STEP ACADEMY

УРОК №14-15

Flexbox. Многострочный

FLEXBOX

- **CSS flexbox** *(Flexible Box Layout Module)* — модуль макета гибкого контейнера — представляет собой способ компоновки элементов. В основе flexbox лежит идея оси.
- Flexbox состоит из **flex-контейнера** — родительского контейнера и **flex-элементов** — дочерних блоков. Дочерние элементы могут выстраиваться в строку или столбик, а оставшееся свободное пространство распределяется между ними различными способами.

FLEXBOX

.Для контейнера нужно прописать свойство.

.display: flex;

ОСИ

- Одним из основных понятий в flexbox являются оси.
- Главной осью flex-контейнера является направление, в соответствии с которым располагаются все его дочерние элементы.
- Поперечной осью называется направление, перпендикулярное главной оси.
- Направление главной оси flex-контейнера можно задавать, используя базовое css свойство flex-direction.
- row, row-reverse, column, column-reverse

`flex-direction` – направление главной оси

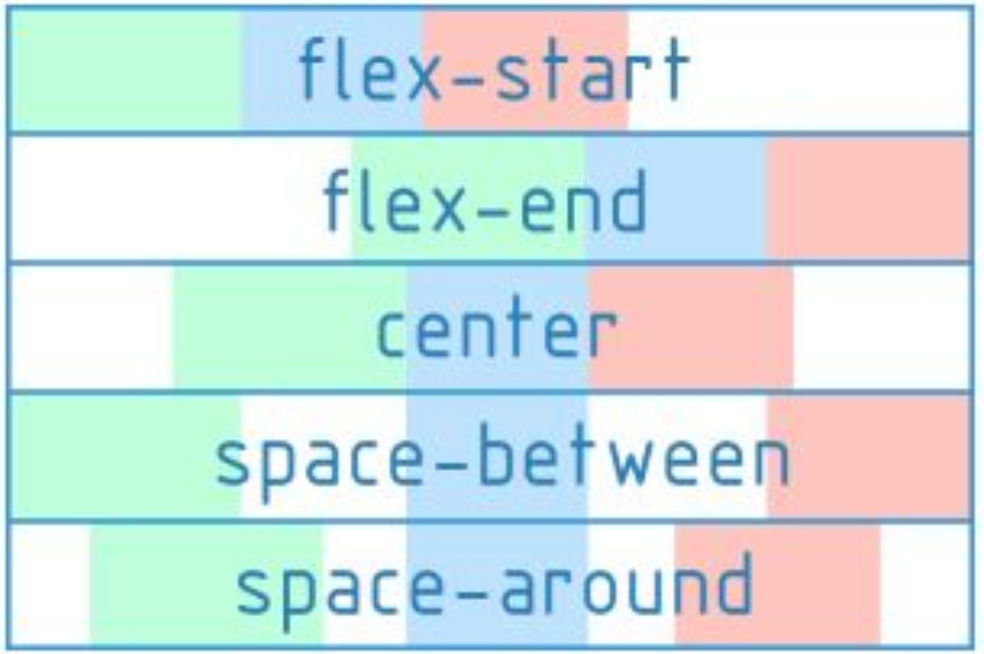
flex-direction: column

flex-direction: row

JUSTIFY-CONTENT – ВЫРАВНИВАНИЕ ПО ГЛАВНОЙ ОСИ

- Css свойство justify-content определяет то, как будут выровнены элементы вдоль главной оси.

