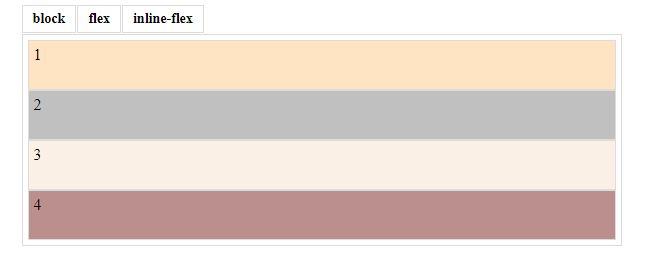
# Модель Flexbox

## Что такое Flexbox?

***Flexbox (CSS Flexible Box Layout)*** — это модель отображения содержимого страницы, при которой вы получаете набор гибких элементов внутри контейнера, имеющего свойство ***display*** со значением ***flex*** или ***inline-flex.***

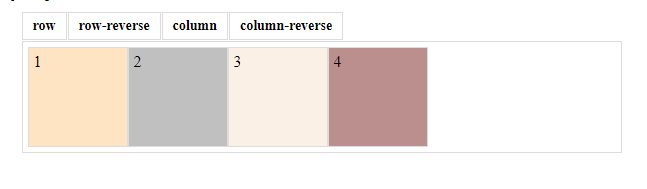
## Свойства flex-контейнера

### Свойство display:



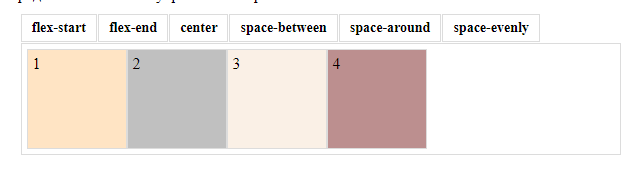
### Свойство flex-direction:

Направление ***row*** или ***row-reverse*** выстраивает дочерние элементы по горизонтали в направлении слева направо или справа налево. Значения ***column*** или ***column-reverse*** меняют направление размещения блоков на вертикальное: сверху вниз или снизу вверх.



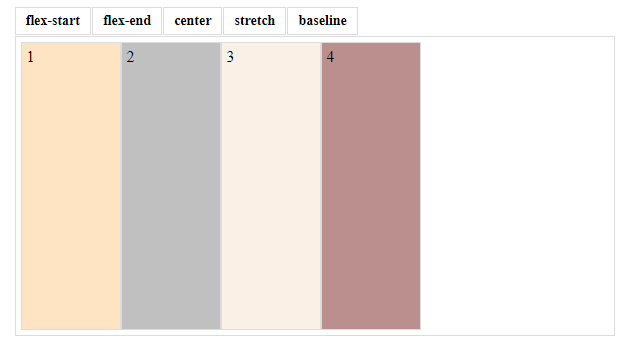
### Свойство justify-content:

определяет выравнивание ***flex***-элементов вдоль главной оси. Оно часто используется при построении колонок, т.к. позволяет распределить контент внутри контейнера.



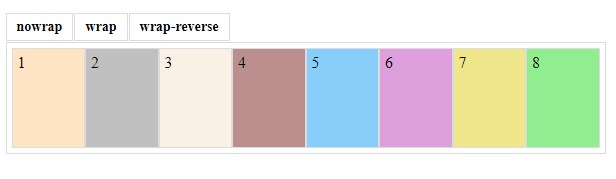
### Свойство align-items:

определяет, каким образом выравнивать ***flex***-элементы вдоль поперечной оси.



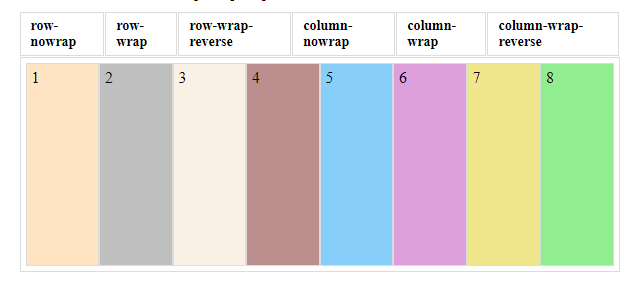
### Свойство flex-wrap:

отвечает за то, как будут размещаться вложенные ***flex***-элементы внутри контейнера, если они не помещаются в одну строку.



