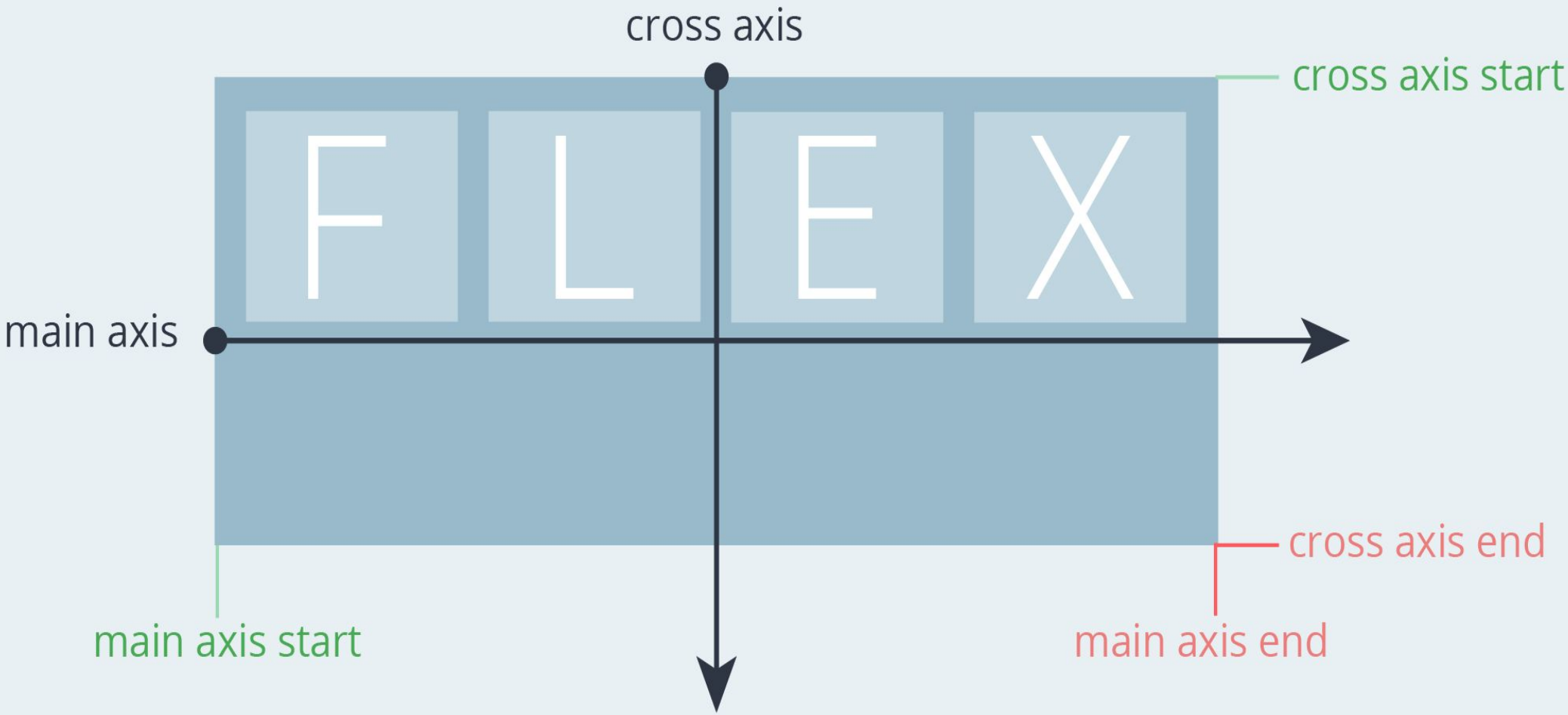

cross axis
cross axis start
F L E X
main axis
cross axis end
main axis start
main axis end

Вступление

На протяжении долгого времени использовали таблицы, float-элементы, inline-block и другие CSS свойства, чтобы придать блокам нужное расположение. Простые вещи, как вертикальное центрирование, осуществлялись достаточно сложно. Создание же макета на основе жидких сеток вообще считается верхом профессионализма, вот почему широкое распространение получили CSS-фреймворки на основе сеток.

Решение всех этих проблем является Flexbox!

Что такое flexbox?

Flexbox призвана кардинально изменить ситуацию в лучшую сторону при решении огромного количества задач. Flexbox позволяет контролировать размер, порядок и выравнивание элементов по нескольким осям, распределение свободного места между элементами и многое другое.

Преимущества flexbox

Все блоки очень легко делаются “резиновым”, что уже следует из названия “flex”. Элементы могут сжиматься и растягиваться по заданным правилам, занимая нужное пространство.

Выравнивание по вертикали и горизонтали, базовой линии текста работает шикарно.

Расположение элементов в html не имеет решающего значения. Его можно поменять в CSS. Это особенно важно для некоторых аспектов responsive верстки.

Элементы могут автоматически выстраиваться в несколько строк/столбцов, занимая все предоставленное место.

Множество языков в мире используют написание справа налево rtl (right-to-left), в отличии от привычного нам ltr (left-to-right). Flexbox адаптирован для этого. В нем есть понятие начала и конца, а не права и лева. Т.е. в браузерах с локалью rtl все элементы будут автоматически расположены в реверсном порядке.

Синтаксис CSS Flexbox очень прост и осваивается довольно быстро.