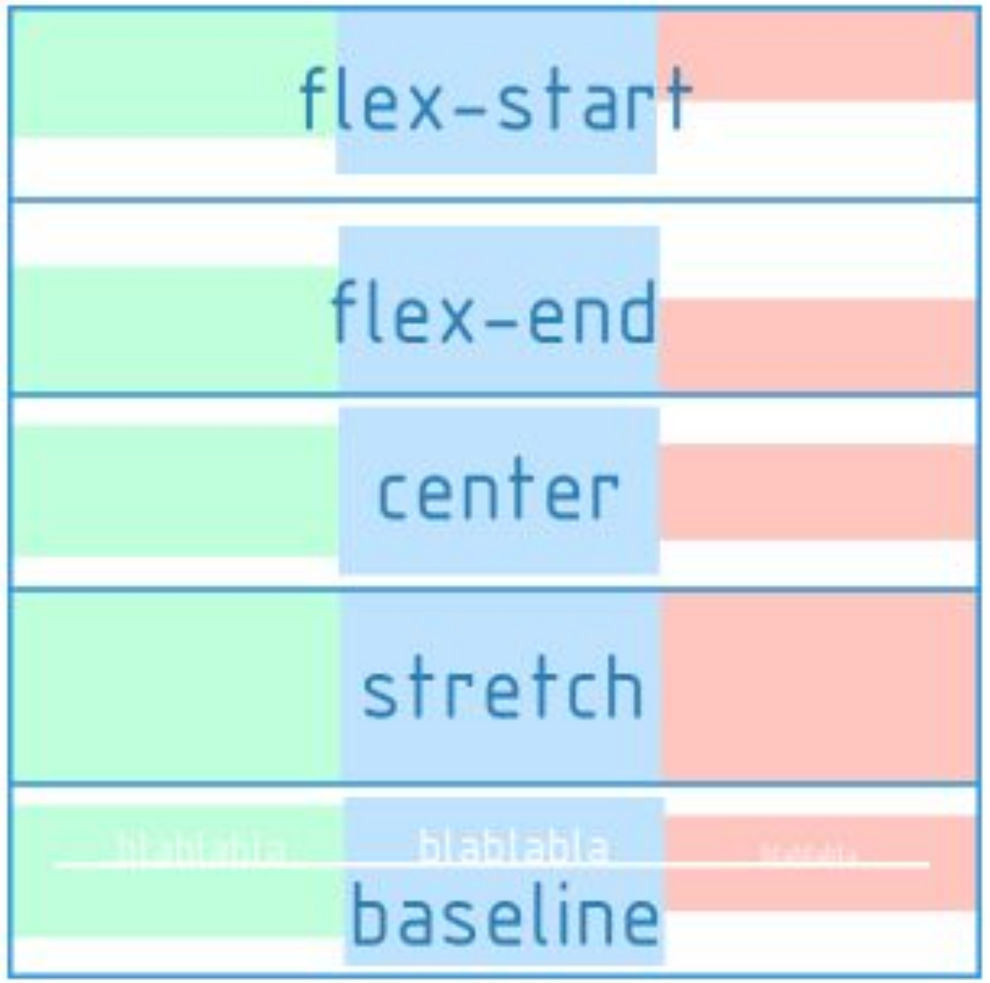
- Доступные значения justify-content:
- flex-start (значение по умолчанию) : блоки прижаты к началу главной оси.
- flex-end: блоки прижаты к концу главной оси.
- center: блоки располагаются в центре главной оси.
- space-between: первый блок располагается в начале главной оси, последний блок – в конце, все остальные блоки равномерно распределены в оставшемся пространстве.
- space-around: все блоки равномерно распределены вдоль главной оси, разделяя все свободное пространство поровну.


flex-start
flex-end
center
space-between
space-around

ALIGN-ITEMS – ВЫРАВНИВАНИЕ ПО ПОПЕРЕЧНОЙ ОСИ

- Css свойство align-items определяет то, как будут выровнены элементы вдоль поперечной оси.

- Доступные значения align-items:
- flex-start: блоки прижаты к началу поперечной оси
- flex-end: блоки прижаты к концу поперечной оси
- center: блоки располагаются в центре поперечной оси
- baseline: блоки выровнены по их baseline
- stretch (значение по умолчанию) : блоки растянуты, занимая все доступное место по поперечной оси, при этом все же учитываются min-width/max-width, если таковые заданы.


flex-start
flex-end
center
stretch
blablabla
blablabla
baseline

. CSS свойства flex-direction, justify-content, align-items должны применяться непосредственно к flex-контейнеру, а не к его дочерним элементам.

ДЕМО

- https://html5.by/blogdemo/flexbox/flex-direction-align-justify.html

ДОПОЛНИТЕЛЬНО

- https://flexboxfroggy.com/

МНОГОСТРОЧНАЯ ОРГАНИЗАЦИЯ БЛОКОВ ВНУТРИ FLEX-КОНТЕЙНЕРА

- flex-wrap
- Доступные значения flex-wrap:
- nowrap (значение по умолчанию) : блоки расположены в одну линию слева направо (в rtl справа налево)
- wrap: блоки расположены в несколько горизонтальных рядов (если не помещаются в один ряд). Они следуют друг за другом слева направо (в rtl справа налево)
- wrap-reverse: то-же что и wrap, но блоки располагаются в обратном порядке.

