейнер*

*flex-direction: column; // задаем направление сверху вниз*

*}*

*.main {*

*flex-grow: 1; // разрешаем занимать все свободное пространство*

*}*

Свойство ***flex-basis*** флекс-элемента позволяет задать размер элемента (в относительных или абсолютных значениях) до того, как на него начнет действовать расширение. Например, в первой строке поместилось три карточки, а на второй – только две, которые нужно растянуть на всю ширину:

*.content {*

*display: flex; // преобразуем во флекс-контейнер*

*flex-wrap: wrap; // разрешаем перенос*

*}*

*.card {*

*flex-basis: 300px; // задаем минимальную ширину*

*flex-grow: 1; // разрешаем занимать все свободное пространство*

*}*

Свойство ***flex-shrink*** флекс-элемента указывает, может ли он сжиматься при нехватке свободного места, игнорируя размер контента. По умолчанию значение равно нулю (сжиматься нельзя, размер равен значению *flex-basis*). Работает аналогично ***flex-grow*** – чем больше значение, тем быстрее сжимается.

Свойство ***flex*** объединяет все три свойства ***flex-grow***, ***flex-shrink***, ***flex-basis***.

*flex: 0 1 auto;*

**Грид** — это *CSS*-механизм, созданный специально для быстрого и удобного распределения элементов на веб-страницах. Свойство ***display*** со значением ***grid*** превращает элемент в грид-контейнер, а все его вложенные элементы – в грид-элементы.

Флексбоксы ориентированы на контент и используются для мест, где нужно выравнивание в строке или колонке. Гриды же ориентированы на структуру и используются для создания общей сетки страницы. Они взаимозаменяемы, но в разных ситуациях удобнее использовать тот или иной инструмент.

**Грид-контейнер** – это набор невидимых вертикальных и горизонтальных **грид-линий**, которые пересекаются и создают области, в которых могут располагаться грид-элементы. Грид-линии имеют нумерацию, начиная с единицы, а также могут принимать отрицательные значения.

По умолчанию грид-контейнер состоит из одного столбца и одной строки, которые занимают весь его размер. При добавлении элементов, для каждого из них создастся новая строка, но элементы все еще будут занимать один столбец.

Свойство ***display*** со значением ***inline-grid*** превращает элемент в грид-контейнер, который будет вести себя как строчный элемент и занимать только то пространство, которое необходимо, чтобы разместить грид-элементы.

Свойство ***grid-template-columns*** позволяет создавать вертикальные грид-линии внутри грид-контейнера. Свойство ***grid-template-rows*** позволяет создавать горизонтальные грид-линии внутри грид-контейнера. Значения могут задаваться в пикселях, процентах или рассчитываться автоматически. Между этими невидимыми грид-линиями будет сформированы **грид-ячейки**.

*.container {*

*display: grid;*

*grid-template-columns: 190px 200px 270px; // создаем три колонки*

*grid-template-rows: repeat(3, 150px); // создаем три столбца через repeat*

*}*

По умолчанию распределение элементов происходит слева направо и сверху вниз. Если элементов больше, чем созданных нами ячеек, то будут сформированы неявные ячейки со значениями *auto*.



Свойство ***row-gap*** позволяет задать расстояние между строками грид-ячеек. Свойство ***column-gap*** позволяет задать расстояние между столбцами грид-ячеек. Оба эти значения можно записать кратко через свойство ***gap***. Такие отступы являются фиксированными и не меняются в результате сжатия или растяжения.

**Фракция**, или ***fr*** – единица доступного пространства, которое можно разделить на сколько угодно частей. Она придумана специально для гридов и её можно комбинировать с фиксированными размерами. Элемент размером 2*fr* будет занимать вдвое больше свободного пространства, чем 1*fr*, а 3*fr* – втрое. Если других фракций нет, забирает все свободное пространство.

*.container {*

*width: 600px;*

*display: grid;*

*grid-template-columns: 100px 2fr 3fr; // получим 3 ячейки с размерами 100px, 200px и 300px*

*}*

По умолчанию каждый грид-элемент занимает одну грид-ячейку. Изменить занимаемый размер грид-элемента можно с помощью свойств ***grid-column-start*** и ***grid-column-end*** для вертикальных грид-линий, ***grid-row-start*** и ***grid-row-end*** для горизонтальных грид-линий. Свойства ***grid-column*** и ***grid-row*** позволяют использовать короткую запись. С помощью этих свойств можно накладывать грид-элементы друг на друга.