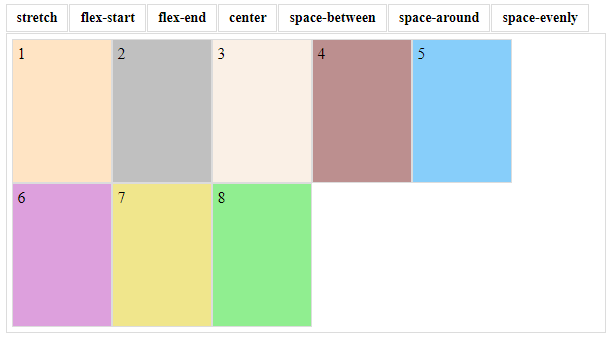
### Свойство flex-flow:

предназначено для объединения свойств ***flex-direction*** и ***flex-wrap***. По умолчанию имеет значение ***flex-flow: row nowrap***. Имеет смысл использовать его при переопределении обоих этих значений или одного из них.



### Свойство align-content:

используется только в многострочном режиме (т.е. в случае, когда свойство ***flex-wrap*** имеет значение ***wrap*** или ***wrap-reverse***). Оно определяет, каким образом ряды ***flex***-элементы будут выровнены по вертикали, а также то, как они будут делить между собой все пространство ***flex***-контейнера.



## Свойства flex-элементов

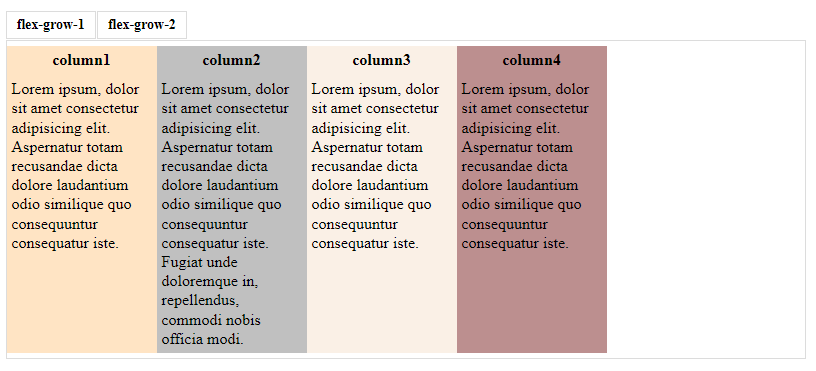
Свойства ***flex***-элементов позволяют управлять каждым из элементов внутри ***flex***-контейнера по отдельности. Это, в первую очередь, свойства, которые управляют размерами дочерних элементов, их растяжимостью и сжимаемостью, порядком отображения вне зависимости от ***html***-разметки, а также возможностью переопределить размещение ***flex***-элементов, заданное свойствами ***flex***-контейнера

### Свойство flex-grow:

определяет фактор (коэффициент) растяжимости ***flex***-элемента при наличии избыточного пространства во ***flex***-контейнере. По умолчанию ***flex-grow*** имеет значение 0, т.е. все **flex**-элементы будут того размера, который им задан с помощью ***width*** или ***flex-basis***, и часть ***flex***-контейнера останется незанятой

например, ***flex***-контейнер с шириной 800px содержит 4 ***flex***-элемента с шириной 150px, т.е. 200px остаются незаполненными. Если каждому из этих элементов назначить ***flex-grow: 1***, то 200px избыточного пространства нужно разделить на сумму этих коэффициентов, т.е. 200px:4 = 50px, а потом добавить результат к ширине каждого элемента. В результате этих действий ширина каждого ***flex***-элемента станет 200px.

если же какой-то один ***flex***-элемент получит свойство ***flex-grow: 2***, то расчет будет уже другим. 200px: 5 = 40px. Цифра 5 — это сумма ***flex-grow*** всех элементов (1+1+1+2= 5). Затем к 3-м из наших элементов нужно добавить по 40px, и их размер станет 190px, а к тому, у которого ***flex-grow: 2***, мы добавим 80px (40px\*2) — и получим 230px. Общая сумма всех ширин будет по-прежнему 800px.



