



Шпаргалка по всем свойствам CSS Flexbox

В этой статье мы рассмотрим основы CSS Flexbox, чтобы вы могли создавать адаптивные сайты.

Объясним, как работает каждое из свойств Flexbox, и дадим вам шпаргалку к каждому свойству. В шпаргалках описано всё, что можно сделать с помощью Flexbox.

Это — перевод [оригинальной статьи на freeCodeCamp](#) от автора Joy Shaheb.

Что такое Flexbox

Когда вы строите дом, вам нужен чертеж. Точно такой же чертеж нам понадобится для создания веб-сайтов. И Flexbox — это тот самый чертеж.

Flexbox позволяет нам компоновать между собой содержимое нашего сайта. Ещё она помогает создавать адаптивные структуры для веб-сайтов под различные устройства.

Вот демо-визитка, написанная на Flexbox в качестве чертежа.

Miya Ruma

[Home](#) [Portfolio](#) [Contact](#)

Hello 🖐️

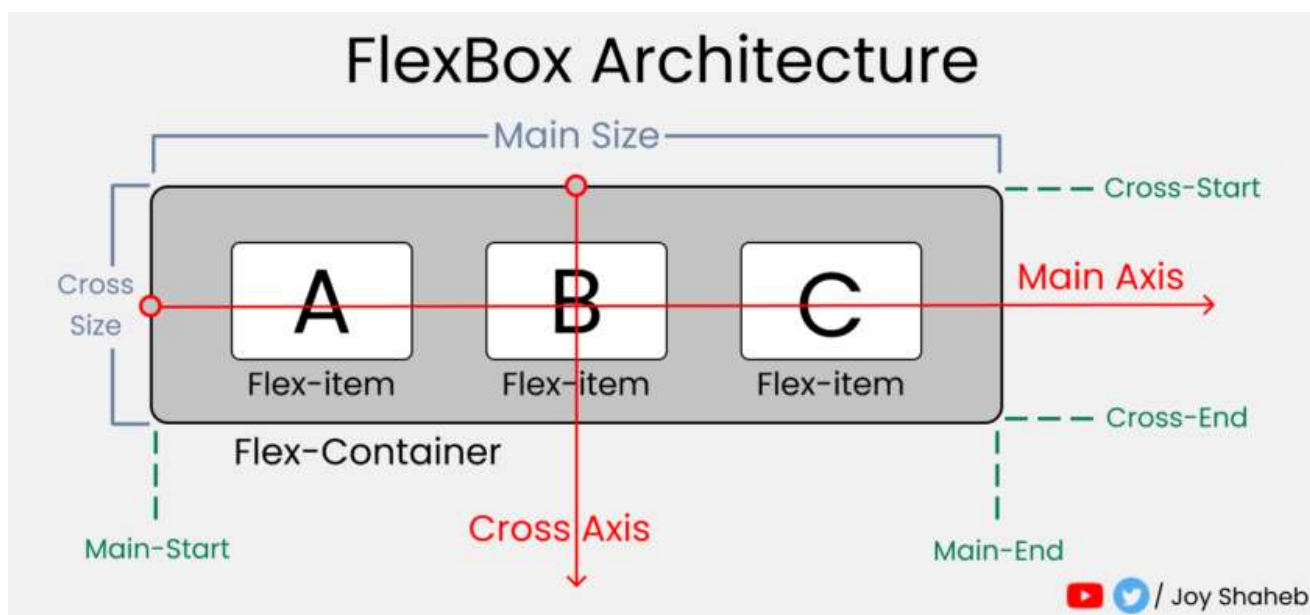
I'm **Miya Ruma**
A Designer From
Tokyo, Japan



Архитектура Flexbox

Как же работает архитектура Flexbox? Контентные flex-элементы распределяются вдоль главной оси и поперечной оси, как в системе координат.

И, в зависимости от значений flex-direction, положение блоков меняется между строками и колонками.




Список свойств Flexbox

Этот список содержит все возможные свойства и значения, которые вы можете использовать при работе с Flexbox.

Вы можете использовать список при выполнении проектов и экспериментировать с различными значениями.