Все эти чудеса верстки возможны только в последних версиях браузеров, потому не всегда можно будет применять флекс в

Поддержка браузерами последней спецификации flexbox:

Chrome 29+

Firefox 28+

Internet Explorer 11+

Opera 17+

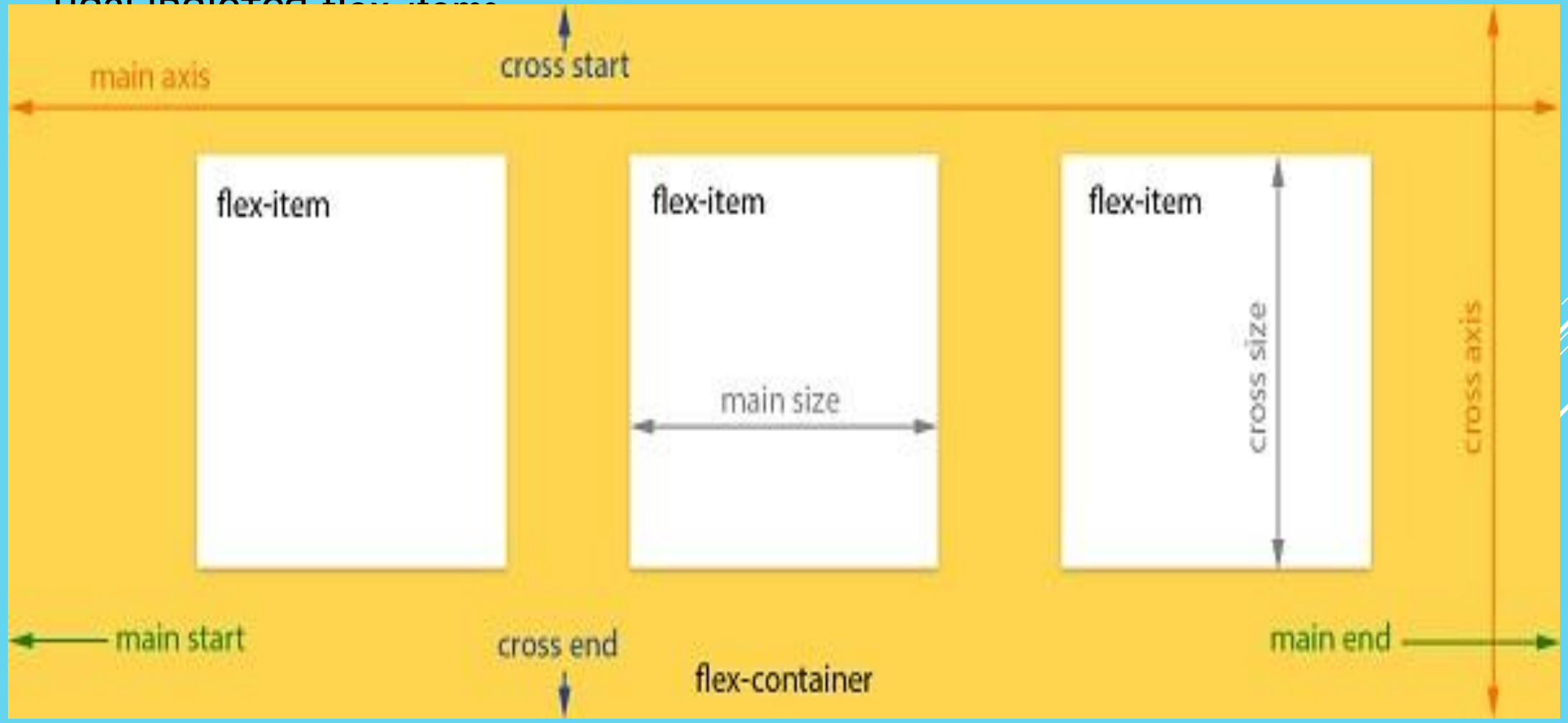
Safari 6.1+ (с префиксом -webkit-)

Android 4.4+

IOS 7.1+ (с префиксом -webkit-)

Flex-макет. Главная и поперечная ось

Flex-макет состоит из родительского контейнера, указанного как flex-container, и его дочерних элементов, которые называются flex-item.



main-axis - главная ось, вдоль которой располагаются flex-элементы. Обратите внимание, она необязательно должна быть горизонтальной, всё зависит от свойства justify-content(см. ниже).

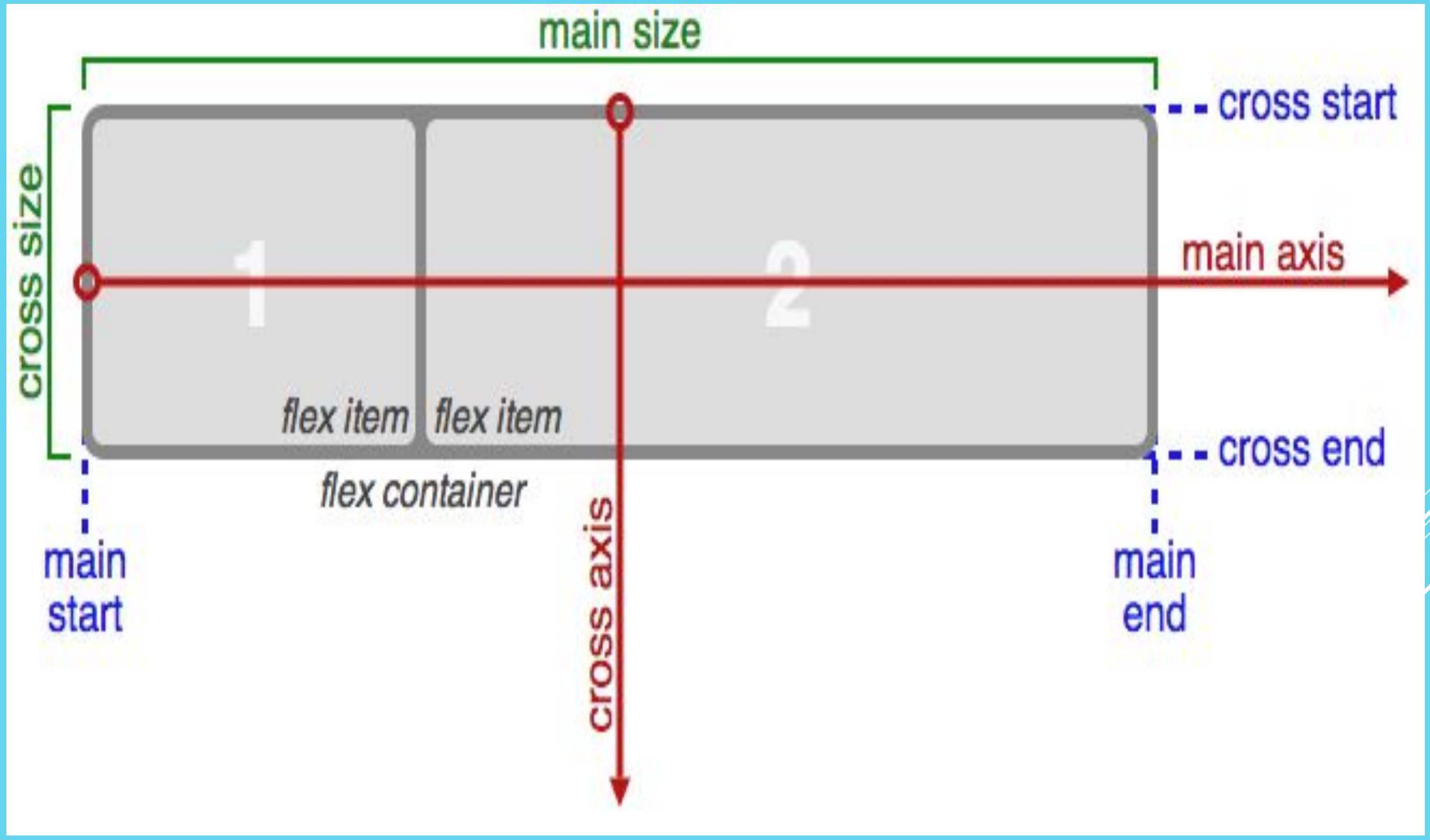
main-start main-end - flex-элементы размещаются в контейнере от позиции main-start до позиции main-end.