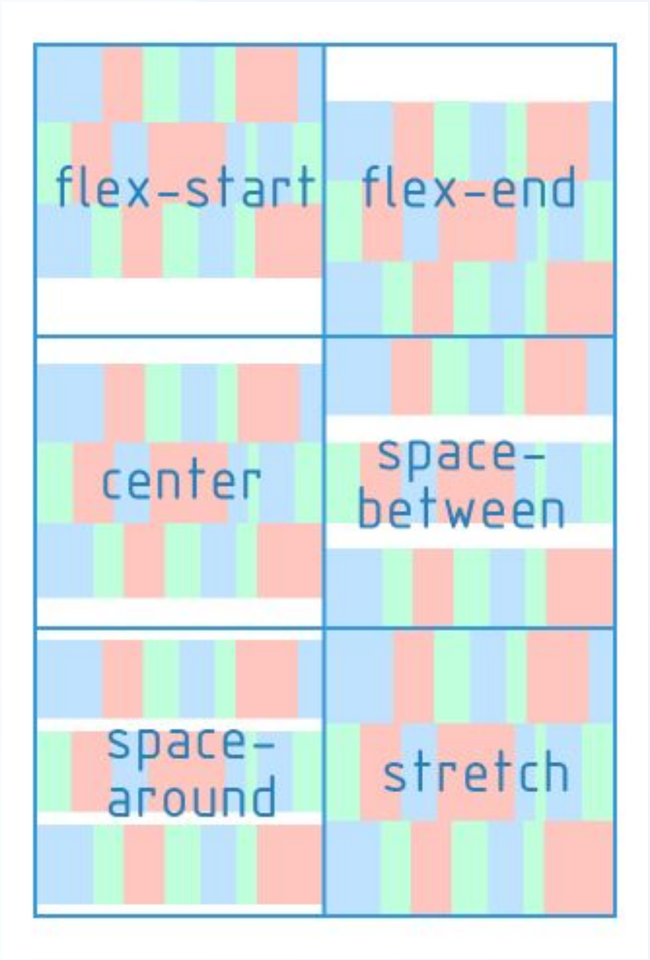
МНОГОСТРОЧНАЯ ОРГАНИЗАЦИЯ БЛОКОВ ВНУТРИ FLEX-КОНТЕЙНЕРА

- flex-flow – удобное сокращение для flex-direction + flex-wrap
- По сути, flex-flow предоставляет возможность в одном свойстве описать направление главной и многострочность поперечной оси. По умолчанию flex-flow: row nowrap.
- flex-flow: <‘flex-direction’> || <‘flex-wrap’>

МНОГОСТРОЧНАЯ ОРГАНИЗАЦИЯ БЛОКОВ ВНУТРИ FLEX-КОНТЕЙНЕРА

- align-content
- Доступные значения align-content:
- flex-start: ряды блоков прижаты к началу flex-контейнера.
- flex-end: ряды блоков прижаты к концу flex-контейнера
- center: ряды блоков находятся в центре flex-контейнера
- space-between: первый ряд блоков располагается в начале flex-контейнера, последний ряд блоков блок – в конце, все остальные ряды равномерно распределены в оставшемся пространстве.
- space-around: ряды блоков равномерно распределены в от начала до конца flex-контейнера, разделяя все свободное пространство поровну.
- stretch (значение по умолчанию): Ряды блоков растянуты, дабы занять все имеющееся пространство.

МНОГОСТРОЧНАЯ ОРГАНИЗАЦИЯ БЛОКОВ ВНУТРИ FLEX-КОНТЕЙНЕРА



CSS правила для дочерних элементов flex-контейнера (flex-блоков)

- flex-basis – базовый размер отдельно взятого flex-блока
- Задает изначальный размер по главной оси для flex-блока до того, как к нему будут применены преобразования, основанные на других flex-факторах. Может быть задан в любых единицах измерения длинны (px, em, %, …) или auto(по умолчанию). Если задан как auto – за основу берутся размеры блока (width, height), которые, в свою очередь, могут зависеть от размера контента, если не указаны явно.

CSS правила для дочерних элементов flex-контейнера (flex-блоков)

- flex-grow – “жадность” отдельно взятого flex-блока.
- Определяет то, на сколько отдельный flex-блок может быть больше соседних элементов, если это необходимо. flex-grow принимает безразмерное значение (по умолчанию 0).
- Если все flex-блоки внутри flex-контейнера имеют flex-grow:1, то они будут одинакового размера.
- Если один из них имеет flex-grow:2, то он будет в 2 раза больше, чем все остальные.

CSS правила для дочерних элементов flex-контейнера (flex-блоков)

- flex-shrink – фактор “сжимаемости” отдельно взятого flex-блока.
- Определяет, насколько flex-блок будет уменьшаться относительно соседних эдементов внутри flex-контейнера в случае недостатка свободного места. По умолчанию равен 1.

- flex – короткая запись для свойств flex-grow, flex-shrink и flex-basis
- flex: none | [<'flex-grow'> <'flex-shrink'>? || <'flex-basis'>]

CSS правила для дочерних элементов flex-контейнера (flex-блоков)

- order – порядок следования отдельно взятого flex-блока внутри flex-контейнера.
- По умолчанию все блоки будут следовать друг за другом в порядке, заданном в html. Однако этот порядок можно изменить с помощью свойства order. Оно задается целым числом и по умолчанию равно 0.

- Значение order не задает абсолютную позицию элемента в последовательности. Оно определяет вес позиции элемента.

ДОПОЛНИТЕЛЬНЫЕ ССЫЛКИ

- https://developer.mozilla.org/ru/docs/Learn/CSS/CSS_layout/Flexbox
- https://html5book.ru/css3-flexbox/
- https://html5.by/blog/flexbox/

Конец

ПОСЛЕСЛОВИЕ

Давайте подведем итоги урока!
Чему мы научились? Что мы использовали?
К чему мы пришли?