Flex Box Chart	
Property	Value(s)
#1 display	flex
#2 flex-direction	row column column-reverse row-reverse
#3 justify-content	flex-start flex-end center space-between space-around space-evenly
#4 align-items	flex-start flex-end center stretch baseline
#5 align-content	flex-start flex-end center stretch space-between space-around
#6 align-self	auto flex-start flex-end center baseline stretch
#7 Order	/* Any positive Value */
#8 flex-grow	/* Any positive Value */
#9 flex-shrink	/* Any positive Value */
#10 flex-wrap	nowrap wrap wrap-reverse

  / Joy Shaheb

Создаём проект

Чтобы создать проект с Flexbox, нужно немного знать HTML, CSS и работу с VS-кодом. Вот, что вам нужно сделать:

1. Создайте папку с именем «Project-1» и откройте VS Code.
2. Создайте файлы index.html и style.css.
3. Установите Live Server и запустите его.

Или вы можете просто открыть [Codepen](#) и начать кодить.

HTML

В HTML напишите эти строки кода внутри тега body



```
<div class="container">
  <div class="box-1"> A </div>
  <div class="box-2"> B </div>
  <div class="box-3"> C </div>
</div>
```

CSS

Создайте класс `.container` и укажите в нём поля. Затем оформите блоки так, чтобы все они выглядели одинаково, как в примере ниже. ????

Примечание: не забудьте указать высоту контейнера.



```
.container{
  height : 100vh;
}

[class ^="box-"]{
  width: 140px;
  height: 140px;
  background-color: skyblue;
  border: 2px solid black;

  // To view the letter better
  font-size: 65px;
}
```

Также, прежде чем начать, необходимо понять отношения между родительскими и дочерними классами.

Flexbox работает только с родительским классом, но не с дочерними классами.

Здесь класс `.container` является родительским, а классы `.box-*` — дочерними.

Поэтому примените `display: flex` внутри класса `.container`. Расположите текст по центру блока.



```
.container{
  display : flex;
  height : 100vh;

  // To place some gap between boxes
  gap : 25px;
}
[class ^="box-"]{
  // Code from previous step are here

  // Placing text at center
  display : flex;
  justify-content : center;
  align-items : center;
}
```

И... все готово! Теперь мы готовы кодить. ????

Свойство flex-direction

Это свойство позволяет нам задавать направление и ориентацию, согласно которым flex-элементы должны располагаться внутри flex-контейнера.

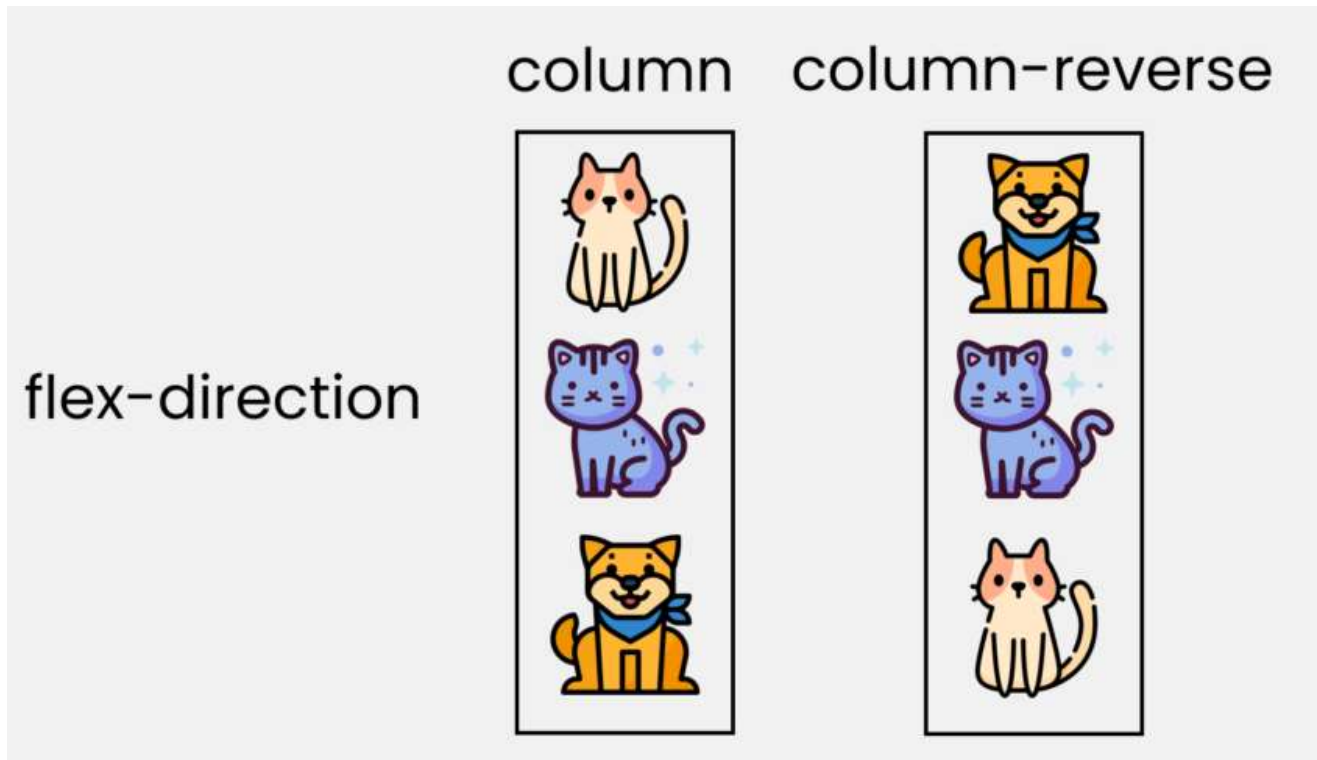
flex-direction :