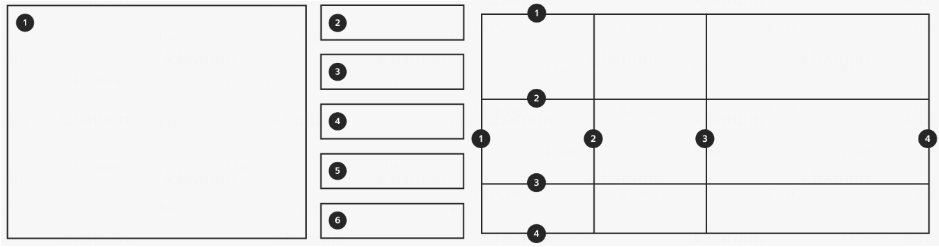
*.block\_size\_big {*

*grid-column-start: 1; // начинается с 1 линии*

*grid-column-end: 3; // заканчивается до 3 линии*

*grid-row: 1/6; // сокращенная запись*

*}*



Грид-линиям можно задавать имена, при этом цифровые названия останутся доступными. Имя указывается в квадратных скобках и может быть любым, кроме ***span***.

*grid-template-rows: [aside-start] 300px [aside-end]; // именуем линии*

*grid-row: aside-start / 4; // задаем размер элемента через имя*

Ключевым словом ***span*** указывают, какое количество строк или столбцов должен занимать элемент до или после какой-то линии.

*.block {*

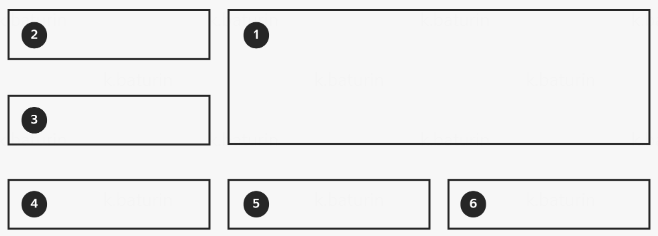
*grid-column-start: 2; // элемент начнется со 2 колонки*

*grid-column-end: span 2; // элемент займет 2 колонки*

*grid-row-start: span 2; // элемент займет 2 строки*

*grid-row-end: 3; // элемент закончится до 3 строки*

*}*



**Грид-область** – это прямоугольная область, которая состоит из одной или нескольких грид-ячеек. Задать имя такой области позволяет свойство ***grid-area***.

*.header {*

*grid-area: header;*

*}*

Свойство ***grid-template-areas*** позволяет разделить грид-контейнер на области. Можно задать только одну область для каждого имени, разъединенные области создать не получится. Каждую ячейку сетки необходимо заполнить, иначе свойство не сработает. Если какую-то область нужно оставить пустой, вместе имени ставится точка.

*.container {*

*display: grid;*

*grid-template-columns: repeat(3, 1fr); // задаем столбцы*

*grid-template-rows: repeat(4, 1fr); // задаем строки*

*grid-template-areas: // распределяем области*

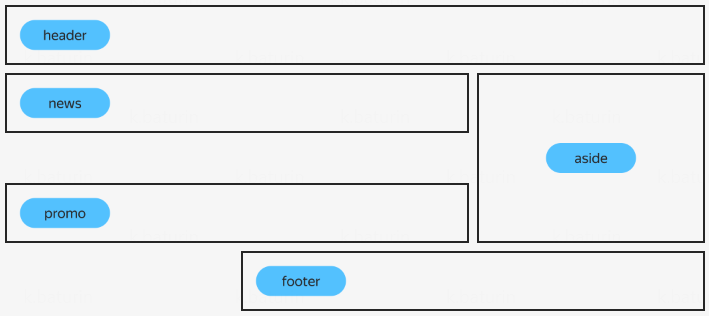
*"header header header"*

*"news news aside"*

*"promo promo aside"*

*". footer footer";*

*}*



Свойства ***align-content*** и ***justify-content*** позволяют выровнять сетку в грид-контейнере по высоте и ширине соответственно. Доступны значения *stretch*, *flex-start*, *flex-end*, *center*, *space-between*, *space-around* и *space-evenly*. Свойство ***place-content*** позволяет кратко записать оба эти свойства.

*.container {*

*align-content: center;*

*justify-content: space-between;*

*place-content: center space-between; // то же самое*

*place-content: center; // если оба значения совпадают*

*}*

Свойства ***align-items*** и ***justify-items*** позволяют выровнять элементы во всех вложенных грид-областях по высоте и ширине соответственно. Доступны значения *stretch*, *start*, *end* и *center*. Свойство ***place-items*** позволяет кратко записать оба эти свойства.

*.container {*

*align-items: center;*

*justify-items: space-between;*

*place-items: center space-between; // то же самое*

*place-items: center; // если оба значения совпадают*

*}*

Свойства ***align-self*** и ***justify-self*** позволяют выровнять элементы в одной конкретной грид-области по высоте и ширине соответственно.

**Поток** – это связка элементов и их зависимость друг от друга. Взаимное расположение элементов в потоке называется **позиционированием**. За него отвечает свойство ***position***. По умолчанию для всех элементов используется **статическое позиционирование** и свойства смещения игнорируются:

* Элемент находится в потоке. Соседние элементы знают, где он расположен, и реагируют на его смещение.
* Элемент перемещается визуально только за счет отступов. Так как отступы – часть блочной модели, то с точки зрения браузера мы двигаем не элемент, а меняем его размеры за счет отступов.

