Методпозиционирования элементов Flexbox

- Свойство flex-direction
- Свойство flex-wrap
- Свойство flex-flow
- Свойство justify-content
- Свойство align-items
- Свойство align-content
- Свойства для дочерних элементов
- Как упростить себе верстку?

Продолжаем знакомство с CSS. Перейдем к относительно новой технологии, которая имеет достаточно широкую поддержку браузерами и значительно помогает в верстке веб-страниц. Flexbox предлагает множество инструментов для верстки сложных и в тоже время гибких макетов.

Главной идеей модуля Flexbox является дать контейнеру возможность динамически менять ширину и высоту для наилучшего заполненияпространства.

Так как Flexbox это не одно свойство, а множество, давайте для начала разберемся с понятиями. Для этого обратимся к рисунку 1.

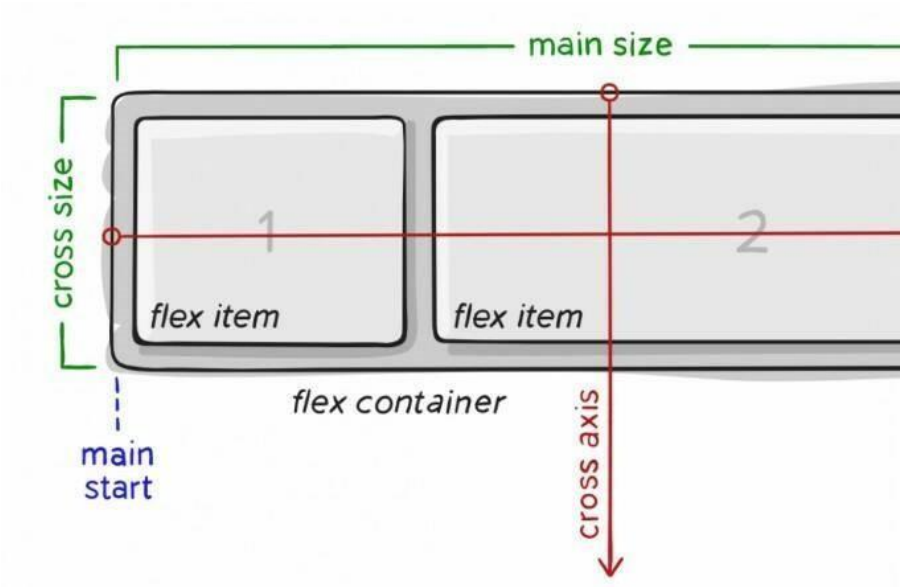

Рисунок взят из спецификации. Поговорим теперь о каждом понятии. Элементы могут располагаться как по main axis (главная ось от main start до main end), так и по cross axis (поперечная ось от cross startдо cross

end). Flex container (flex-контейнер) – родительский блок, который в последствии мы будем наделять свойством гибкости (flex). Flex item (flex- элемент) – дочерние элементы. Остальное интуитивно понятно. Так же стоит заметить, что главная ось flex- контейнера не обязательно горизонтальная (этот момент зависит от свойства flex-direction, о чем будет далее).

Создадим пару блоков с разным по объему содержимым и начнем изучать свойства Flexbox.

Пример

```
<div class = "main">
    <div class = "block">Текст Текст Текст Текст Текст Текст Текст </div>
    <div class = "block">Текст Текст </div>
    <div class = "block">Текст Текст Текст Текст Текст Текст Текст Текст Текст
</div>
    <div class = "block">Текст </div>
    <div class = "block">Текст Текст Текст </div>
</div>
```

И код CSS:

Пример

```
.main {
    width: 500px;
    background: #f1f1f1;
}
.main .block {
    width: 50px;
    background: #00FF00;
    margin: 0 0 5px 5px;
}
```

Получаем результат как на рисунке 2.

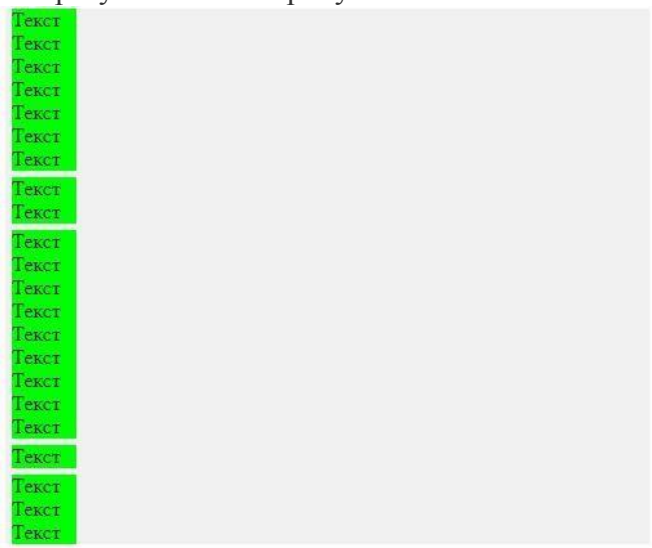

Рисунок 2. Результат верстки

Определим flex-контейнер (в нашем случае это блок с классом .main). Далеекод CSS:

Пример

```
.main {
    /* предыдущий код */
    display: flex;
}
```

Свойство flex-direction

По умолчанию свойство flex-direction имеет значение row, поэтому наши блоки разложились по главной оси слева направо. Свойство flex- direction имеет следующие значения:

- row– о нем уже сказали;
- row-reverse– раскладка блоков справа налево по главной оси;
- column– все то же самое, но по вертикальной (поперечной) оси сверху вниз;
- column-reverse– снизу вверх.

Свойство flex-wrap

Следующее свойство это flex-wrap, которое по умолчанию имеет значение nowrap. Значение nowrap означает, что элементы будут стараться уместиться на одной строке. Теперь уменьшим блок .main до 200 пикселей и посмотрим результат. Как видите, дочерние блоки вышли за пределы родительского блока, так как по умолчанию у свойства flex- wrapзначение nowrap. Напишем свойство flex-wrapсозначением wrap(означает перенос блоков, если они не помещаются):

Пример

```
.main {
    width: 200px;
    background: #f1f1f1;
    display: flex;
    flex-wrap: wrap;
}
```

Свойство flex-flow

Свойства flex-direction и flex-wrap можно сократить, объединив их в одно свойство: flex-flow:

Пример

```
.main {
    /* предыдущий код */
    flex-flow: row wrap;
}
```

Свойство justify-content

Следующим свойством является justify-content, которое определяет выравнивание вдоль главной оси. Данное свойство позволяет распределить оставшееся свободное место. В нашем случае, правая часть родительскогоконтейнера пустует, рисунок 3.

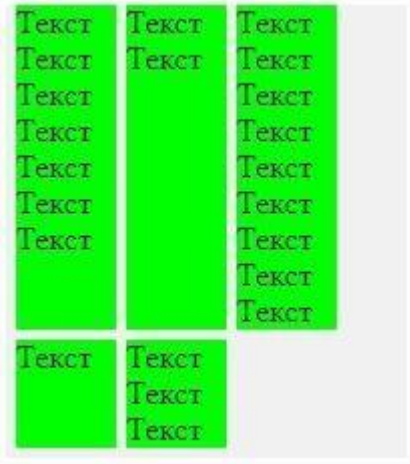

Рисунок 3

По умолчанию свойство justify-content имеет значение flex-start, что означает о выравнивании дочерних элементов по левому краю. Другие значения свойства justify-content:

- flex-end- элементы притянуты к правой части родительского блока;
- space-between- равномерное распределение элементов по главной оси, гдепервый элемент на линии расположен main start, а последний на main end.
- space-around- элементы расположены на одной линии с одинаковымпространством вокруг них.
- space-evenly- элементы расположены таким образом, что расстояниемежду двумя любыми элементами одинаковое;
- center- элементы располагаются по центру.

Свойство align-items

Перейдем к свойству align-items. Это свойство определяет, как будут располагаться элементы относительно горизонтальной (поперечной) оси. Доступны следующие значения свойства align-items:

- stretch– элемент растягивается по высоте и заполняет все доступноепространство;
- flex-start– элементы выравниваются по верхней линии строки;
- flex-end– элементы выравниваются по нижней линии строки;
- center– элементы центрируются по горизонтальной оси;
- baseline– элементы выравниваются по их базовой линии.

Свойство align-content

Свойство align-content выравнивает на поперечной оси в пределах flex- контейнера, когда есть свободное пространство. Это свойство не сработает, еслиесть только одна строка.

- flex-start– элементы сдвинуты к верхней части контейнера;
- flex-end– элементы сдвинуты к нижней части контейнера;
- stretch– элементы растягиваются и заполняют все свободное пространство;
- center– элементы выравниваются по центру контейнера;
- space-between– элементы равномерно распределяются в контейнере;
- space-around– элементы распределены с равномерным пространствомвокруг;
- space-evenly– элементы растянуты на одинаковое расстояние друг отдруга.

Свойства для дочерних элементов

Перейдем к свойствам для дочерних элементов. Для каждого элемента можно задавать пропорцию, которую он будет занимать при помощи свойства flex- grow. Если задать для всех дочерних элементов значение 1:

Пример

```
.main .block {
    /* предыдущий код */
    flex-grow: 1;
}
```

Получится следующий результат, рисунок 4.