row



row-reverse





Чтобы у нас получилось то же самое, что и на картинке с котиками, нужно написать такой CSS-код.

Обратите внимание, что мы пишем код внутри класса `.container`.

```
.container{  
  //code from setup stage are here  
  
  // Change the value  ???? here to see results  
  flex-direction : row;  
}
```

Свойство `justify-content`

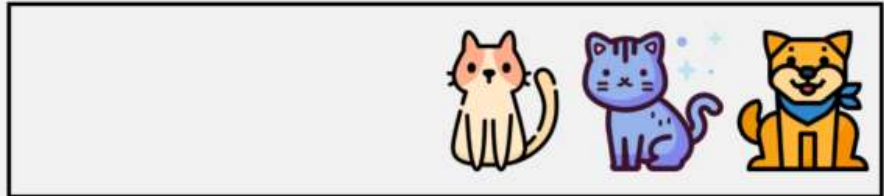
Это свойство выстраивает flex-элементы вдоль **главной оси** внутри flex-контейнера.

justify-content

flex-start



flex-end

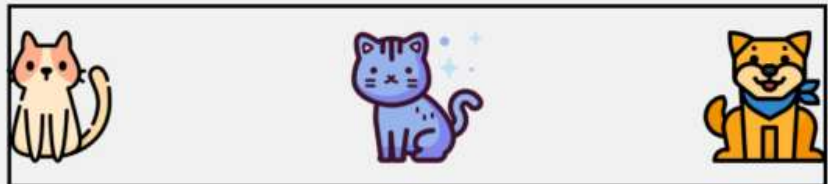


center



justify-content

space-between



space-around



Half Equal Equal Half

space-evenly



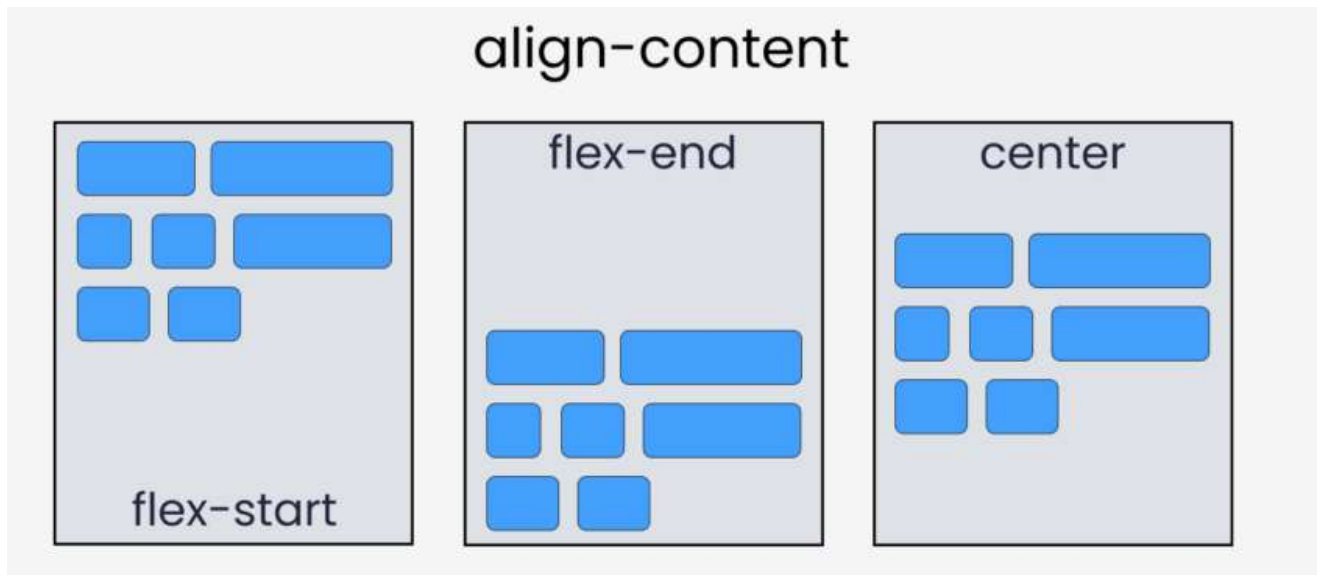
Equal Equal Equal Equal

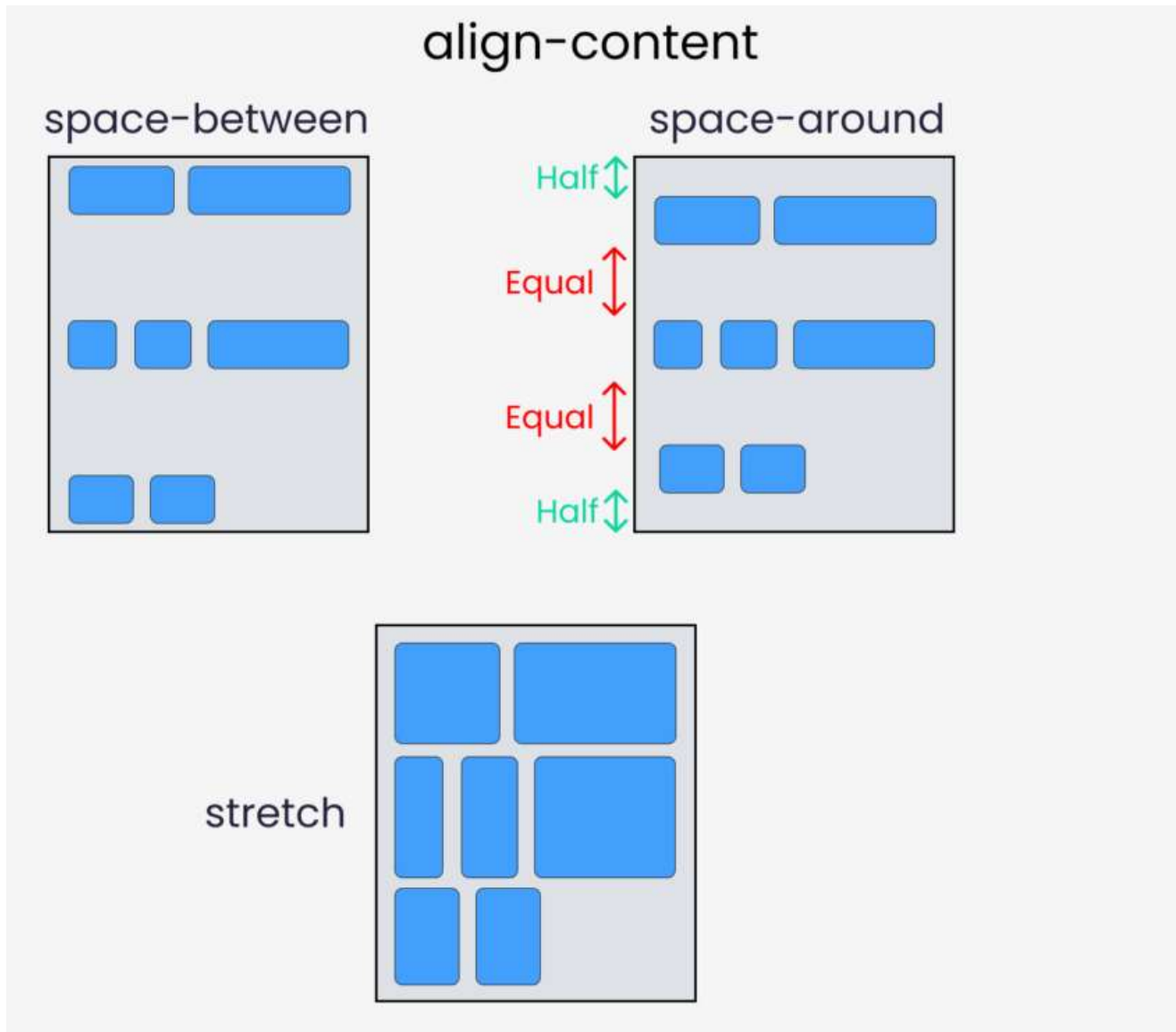
Попробуйте воссоздать это с помощью этого кода CSS:

```
.container{  
  //code from setup stage are here  
  
  //  Change the value  ??? here to see results  
  justify-content: flex-start;  
}
```

