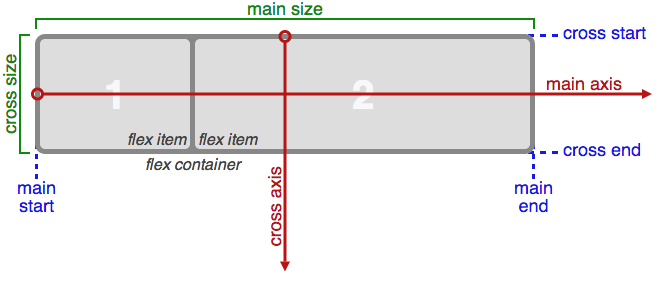
**FLEXBOX**

**main-axis** - главная ось, вдоль которой располагаются flex-элементы. Oна необязательно должна быть горизонтальной, всё зависит от свойства **justify-content**

**main-start** | **main-end** - flex-элементы размещаются в контейнере от позиции main-start до позиции main-end.

**main size** - ширина или высота flex-элемента в зависимости от выбранной основной величины. Основная величина может быть либо шириной, либо высотой элемента.

**cross axis** - поперечная ось, перпендикулярная к главной. Её направление зависит от направления главной оси.

**cross-start** | **cross-end** - flex-строки заполняются элементами и размещаются в контейнере от позиции cross-start и до позиции cross-end.

**cross size** - ширина или высота flex-элемента в зависимости от выбранной размерности равняется этой величине. Это свойство совпадает с width или height элемента в зависимости от выбранной размерности

**display: flex | inline-flex;**

Отображение как блок или как инлайн-эелемент

Имейте в виду:

* CSS-столбцы **columns** не работают с flex-контейнером
* **float, clear и vertical-align** не работают с flex-элементами

**flex-direction**

Устанавливает ось main-axis

* **row** (по умолчанию): слева направо для **ltr,** справа налево для **rtl;**
* **row-reverse:** справа налево для **ltr,** слева направо для **rtl;**
* **column:** аналогично **row,** сверху вниз;
* **column-reverse:** аналогично **row-reverse,** снизу вверх.

**flex-wrap:**

Контейнер будет однострочным или многострочным

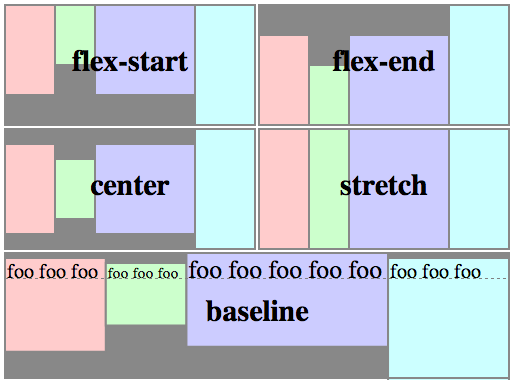
* **nowrap**(по умолчанию): однострочный / слева направо для ltr, справа налево для rtl;
* **wrap:** многострочный / слева направо для ltr, справа налево для rtl;
* **wrap-reverse:** многострочный / справа налево для ltr, слева направо для rtl.

**flex-flow: <'flex-direction'> || <'flex-wrap'>** /\* сокращение для свойств flex-direction и flex-wrap, вместе определяющих главную и поперечную оси. По умолчанию принимает значение row nowrap.\*/

**justify-content**

Определяет выравнивание относительно главной оси (горизонтальной по ум.)

* **flex-start** (по умолчанию): элементы сдвигаются к началу строки;
* **flex-end:** элементы сдвигаются к концу строки;
* **center:** элементы выравниваются по центру строки;
* **space-between**: элементы распределяются равномерно (первый элемент в начале строки, последний — в конце);
* **space-around:** элементы распределяются равномерно с равным расстоянием между собой и границами строки.

**align-items**

Позволяет выравнивать элементы в строке относительно друг друга по вторичной оси (вертикальной по ум.)

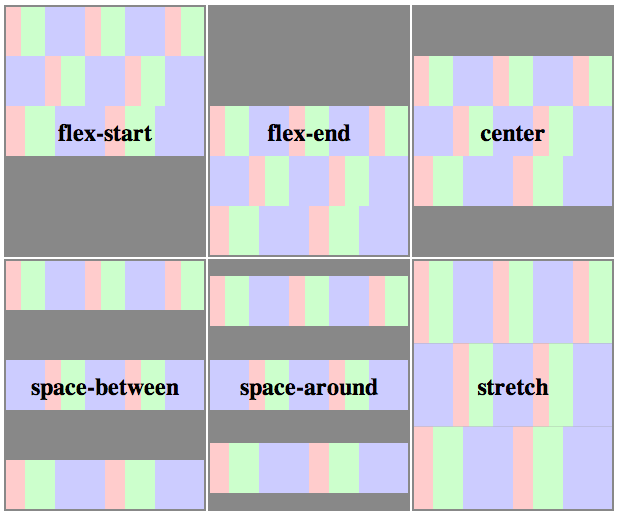
**flex-start:** граница cross-start для элементов располагается на позиции cross-start;

**flex-end:** граница cross-end для элементов располагается на позиции cross-end;

**center:** элементы выравниваются по центру поперечной оси;

**baseline:** элементы выравниваются по своей базовой линии;

**stretch** (по умолчанию): элементы растягиваютcя, заполняя контейнер (с учётом min-width/max-width).

**align-content**

Отвечает за выравнивание целых строк относительно гибкого контейнера. Оно не будет давать эффекта если гибкие элементы занимают одну строку.