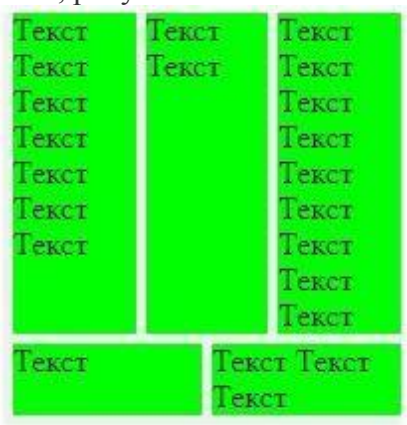

Рисунок 4

Противоположным для flex-grow, является свойство flex-shrink, которое определяет

возможность сжатия элемента. Оба свойства не принимаютотрицательные свойства.

Следующим свойством для дочерних элементов рассмотрим flex-basis, которое определяет размер перед заполнением оставшегося пространства по умолчанию. В качестве значения принимает длину (50%, 100px и т.д.) или ключевое слово, например, content. Значение по умолчанию у свойства flex-basis: auto.

Кстати, у этих трех свойств есть сокращение. Вместо них можно использовать одно свойство flex. Синтаксис следующий:

Пример

```
.main .block {
    /* предыдущий код */
    flex: none | [ <'flex - grow'> <'flex - shrink'>? || <'flex - basis'> ]
}
```

Второй и третий параметры необязательные.

Так же для отдельного элемента можно переопределить выравнивание, которое указано у родителя при помощи свойства align-items. Переопределить можно при помощи свойства align-self.

Как упростить себе верстку?

Откройте страницу с вашим flexbox’ом, после чего откройте консоль разработчика (кнопка F12). Теперь выберите flex-контейнер, рисунок 5.

Рисунок 5

И теперь найдите CSS свойство displayсо значением flex. После нажмите наиконку рядом со значением flex, рисунок 6.

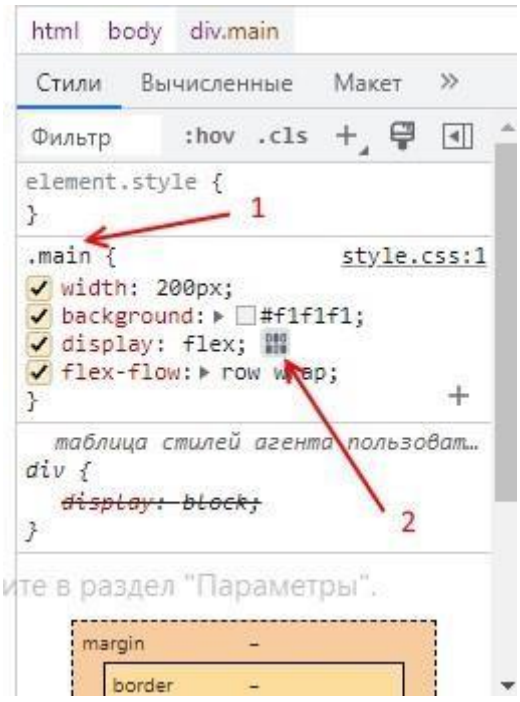

Рисунок 6

Как видите, все что мы сегодня обсуждали очень понятно визуализировано вконсоли разработчика, рисунок 7.

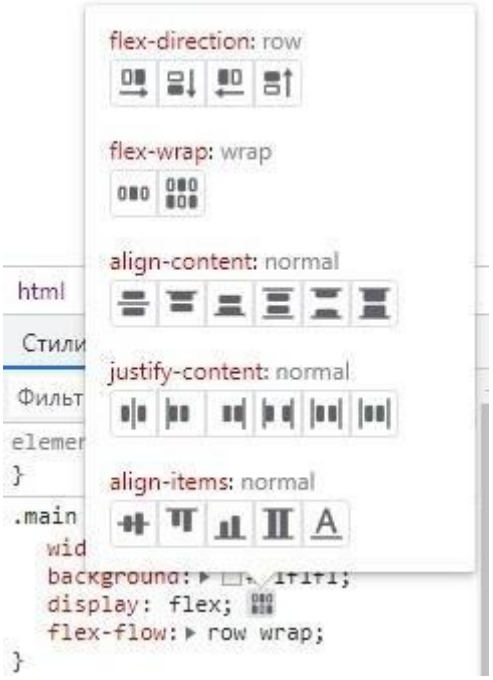

Рисунок 7. Открытая вкладка свойств для flexbox

В этом уроке мы познакомились с относительно новой технологией – Flexbox. Разобрались с базовыми понятиями, основными свойствами для управления компоновкой гибких блоков и свойствами дочерних элементов в контексте данной темы.