Свойство align-content

Это свойство выравнивает flex-элементы по **поперечной оси** внутри flex-контейнера. Оно аналогично свойству justify-content, которое мы рассматривали только что. Разница только в осях выравнивания.





Обратите внимание, что без `flex-wrap` это свойство не работает. Вот, что должно быть в вашем коде:

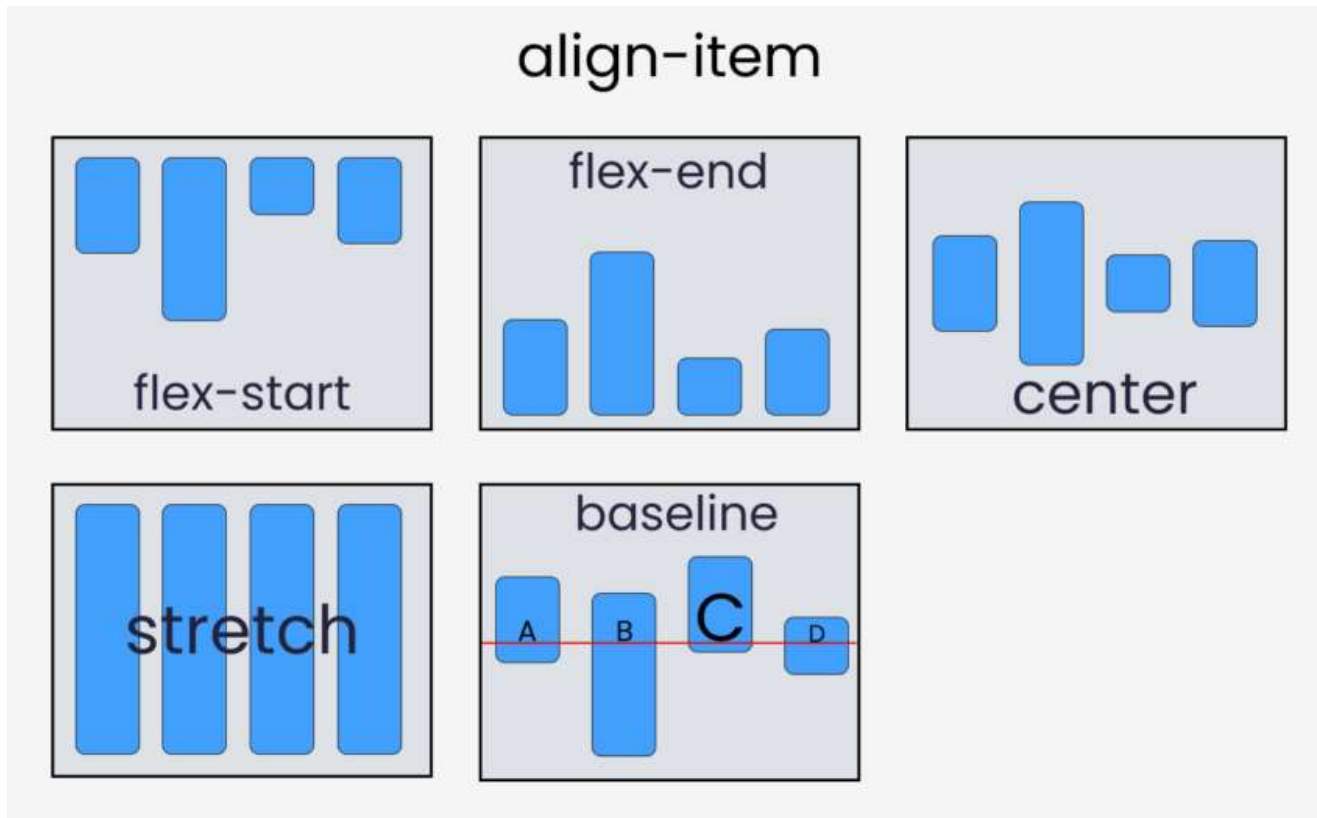
```
.container{

//  Change the value  ??? here to see results
  align-content: center;

// without this line, align-content won't work
  flex-wrap: wrap;
}
```