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


main size
cross start
cross size
main axis
1
2
flex item
flex item
flex container
cross end
cross axis
main
start
main
end

Если обычный layout основывается на направлениях потоков блочных и инлайн-элементов, то flex- layout основывается на осях

Главной осью flex-контейнера является направление, в соответствии с которым располагаются все его дочерние элементы.

Поперечной осью называется направление, перпендикулярное главной оси.

Главная ось в ltr локали по умолчанию располагается слева направо. Поперечная – сверху вниз. Направление главной оси flex-контейнера можно задавать, используя базовое css свойство flex-direction. Поэтому в основном элементы будут распределяться либо вдоль главной оси (от main-start до main-end), либо вдоль поперечной оси (от cross-start до cross-end).

Основные свойства flex-контейнера

Так же следует иметь в виду, что при использовании Flexbox для внутренних блоков не работают float, clear и vertical-align, а так же свойства, задающие колонки в тексте.

Display: flex

Чтобы использовать макет flexbox нужно просто установить свойство display для родительского HTML-элемента:

display: flex; отображает контейнер как блочный элемент, при рассчете ширины блоков приоритет у раскладки (при недостаточной ширине блоков контент может вылезать за границы);

display: inline-flex; отображает контейнер как строчный элемент, приоритет у содержимого (контент растопыривает блоки до необходимой ширины, чтобы строчки, по возможности, поместились).

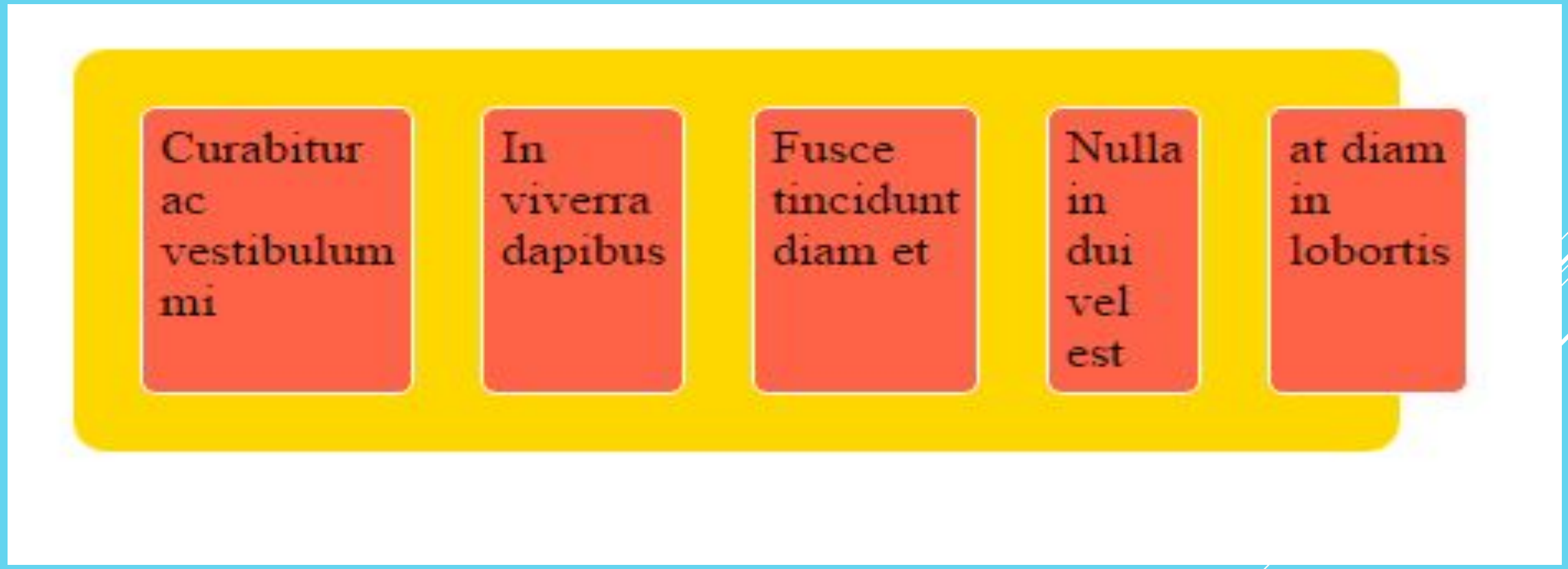
Теперь подготовим наше место для экспериментов !

```
<body>
  <div class="flex-container">
    <div class="flex-item">Curabitur ac vestibulum mi</div>
    <div class="flex-item">   In viverra dapibus </div>
    <div class="flex-item">  Fusce tincidunt diam et </div>
    <div class="flex-item">  Nulla in dui vel est  </div>
    <div class="flex-item"> at diam in lobortis  </div>
  </div>
</body>
```

```
body {
  padding: 20px;
  background: white;
  }
.flex-container {
  padding: 10px;
  background: gold;
  border-radius: 10px;
  }
.flex-item {
  margin: 10px;
  padding: 5px;
  background: tomato;
  border-radius: 5px;
  border: 1px solid #FFF;
  }
```

