Flexbox Layout

- Презентация: ссылка

Что это такое?

Модуль Flexbox Layout (Flexible Box) направлен на обеспечение более эффективного способа компоновки, выравнивания и распределения пространства между элементами в контейнере, даже если их размер неизвестен и/или меняется (отсюда и слово «гибкий»).

Основная идея гибкого макета состоит в том, чтобы дать контейнеру возможность изменять ширину/высоту (и порядок) своих элементов, чтобы наилучшим образом заполнить доступное пространство (в основном, чтобы приспособиться ко всем типам устройств отображения и размерам экрана). Гибкий контейнер расширяет элементы, чтобы заполнить доступное свободное пространство, или сжимает их, чтобы предотвратить переполнение.

Самое главное, макет flexbox не зависит от направления, в отличие от обычных макетов (блок, который расположен по вертикали, и встроенный, который расположен по горизонтали). Хотя они хорошо работают для страниц, им не хватает гибкости (без каламбура) для поддержки больших или сложных приложений (особенно когда речь идет об изменении ориентации, изменении размера, растяжении, сжатии и т. д.).

Терминология

Поскольку flexbox — это целый модуль, а не отдельное свойство, он включает в себя множество вещей, включая весь набор свойств. Некоторые из них предназначены для установки в контейнере (родительский элемент, известный как «гибкий контейнер»), тогда как другие предназначены для установки в дочерних элементах (так называемые «гибкие элементы»).

Если «обычная» компоновка основана как на блочном, так и на встроенном направлениях потока, то гибкая компоновка основана на «направлениях гибкого потока». Пожалуйста, взгляните на этот рисунок из спецификации, объясняющий основную идею гибкого макета.

Элементы будут располагаться либо по главной оси (от начала до конца), либо по поперечной оси (от начала до конца).

- main-axis — главная ось гибкого контейнера — это основная ось, вдоль которой располагаются гибкие элементы. Внимание, она не обязательно горизонтально расположена; это зависит от свойства flex-direction (см. ниже).
- main-start | main-end — flex-элементы размещаются внутри контейнера, начиная с main-start и заканчивая main-end.
- cross axis – ось, перпендикулярная главной оси, называется поперечной осью. Его направление зависит от направления главной оси.
- cross-start | cross-end — гибкие линии заполняются элементами и помещаются в контейнер, начиная с cross-start стороны флекс-контейнера и двигаясь к cross-end.

Свойства контейнера(родителя):

display

Это свойство определяет флекс-контейнер, это необходимо, чтобы его дочерние элементы могли гибко располагаться.

```
.container {
  display: flex;
}
```

flex-direction

Это свойство определяет направление главной оси: горизонтально

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

- row (default): строка
- row-reverse: строка в обратном
- column: столбец
- column-reverse: столбец снизу вверх

flex-wrap

По умолчанию все флекс элементы будут пытаться уместиться на строке, это можно изменить, если поменять значение свойства flex-wrap на wrap

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

- nowrap (default): все элементы будут на одной строке
- wrap: элементы будут перноситься с однйо строки на следующую сверху-вниз
- wrap-reverse: элементы будут перноситься с однйо строки на следующую снизу-вверх

flex-flow

Сокращенная запись - которая объединяет flex-direction и flex-wrap Значением по умолчанию является row nowrap.

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

justify-content

Определяет то, как располагаются элементы относительно главной оси.

```
.container {
  justify-content: flex-start | flex-end | center | space-between |
space-around | space-evenly | start | end | left | right;
}
```

- flex-start (default): элементы сдвинты к началу flex-direction.
- flex-end: элементы сдвинты к концу flex-direction.
- start: элементы сдвинты к началу writing-mode direction.
- end: элементы сдвинты к концу writing-mode direction.
- center: элементы сдвинуты в центр
- space-between: элементы равномерно расположены
- space-around: элементы расположены равномерно, добавлено место по краям

- space-evenly: элементы расположены равномерно, расстояние между элементами и от элементов до краев равно

align-content

Это свойство выравнивает положение строк внутри гибкого контейнера.

```
.container {
  align-content: flex-start | flex-end | center | space-between | space-around | space-evenly | stretch | start | end | baseline | first baseline | last baseline + ... safe | unsafe;
}
```

- normal (default): элементы находятся в положении по умолчанию, как если бы не было задано свойство
- flex-start / start: строки объектов расположены ближе к началу контейнера (flex-start и start работают одинакого, но flex-start лучше поддерживается)
- flex-end / end: строки объектов расположены ближе к концу контейнера
- center: объекты расположены ближе к центру контейнера
- space-between: элементы равномерно распределены, первая строка в начале контейнера, последняя - в конце
- space-around: элементы равномерно распределен с равным пространством вокруг каждой строки
- space-evenly: элементы равномерно распределен с равным пространством вокруг каждой строки
- stretch: строки растянуты чтобы заполнить свободно пространство

gap

```
.container {
  display: flex;
  ...
  gap: 10px;
  gap: 10px 20px; /* row-gap column gap */
  row-gap: 10px;
  column-gap: 20px;
}
```

Свойство gap управляет расстоянием между флекс-элементами. Место добавляется только между самими элементами, у внешних краев - не добавляется. Стоит воспринимать - как минимальное расстояние.

- Игры на закрепление материала:
    1. https://flexboxfroggy.com/#ru
    2. http://www.flexboxdefense.com/