Свойство align-items

Это свойство распределяет Flex-элементы вдоль поперечной оси.

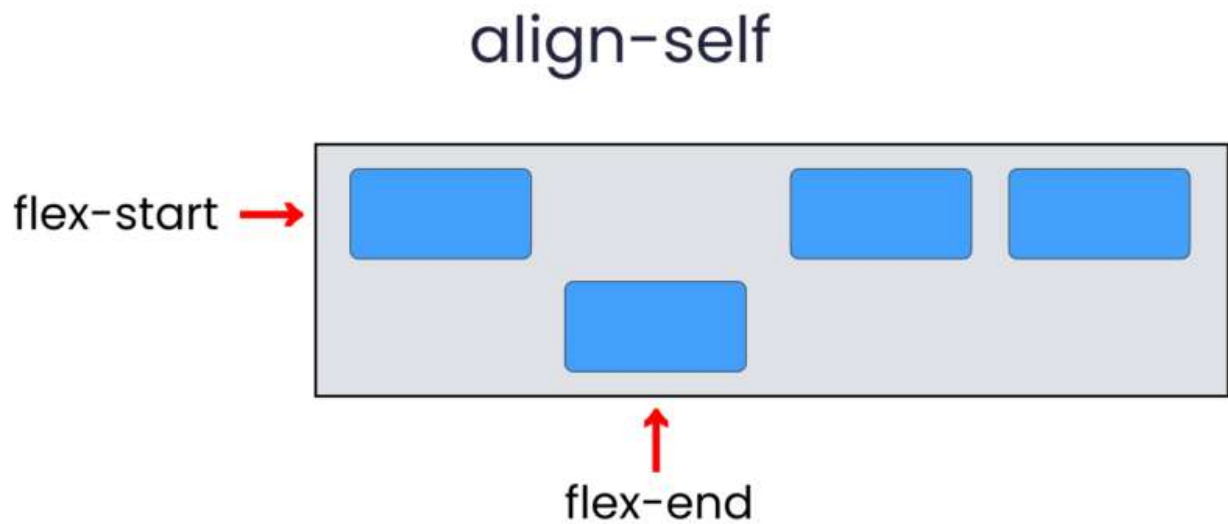


Чтобы протестировать свойство, напомним следующий код в CSS:

```
.container{  
  //code from setup stage are here  
  
  // Change the value ???? here to see results  
  align-items: flex-end;  
}
```

Свойство align-self

Это свойство работает с дочерними классами. Оно позиционирует выбранный элемент **вдоль поперечной оси**.



Всего есть 6 значений `align-self`:

- `flex-start`;
- `flex-end`;
- `center`;
- `baseline`;
- `stretch`;
- `auto`.