- Curabitur ac vestibulum mi
- In viverra dapibus
- Fusce tincidunt diam et
- Nulla in dui vel est
- at diam in lobortis

Родителю - flex-container, добавляем свойство display: flex; и у нас должен получиться вот такой результат :

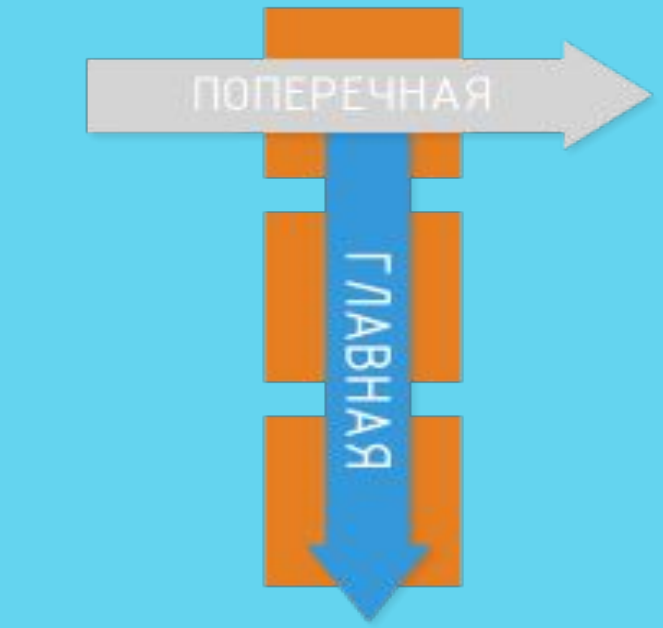


В случае добавления display: inline-flex; мы получим :

После установки данных значений свойства каждый дочерний элемент автоматически становится flex-элементом, выстраиваясь в ряд (вдоль главной оси) колонками одинаковой высоты, равной высоте блока-контейнера. При этом блочные и строчные дочерние элементы ведут себя одинаково, т.е. ширина блоков равна ширине их содержимого с учетом внутренних полей и рамок элемента.

Flex-direction

Направление раскладки блоков управляется свойством flex-direction. Блоки могут быть установлены в двух основных направлениях, как строки по горизонтали или как колонки по вертикали.

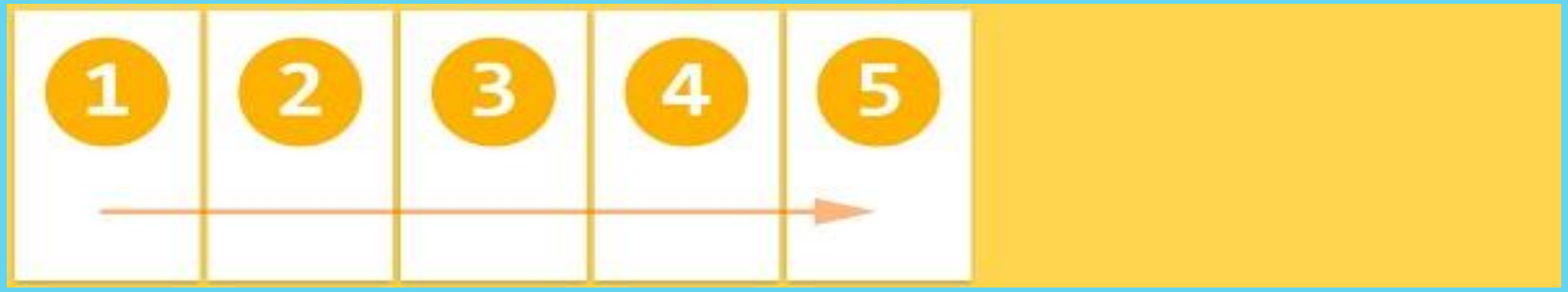


flex-direction: column

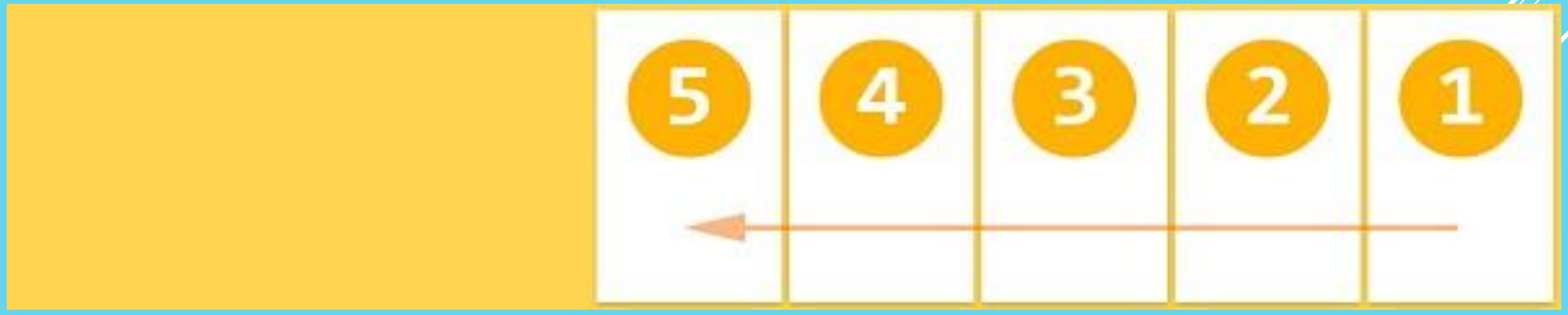
flex-direction: row

Доступные значения flex-direction:

flex-direction:row; (значение по умолчанию) : слева направо (в rtl справа налево)