**flex-start:** строки выравниваются относительно начала контейнера;

**flex-end:** строки выравниваются относительно конца контейнера;

**center:** строки выравниваются по центру контейнера;

**space-between:** строки распределяются равномерно (первая строка в начале строки, последняя — в конце);

**space-around:** строки распределяются равномерно с равным расстоянием между собой;

**stretch**(по умолчанию): строки растягиваются, заполняя свободное пространство.

**Применяется к дочерным элементам FLEX**

**order: <integer>;** /\*управляет порядком расположения элементов\*/

**flex-grow: <number> (по умолчанию 0) /\*** Определяет, какую долю свободного места внутри контейнера элемент может занять. Если у всех элементов свойство flex-grow задано как 1, то каждый потомок получит внутри контейнера одинаковый размер. Если вы задали одному из потомков значение 2, то он заберёт в два раза больше места, чем другие.\*/

**flex-shrink: <number> (default 1) –** элемент занимает меньше места, как со свойством grow

**flex-basis: <length> | auto (default auto)**  - Определяет размер по умолчанию для элемента перед распределением пространства в контейнере.

**flex: none | [ <'flex-grow'> <'flex-shrink'>? || <'flex-basis'> ]** сокращение для flex-grow, flex-shrink и flex-basis. Второй и третий параметры (flex-shrink, flex-basis) необязательны. Значение по умолчанию — 0 1 auto.

**align-self: auto | flex-start | flex-end | center | baseline | stretch** Позволяет переопределить выравнивание, заданное по умолчанию или в align-items, для отдельных flex-элементов.

**НОВОЕ В CSS**

**calc() -** позволяет совершать вычисления прямо в коде

**left: calc(50% - 25px);**

**vh, vw** - viewport height, viewport width;

**width: 25vw** //25% от ширины экрана устройства

**position: sticky:**  похож на позицию: position: fixed. Разница в том что элемент: position: sticky ведет себя как: position: relative относительно в пределах своего родителя, пока данное смещение порогового значения в окне просмотра

**box-reflect: below, right, left, above. -** отражает элемент, как в зеркале

-webkit-box-reflect: below 0px -webkit-gradient(linear, left top, left bottom, from(transparent), to(rgba(250, 250, 250, 0.1))); //изображение плавно становится невидимым

**Multicolumn** - разбитие конетнта на колонки. Работает почти везде (IE10+) с префиксами

**pointer-events: none; -** позволяет средствами CSS отменить евенты на элементе (типа клика или hover), вместо этого, евент происходит на чем угодно, что лежит за элементом

**text-overflow: ellipsis; -** выводит многоточие, если обьем текста превышает обьем контейнера

**toggle()** позволяет элементам-потомкам зациклиться на списке значений вместо того, что бы наследовать одно и тоже значение:

**ul { list-style-type: disc; }**

**ul ul {**

**list-style-type: toggle(disc, circle, square, box);**

**}**

**filter: -** позволяет накладывать фильтры на изображения. Использовать с префиксами. Можно указывать даже несколько эффектов сразу:

filter: blur(20px) grayscale(20%);

**blur() - размытие по Гауссу.** Значения принимаются только в длинне, без процента

filter: blur(5px);

**brightness() - яркость.** 100% - неизмененное изображение

filter: brightness (3);

**contrast() - контраст.** 100% - полностью темное изображение

filter: contrast(4);

**grayscale() - перевод в ч\б.** Указывается в процентах или в числовом значении

filter: grayscale(20%);

**hue-rotate() - изменения оттенка.** Указывается в градусах, от 0 до 360

filter: hue-rotate(90deg);

**invert() - инвертирует цвета.** 0% - неизмененное изображение

filter: invert(.9);

**opacity() - похоже на обычное свойство opacity, но выглядит лучше.** В процентах

filter: opacity(.2);

**saturate() - насыщенность.** Указывается в процентах или в числовом значении

filter: saturate(8);

**sepia() - эффект серпии.** Указывается в процентах или в числовом значении

filter: sepia(20%);

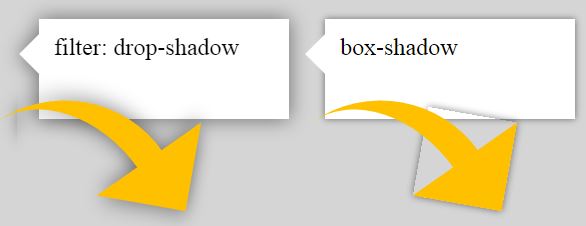
**url() - for applying SVG filters.** Tutorial <https://css-tricks.com/gooey-effect/> - прикольная размножающаяся меню

**drop-shadow() - похоже на box-shadow.** В отличии от него, фильтр так же отеняет вложенный рисунок внутри, а не сам контейнер

HTML  
**<div class="tri drop-shadow">filter: drop-shadow</div>  
<div class="tri box-shadow">box-shadow</div>  
<div style="clear:both"></div>  
<div class="flecha drop-shadow"></div>  
<div class="flecha box-shadow"></div>**

CSS

**.drop-shadow { filter: drop-shadow(0px 0px 10px rgba(0,0,0,.5));}  
.box-shadow { box-shadow: 0px 0px 10px rgba(0,0,0,.5) }**



Так же, фильты можно использовать вместе с **анимацией**

**img.animated { -webkit-animation: filter-animation 5s infinite; }  
  
@-webkit-keyframes filter-animation {  
 0% { -webkit-filter: sepia(0) saturate(2); }  
 50% { -webkit-filter: sepia(1) saturate(8); }  
 100% { -webkit-filter: sepia(0) saturate(2); }  
}**

**НОВОЕ В HTML5**

**<output>** - Определяет область в которую выводится информация, преимущественно с помощью скриптов.

**<progress**> Используется для отображения прогресса завершённости задачи через JS

**<samp>** используется для отображения текста, который является результатом вывода компьютерной программы или скрипта.