Чтобы воссоздать результат, выберите любой `.box-*` и напишите следующий код:

```
.box-2{  
  // Change the value ???? here to see results  
  align-self : center;  
}
```

Свойства `flex-grow`, `shrink`, `wrap` и `basis`

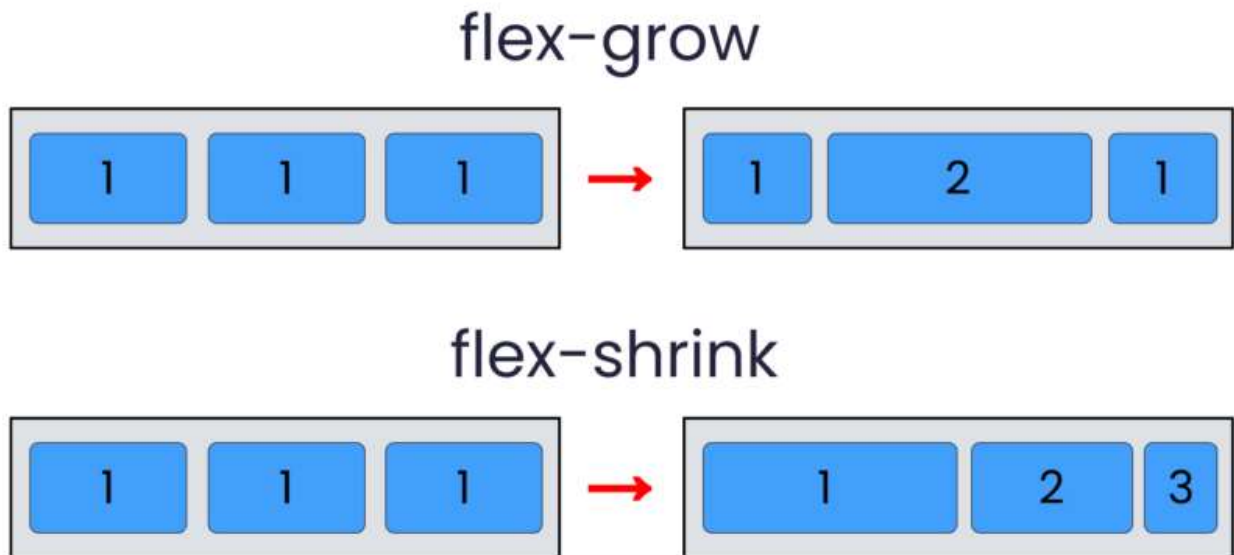
Свойства, которые мы сейчас обсудим, будут работать при изменении размера окна. Давайте приступим.

Flex-grow

Это свойство **увеличивает** размер flex-элемента на основе ширины flex-контейнера.

Flex-shrink

Это свойство **уменьшает** flex-элемент в зависимости от ширины flex-контейнера. Оно противоположно `flex-grow`.



Обратите внимание, что `flex-grow` и `flex-shrink` работают на дочерних классах. Поэтому мы изменим все наши блоки следующим образом:

```
.box-1{
  flex-grow: 1;
}
.box-2{
  flex-grow: 5;
}
.box-1{
  flex-grow: 1;
}
```

Поиграйтесь с размером окна браузера, и вы увидите результат.

Теперь используем `flex-shrink`.

Обратите внимание, что сначала нужно удалить свойство `flex-wrap`, иначе `flex-shrink` не будет работать.



```
.box-1{  
    flex-shrink: 1;  
}  
.box-2{  
    flex-shrink: 5;  
}  
.box-1{  
    flex-shrink: 1;  
}
```

Измените размер окна, и вы увидите результат.

Flex-wrap

Это свойство помогает установить количество flex-элементов в строке или в колонке.

flex-wrap

no-wrap



wrap



wrap-reverse



Свойство работает в родительском классе `.container`. Итак, напишите код:

```
.container{
  //other codes are here

  // Change value ???? here to see results
  flex-wrap : wrap;
```

Flex-basis

Свойство похоже на добавление ширины к flex-элементу, но работает более гибко.

К примеру, `flex-basis: 10em` установит начальный размер flex-элемента равным `10em`. Его конечный размер будет основан на доступном пространстве, на данных `flex-grow` и `flex-shrink`.

Как читать сокращения свойств Flexbox

Сокращение flex

Вот, как могут быть одновременно указаны свойства `flex-grow`, `flex-shrink` и `flex-basis` вместе взятые.

The diagram illustrates the components of the `flex` shorthand property. It shows the shorthand `flex : 2 1 30em ;` with three red arrows pointing to its parts: the first value '2' is linked to `flex-grow`, the second value '1' is linked to `flex-shrink`, and the unit '30em' is linked to `flex-basis`.