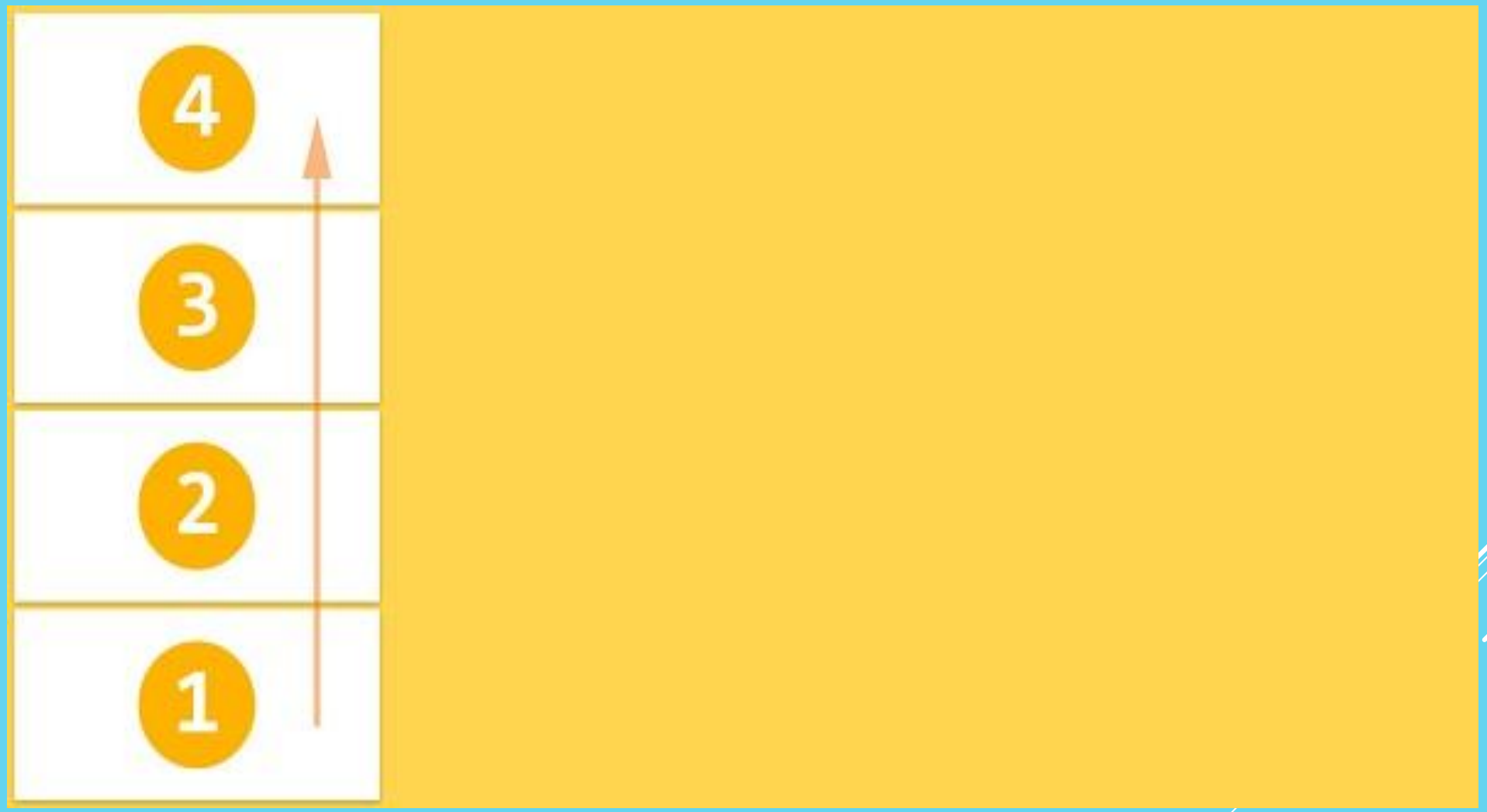


flex-direction: row-reverse;справа налево (в rtl слева направо)