*position: static;*

**Относительное позиционирование** позволяет оставить элемент в потоке и смещаться относительно самого себя (технически для остальных элементов место забронировано, но смещено визуально).

*div {*

*position: relative;*

*top: 20px; // сместить элемент вниз*

*right: 40px; // сместить элемент влево (игнор)*

*bottom: 10px; // сместить элемент вверх (игнор)*

*left: 10px; // сместить элемент вправо*

*}*

Если *left* и *right* противоречат друг другу, то *right* игнорируется.

Если *top* и *bottom* противоречат друг другу, то *bottom* игнорируется.

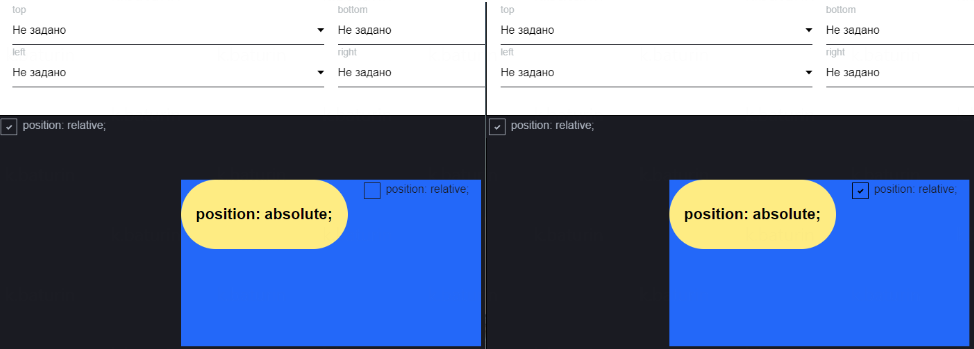
Свойство ***inset*** объединяет в себе все четыре свойства ***top***, ***right***, ***bottom*** и ***left***, по аналогии с ***margin***.

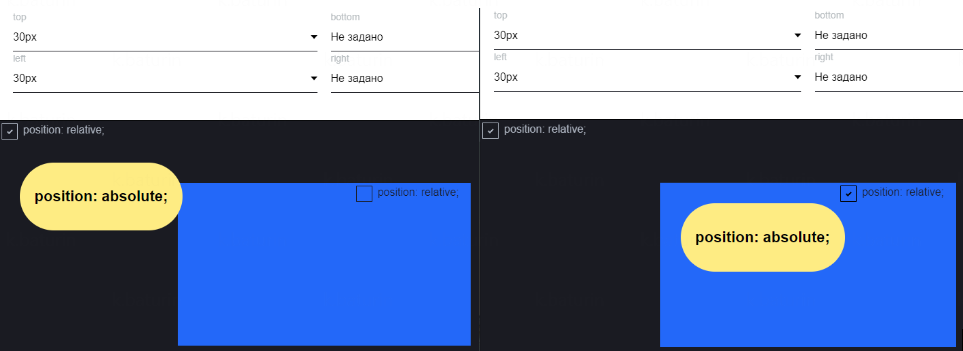
**Абсолютное позиционирование** позволяет убрать элемент из потока и располагает его относительно границ родителя.

Если родительский элемент спозиционирован (значение *position* отлично от *static*), то элемент с абсолютным позиционированием будет размещен согласно свойствам смещения относительно границ родителя. Например, картинка займет всю площадь родителя.

Если у родительского элемента не менялось статическое позиционирование, то элемент с абсолютным позиционированием попробует найти другие родительские элементы с нестатическим позиционированием. Если не найдет, то расположится относительно границ *body*.

*position: absolute;*





Пространство веб-страницы не двумерное, а трехмерное. Ось *Z* направлена перпендикулярно экрану, из глубины монитора по направлению к пользователю. Когда при позиционировании мы вырываем элементы из потока, они могут перекрывать друг друга. Для управления близостью элементов используется свойство ***z-index***. Чем больше целое число, тем ближе оно к пользователю. По умолчанию равно нулю.

Свойство близости не действует на элементы со статическим позиционированием, однако если элемент находится во *flex*- или *grid*-контейнере, то оно сработает.

**Фиксированное позиционирование** позволяет убрать элемент из потока и располагает его относительно окна браузера. Игнорирование противоречащих свойств смещения меняется на приоритетность (например, свойства *left* и *right* позволяют растянуть шапку на весь экран). Подходит для фиксированной шапки сайта или всплывающего окна (***popup***).

*position: fixed;*

**Липкое позиционирование** позволяет задать момент, в который элемент будет вырван из потока и “прилипнет” к окну браузера при прокрутке страницы. Такой элемент “отлипнет” по достижению границ родительского элемента. Обязательно требует свойство смещения.

*position: sticky;*

*top: 20px;*