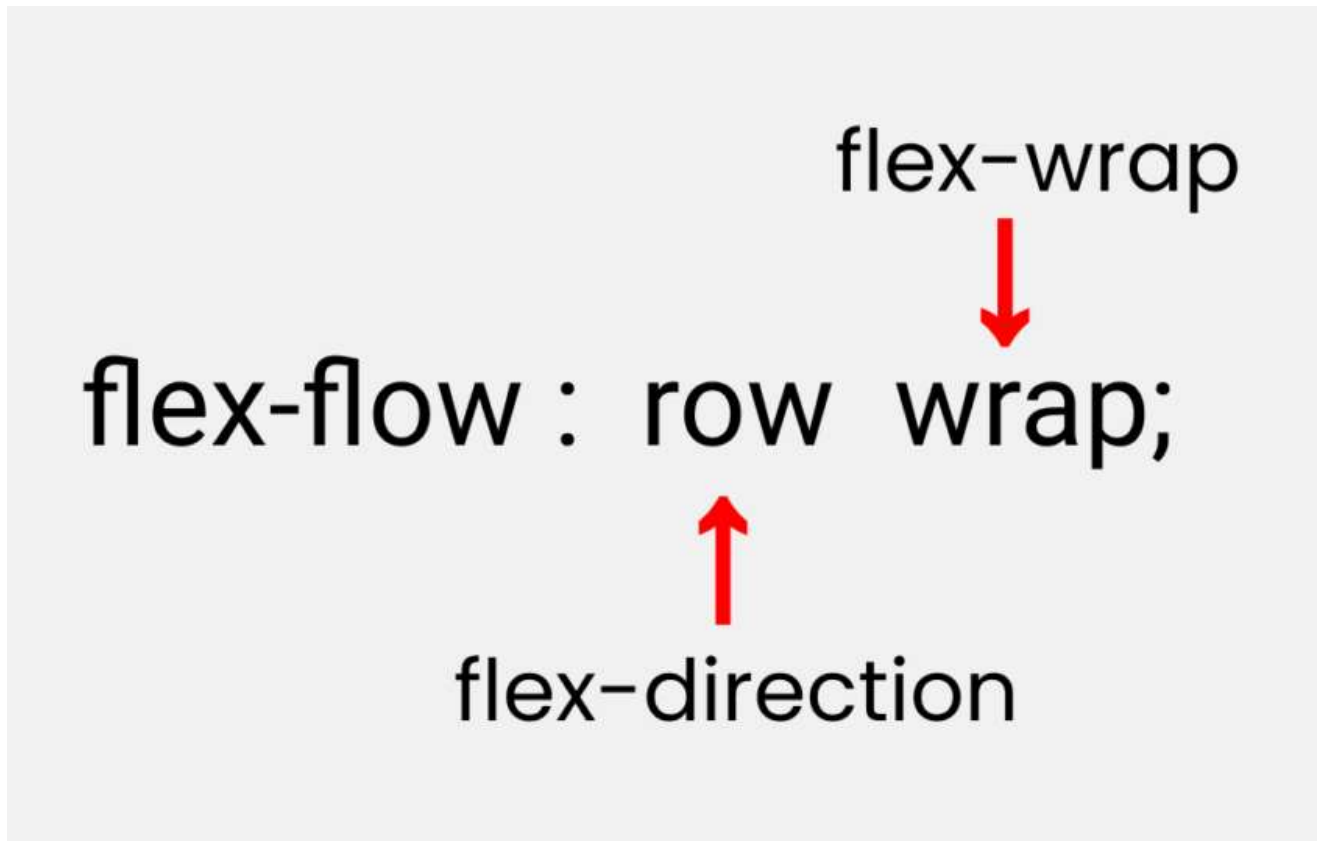
Попробуйте понять сокращение, написав следующий код.

Обратите внимание, что он работает только для дочерних классов.

```
.box-2{  
  flex : 2 1 30em;  
}
```

Сокращение flex-flow

Это сокращение для свойств `flex-direction` и `flex-wrap`.



Напишите следующий код, который работает в родительском классе:

```
.container{  
  flex-flow : row wrap;  
}
```

Сокращение place-content

Это сокращение для свойств `justify-content` и `align-content`.

Обратите внимание, что оно работает в родительском классе.

```
.container{  
  place-content : center flex-end;  
}
```

Если вы дочитали до конца, вам полагается большущая медаль. <3