flex-direction: column; сверху вниз

flex-direction: column-reverse; снизу вверх

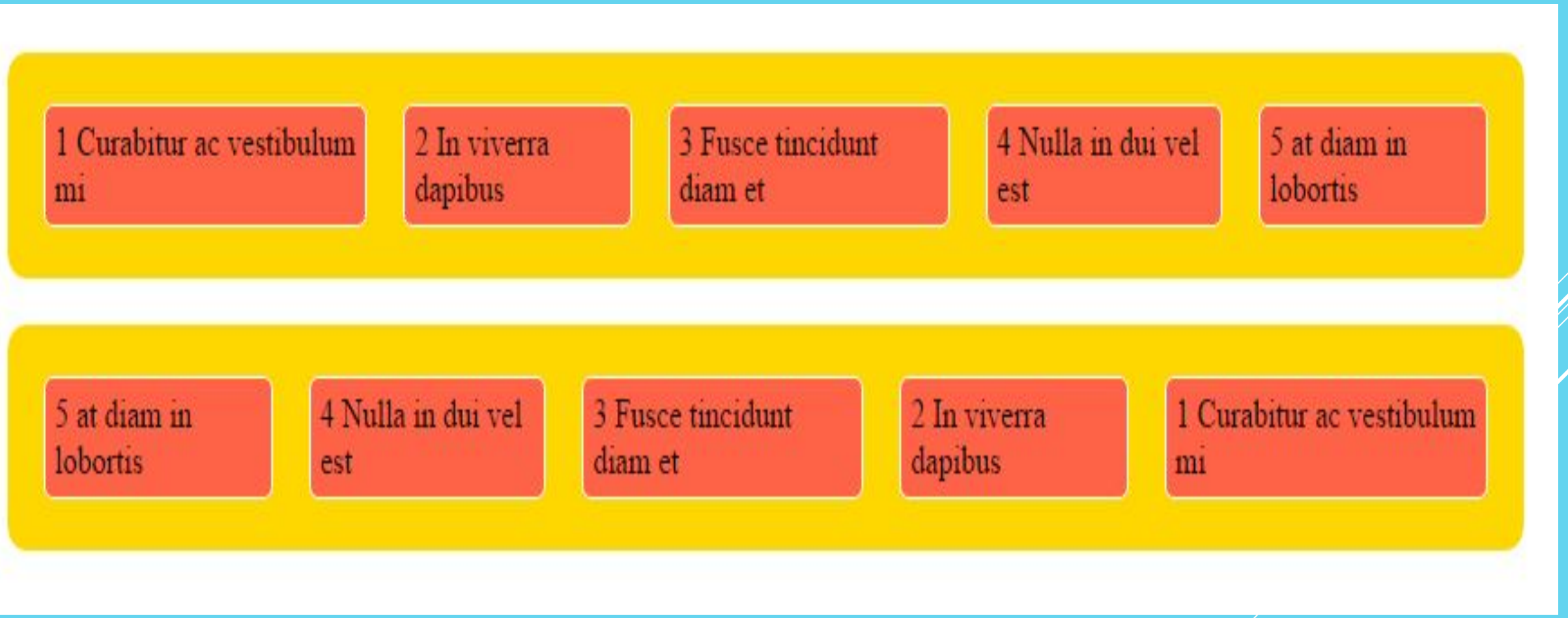


Для сравнения свойств добавим такой же блок и добавим сами свойства

```
<div class="flex-container row-reverse">
   <div class="flex-item"> 1 Curabitur ac vestibulum mi    </div>
   <div class="flex-item">2   In viverra dapibus   </div>
   <div class="flex-item">3  Fusce tincidunt diam et  </div>
   <div class="flex-item">4  Nulla in dui vel est  </div>
   <div class="flex-item">5 at diam in lobortis  </div>
 </div>
```

```
.flex-container {  display: flex;  flex-direction: row; width:40%; }
.row-reverse {  flex-direction: row-reverse;  }
```

Результат:


1 Curabitur ac vestibulum mi
2 In viverra dapibus
3 Fusce tincidunt diam et
4 Nulla in dui vel est
5 at diam in lobortis
5 at diam in lobortis
4 Nulla in dui vel est
3 Fusce tincidunt diam et
2 In viverra dapibus
1 Curabitur ac vestibulum mi

Теперь заменим родителям (flex-container) . свойства и для второго родителя заменим класс на column-reverse

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

1 Curabitur ac vestibulum mi

2 In viverra dapibus

3 Fusce tincidunt diam et

4 Nulla in dui vel est

5 at diam in lobortis

5 at diam in lobortis

4 Nulla in dui vel est

3 Fusce tincidunt diam et

2 In viverra dapibus

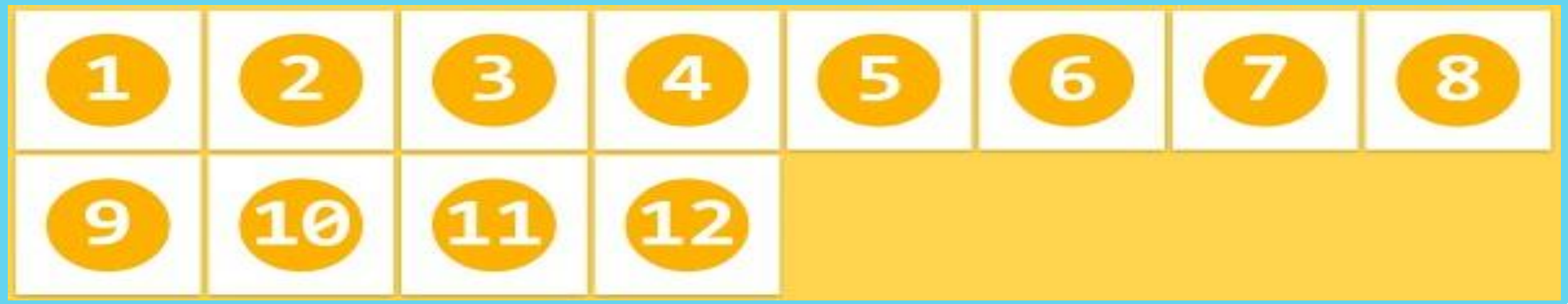
1 Curabitur ac vestibulum mi

Flex-wrap

В одной строке может быть много блоков. Переносятся они или нет определяет свойство flex-wrap.

flex-wrap: nowrap; (по умолчанию) элементы располагаются в одну строку, по-умолчанию они сжимаются, чтобы занять всю ширину flex-контейнера.

flex-wrap:wrap; Flex-элементы располагаются в несколько строк, если это необходимо, слева-направо и свехру-вниз.



flex-wrap: wrap-reverse; Flex-элементы располагаются в несколько строк, если это необходимо, слева-направо и снизу-вверх.

Снова заменим родителям (flex-container) . свойства и для второго, третьего родителя заменим классы

```
.flex-container {
display: flex;
flex-wrap: nowrap;
}
.wrap { flex-wrap: wrap;}
.wrap-reverse { flex-wrap: wrap-reverse;}
```

```
<body>
  <div class="flex-container">
    <div class="flex-item">1 Curabitur ac vestibulum mi </div>
    <div class="flex-item">2   In viverra dapibus   </div>
    <div class="flex-item">3  Fusce tincidunt diam et  </div>
    <div class="flex-item">4  Nulla in dui vel est  </div>
    <div class="flex-item">5 at diam in lobortis  </div>
  </div>
</br>
<div class="flex-container wrap">
    <div class="flex-item"> 1 Curabitur ac vestibulum mi </div>
    <div class="flex-item">2   In viverra dapibus   </div>
    <div class="flex-item">3  Fusce tincidunt diam et  </div>
    <div class="flex-item">4  Nulla in dui vel est  </div>
    <div class="flex-item">5 at diam in lobortis  </div>
  </div>
</br>
```

```
<div class="flex-container wrap-reverse">
    <div class="flex-item">1 Curabitur ac vestibulum mi</div>
    <div class="flex-item">2   In viverra dapibus   </div>
    <div class="flex-item">3  Fusce tincidunt diam et  </div>
    <div class="flex-item">4  Nulla in dui vel est  </div>
    <div class="flex-item">5 at diam in lobortis  </div>
  </div>
</body>
```

```
.flex-container {  display: flex;  flex-wrap: nowrap;  }
.wrap {  flex-wrap: wrap;  }
.wrap-reverse {  flex-wrap: wrap-reverse;  }
```

В результате мы получим вот такое поведение дочерних блоков

flex-flow.

Для короткой записи свойств flex-direction и flex-wrap существует свойство: flex-flow.