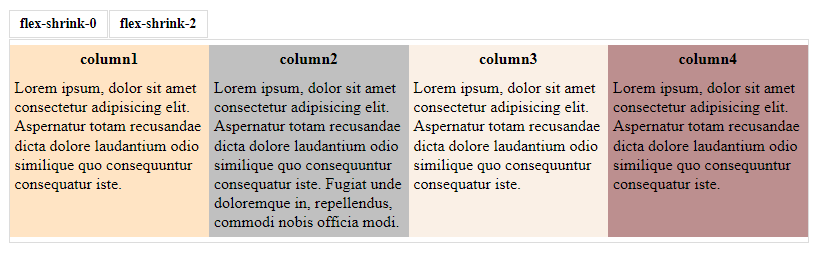
### Свойство flex-shrink:

является фактором (коэффициентом) уменьшения ширины (высоты в случае ***flexdirection: column***) ***flex***-элемента, если во ***flex***-контейнере недостаточно места. По умолчанию он равен 1, т.е. ширина всех элементов будет уменьшена, если ширина контейнера недостаточна для размещения всех элементов или для каждого них задан размер, который подразумевает большую ширину ***flex***-контейнера.

например, ширина ***flex***-контейнера равна 800px, а ширина каждого из 5 ***flex***-элементов 250px. Получится, что один из элементов должен выйти за пределы ***flex***-контейнера. Однако, этого не происходит из-за ***flex-shrink: 1***. Ширина каждого элемента уменьшится на 50px: 250px: 5 = 50px, и станет равна 200px: 250px-50px=200px.

если добавить какому-либо одному элементу ***flexshrink: 0***, то он перестанет уменьшаться, оставаясь размером 250px. А нехватка в 250px распределится уже между 4 элементами, т.е. 250px: 4 = 62,5px. Затем браузер отнимет эту величину от ширины каждого элемента: 250px–62.5px = 187.5px. Эта величина может быть изменена в зависимости от контента (текста, картинок) внутри ***flex***-элемента.

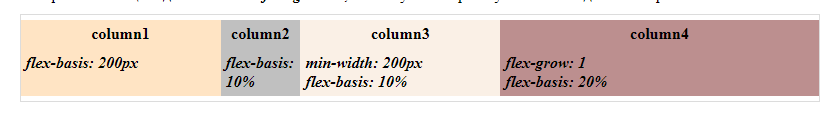
Если же у одного из элементов ***flex-shrink: 2***, то этот элемент станет уже, чем его соседи. Расчет сжимания будет несколько сложнее для восприятия, т.к. 250px:6 = 41.66px, т.к. цифра 6 берется путем складывания коэффициентов ***flex-shrink*** для всех ***flex***-элементов, т.е. 1+1+1+1+2 = 6.  
А затем от ширины каждого элемента браузер отнимет 41.66px: 250px – 41.66px = 208.34, а для элемента с ***flexshrink: 2*** уменьшение будет больше, а ширина — меньше: 250px – 41.66\*2 = 166.68px. Опять же — из-за размеров контента эта величина может изменяться.



### Свойство flex-basis:

Свойства ***flex-basis*** позволяет назначить базовый размер ***flex***-элемента по основной оси. Вы можете задать его в любых единицах (5em, 4rem, 48%, 200px и др.) или указать, как ***auto*** (значение по умолчанию), но реальный размер ***flex***-элемента будет зависеть от 2-х коэффициентов — ***flex-grow и flex-shrink***. Также реальная ширина flex-элемента будет зависеть от ***min-width*** и ***max-width***, назначенных для ***flex***-элемента. Значение ***flex-basis*** будет корректироваться в зависимости от минимальной и максимальной ширины, т.е. ширина элемента не может быть меньше ***min-width*** и не может быть больше ***maxwidth***.  
То же относится к размерам по вертикали ***(height, min-height, max-height)***, если свойство ***flex-direction*** имеет значение ***column***.