Возможные значения: можно задавать оба свойства или только какое-то одно.

flex-flow: column;

flex-flow: wrap-reverse;

flex-flow: column-reverse wrap;

flex-flow: row-reverse wrap-reverse

пример row-reverse wrap-reverse

Justify

Для выравнивания элементов по осям внутри контейнера есть несколько свойств:

justify-content, align-items и align-self.

justify-content и align-items применяются к родительскому контейнеру,

align-self — к дочерним.

justify-content

justify-content отвечает за выравнивание по главной оси (напоминаю , при этом неважно какая ваша ось главная по горизонтали или по вертикали ! все зависит от ваших задач-)

justify-content: flex-start; элементы выравниваются от начала главной оси (значение по умолчанию);

justify-content: flex-end; элементы выравниваются от конца главной оси;

Justify-content: center; элементы выравниваются по центру главной оси;

justify-content: space-between; элементы выравниваются по главной оси, распределяя свободное место между собой;

justify-content: space-around; элементы выравниваются по главной оси, распределяя свободное место вокруг себя.

Как мы делали ранее, меняем родителям (flex-container) . свойства и классы

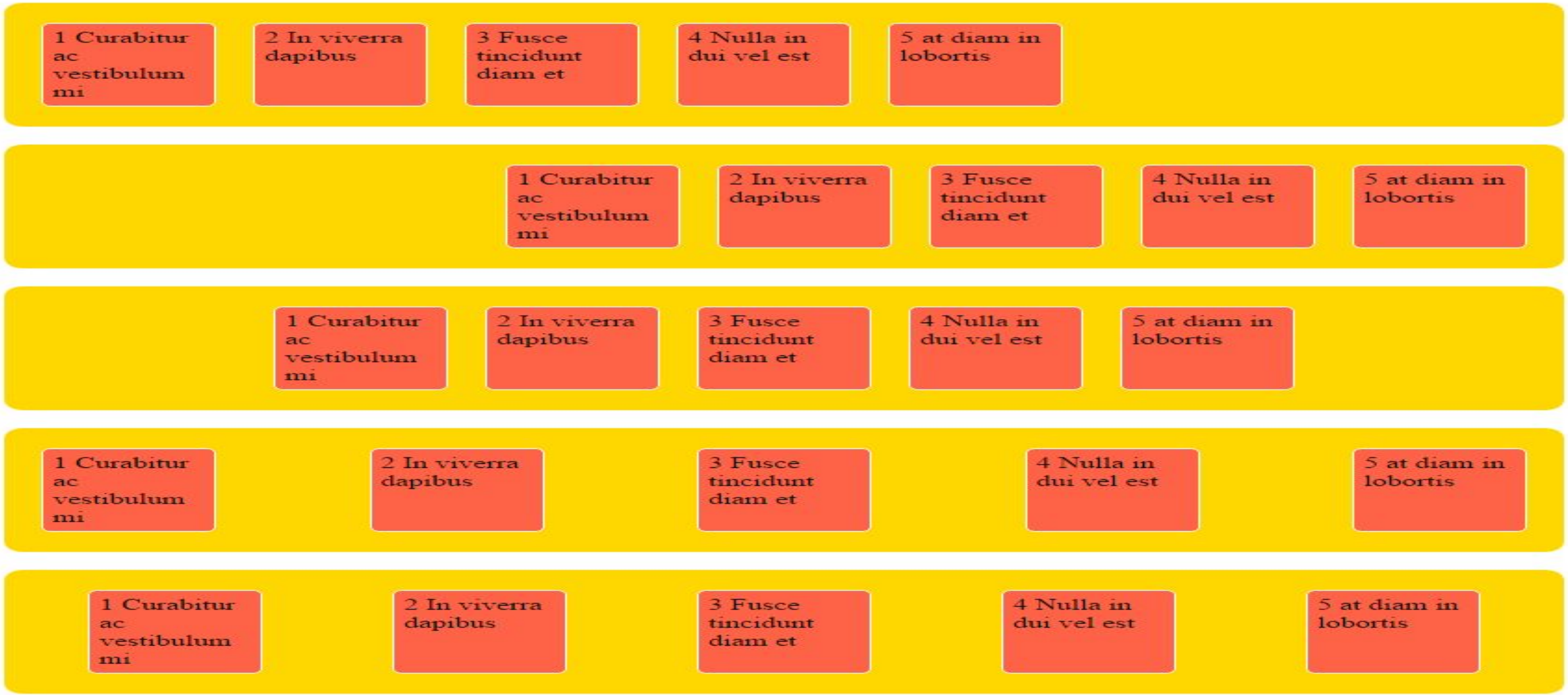
```
<body>
  <div class="flex-container">
    <div class="flex-item"> 1 Curabitur ac vestibulum mi    </div>
    <div class="flex-item">2   In viverra dapibus   </div>
    <div class="flex-item">3  Fusce tincidunt diam et  </div>
    <div class="flex-item">4  Nulla in dui vel est  </div>
    <div class="flex-item">5 at diam in lobortis  </div>
  </div>
</br>
<div class="flex-container flex-end">
    <div class="flex-item"> 1 Curabitur ac vestibulum mi    </div>
    <div class="flex-item">2   In viverra dapibus   </div>
    <div class="flex-item">3  Fusce tincidunt diam et  </div>
    <div class="flex-item">4  Nulla in dui vel est  </div>
    <div class="flex-item">5 at diam in lobortis  </div>
  </div>
```

```
</br>
<div class="flex-container center">
    <div class="flex-item"> 1 Curabitur ac vestibulum mi    </div>
    <div class="flex-item">2   In viverra dapibus   </div>
    <div class="flex-item">3  Fusce tincidunt diam et  </div>
    <div class="flex-item">4  Nulla in dui vel est  </div>
    <div class="flex-item">5 at diam in lobortis  </div>
  </div>
</br>
<div class="flex-container space-between">
    <div class="flex-item"> 1 Curabitur ac vestibulum mi    </div>
    <div class="flex-item">2   In viverra dapibus   </div>
    <div class="flex-item">3  Fusce tincidunt diam et  </div>
    <div class="flex-item">4  Nulla in dui vel est  </div>
    <div class="flex-item">5 at diam in lobortis  </div>
  </div>
</br>
```

```
<div class="flex-container space-around">
    <div class="flex-item"> 1 Curabitur ac vestibulum mi    </div>
    <div class="flex-item">2   In viverra dapibus   </div>
    <div class="flex-item">3  Fusce tincidunt diam et  </div>
    <div class="flex-item">4  Nulla in dui vel est  </div>
    <div class="flex-item">5 at diam in lobortis  </div>
  </div>
</body>

.flex-container {  display: flex;  justify-content: flex-start;  }
.flex-end {  justify-content: flex-end;  }
.center {  display: flex;  justify-content: center;  }
.space-between {  justify-content: space-between;  }
.space-around {  justify-content: space-around; }
```

В итоге мы получим вот такие результаты : justify-content: flex-start; justify-content: flex-end; justify-content: center; justify-content: space-between; justify-content:space-around



Align-items

Flex-элементы могут быть выравнены по поперечной оси текущей линии flex-контейнера, похоже на justify-content, но в перпендикулярном направлении. align-items устанавливает выравнивание для всех flex-элементов.

align-items: stretch; (значение по умолчанию)Flex-элементы заполняют всю высоту (или ширину) поперечной оси flex-контейнера.

align-items: flex-start; Flex-элементы располагаются в начале поперечной оси flex-контейнера

align-items: flex-end; Flex-элементы располагаются в конце поперечной оси flex-контейнера.

align-items: center; Flex-элементы располагаются в центре поперечной оси flex-контейнера.

align-items: baseline; Flex-элементы выровнены таким образом, что их базовая линия находится на одном уровне.

На практике мы опять же добавляем соответственные классы и свойства на родителей

```
.stretch {align-items: stretch;}
.flex-start {align-items: flex-start;}
.flex-end {align-items: flex-end;}
.center {align-items: center;}
.baseline {align-items: baseline;}
```

```
<body>
  <div class="flex-container stretch">
    <div class="flex-item"> 1 Curabitur ac vestibulum mi    </div>
    <div class="flex-item">2   In viverra dapibus   </div>
    <div class="flex-item">3  Fusce tincidunt diam et  </div>
    <div class="flex-item">4  Nulla in dui vel est  </div>
    <div class="flex-item">5 at diam in lobortis  </div>
  </div>
</br>
<div class="flex-container align-start ">
    <div class="flex-item"> 1 Curabitur ac vestibulum mi    </div>
    <div class="flex-item">2   In viverra dapibus   </div>
    <div class="flex-item">3  Fusce tincidunt diam et  </div>
    <div class="flex-item">4  Nulla in dui vel est  </div>
    <div class="flex-item">5 at diam in lobortis  </div>
  </div>
```

```
</br>
<div class="flex-container align-end">
    <div class="flex-item"> 1 Curabitur ac vestibulum mi    </div>
    <div class="flex-item">2   In viverra dapibus   </div>
    <div class="flex-item">3  Fusce tincidunt diam et  </div>
    <div class="flex-item">4  Nulla in dui vel est  </div>
    <div class="flex-item">5 at diam in lobortis  </div>
  </div>
</br>
<div class="flex-container align-center">
    <div class="flex-item"> 1 Curabitur ac vestibulum mi    </div>
    <div class="flex-item">2   In viverra dapibus   </div>
    <div class="flex-item">3  Fusce tincidunt diam et  </div>
    <div class="flex-item">4  Nulla in dui vel est  </div>
    <div class="flex-item">5 at diam in lobortis  </div>
  </div>
```

```
</br>
<div class="flex-container baseline">
    <div class="flex-item"> 1 Curabitur ac vestibulum mi    </div>
    <div class="flex-item">2   In viverra dapibus   </div>
    <div class="flex-item">3  Fusce tincidunt diam et  </div>
    <div class="flex-item">4  Nulla in dui vel est  </div>
    <div class="flex-item">5 at diam in lobortis  </div>
  </div>
</body>


.flex-container {  display: flex;}
.stretch { align-items: stretch; }
.center { align-items: center; }
.baseline { align-items: baseline; }
.align-start {  align-items: flex-start;  }
.align-end {  justify-content: flex-end;  }
```


1 Curabitur ac vestibulum mi
2 In viverra dapibus
3 Fusce tincidunt diam et
4 Nulla in dui vel est
5 at diam in lobortis
1 Curabitur ac vestibulum mi
2 In viverra dapibus
3 Fusce tincidunt diam et
4 Nulla in dui vel est
5 at diam in lobortis
1 Curabitur ac vestibulum mi
2 In viverra dapibus
3 Fusce tincidunt diam et
4 Nulla in dui vel est
5 at diam in lobortis
1 Curabitur ac vestibulum mi
2 In viverra dapibus
3 Fusce tincidunt diam et
4 Nulla in dui vel est
5 at diam in lobortis
1 Curabitur ac vestibulum mi
2 In viverra dapibus
3 Fusce tincidunt diam et
4 Nulla in dui vel est
5 at diam in lobortis

Align-self

Свойство align-self позволяет определять выравнивание для отдельных flex-элементов.
align-self: auto; align-self:flex-start align-self:flex-end align-self:center align-self:baseline align-self: stretch;


```
.flex-container {display: flex;  align-items: flex-start;  }
.flex-item:nth-child(3) { align-self: stretch }
.flex-item:nth-child(4) { align-self: flex-end}
```

В заключение

1.Разберитесь в flexbox и знайте его основы. Так намного легче достичь ожидаемого результата.

2.Значение float у flex-блоков не учитывается и не имеет значения. Это, наверно, как-то можно использовать для graceful degradation при переходе на flexbox.

3.flexbox очень хорошо подходит для верстки веб-компонентов и отдельных частей веб-страниц, но показал себя не с лучшей стороны при верстке базовых макетов (расположение article, header, footer, navbar и т. п.)

4.Не следует использовать flexbox там, где в этом нет необходимости.

5.Не забывайте про margin-ы. Они учитываются при установке выравнивания по осям. Также важно помнить, что margin-ы в flexbox не “коллапсятся”, как это происходит в обычном потоке.

6.Определение регионов и изменение порядка контента во многих случаях все-таки полезно делать зависимым от структуры страницы. Продумывайте это.

Я думаю, что flexbox, конечно же, не вытеснит все остальные способы верстки, но, безусловно, в ближайшее время займет достойную нишу при решении огромного количества задач. И уж точно, пробовать работать с ним нужно уже сейчас

ДЗ

http://html5.by/blog/flexbox/

https://habrahabr.ru/post/242545/

http://www.area53.ru/css3/kak-rabotaet-flexbox.html

http://frontender.info/a-guide-to-flexbox/