например, ширина ***flex***-контейнера составляет 800px. Первый столбец имеет ***flex-basis: 200px***, второй — ***flex-basis: 10%***, т.е. должен быть шириной 100px, однако текста в нем больше, поэтому реальная ширина его 149px. В 3-м столбце ***flex-basis: 10%***, но задано еще свойство ***min-width: 300px***, поэтому его ширина составляет 300px. Для четвертого столбца задано свойство ***flex-grow: 1***, поэтому его ширина увеличилась до 350.875px.



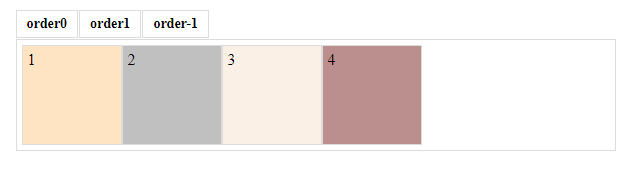
### Обобщенное свойство flex:

позволяет задать сразу 3 предыдущих значения через пробел в такой последовательности:  
**flex: flex-grow flex-shrink flex-basis;**

Значение по умолчанию: flex: 0 1 auto;  
Варианты написания:  
/\* Одно значение, число без размерности: flex-grow \*/  
**flex: 2;**  
/\* Одно значение, ширина/высота: flex-basis \*/  
**flex: 250px;**  
/\* Два значения: flex-grow | flex-basis \*/  
**flex: 1 30px;**  
/\* Два значения: flex-grow | flex-shrink \*/  
**flex: 2 2;**  
/\* Три значения: flex-grow | flex-shrink | flex-basis \*/  
**flex: 2 1 25%;**

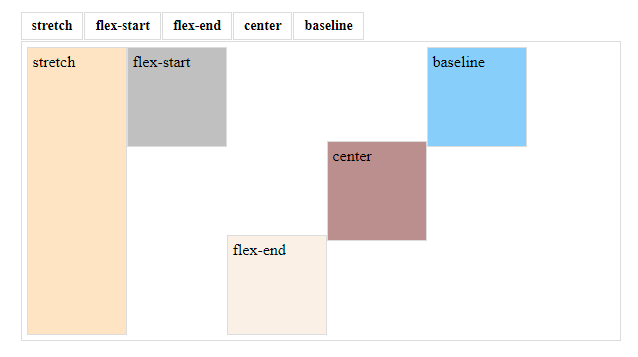
### Свойство order:

позволяет изменить последовательность расположения элементов. Это свойство определяет порядок следования ***flex***-элементов и назначается в виде целочисленного значения как положительного, так и отрицательного. По умолчанию ***order*** имеет значение 0, и элементы размещаются в той последовательности, в которой они расположены в ***html***-разметке. Если вы назначаете какому-либо элементу значение ***order: 1***, он переместится в конец, если ***order: -1*** — в начало.



### Свойство align-self:

меняет выравнивание отдельно взятого ***flex***-блока по поперечной оси по сравнению со свойством ***flex***-контейнера ***align-items.***



цвета:

.flex-item{

    width: 100%;

    min-height: 50px;

    border: 1px solid var(--color-gainsboro);

    padding: 5px;

}

.flex-item:nth-child(1){

    background: var(--color-bisque);

}

.flex-item:nth-child(2){

    background: var(--color-silver);

}

.flex-item:nth-child(3){

    background: var(--color-linen);

}

.flex-item:nth-child(4){

    background: var(--color-rosy-brown);

}

.flex-item:nth-child(5){

    background:#87CEFA;

}

.flex-item:nth-child(6){

    background: #DDA0DD;

}

.flex-item:nth-child(7){

    background: #F0E68C;

}

.flex-item:nth-child(8){

    background: #90EE90;

}

 <script>

        let items = document.querySelectorAll('.flex-nav');

        function example(selector){

            items.forEach(item =>{

                item.addEventListener('click', (e) => {

                    let attr = e.target.parentNode.getAttribute('data-name');

                    let text = e.target.innerText;

                    let parent = e.target.parentNode.parentNode;

                    let box = parent.querySelector('[data-box]');

                    if(box.hasAttribute('class')){

                        box.removeAttribute('class');

                    }

                    if(!box.classList.contains(text)){

                        box.classList.add(text);

                    }

                });

            });

        };

        example(items);

    </script>