

Инженерная школа Информационных технологий и робототехники



Направление подготовки 09.04.01 Информатика и вычислительная техника

Профиль Разработка интернет-приложений

Отделение Информационных технологий

Отчет по лабораторной работе №7 по дисциплине

**«**Пользовательские интерфейсы в Интернет-приложениях**»**

Flexbox

**Тема работы**

Вариант 6

Студент

|  |  |  |  |
| --- | --- | --- | --- |
| **Группа** | **ФИО** | **Подпись** | **Дата** |
| 8ВМ91 | Зайцев Владимир Александрович |  |  |

Проверил

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Должность** | **ФИО** | **Ученая степень, звание** | **Подпись** | **Дата** |
| доцент ОИТ | Саврасов Фёдор Витальевич | к.т.н. |  |  |

Томск – 2020 г.

Задание:

Создать дизайн HTML-документа (SPA или несколько страниц) и наполнить его текстом, формами и изображениями в соответствии с тематикой своего. К документу предъявляются следующие требования:

* документ должен корректно отображаться в различных браузерах;
* документ должен корректно отображаться при различных размерах ок- на браузера;
* следует придерживаться принципа разделения структуры документа и представления документа; все элементы, относящиеся к представлению документа, следует вынести в общую каскадную таблицу стилей в от- дельный файл;
* HTML-документы должны проходить без ошибок проверку на сервисе [http://validator.w3.org](http://validator.w3.org/)
* Применить следующие возможности css:
* Flex container
* Направление, flex-direction
* Flex-wrap – 1
* Flex-flow, порядок элементов – 1
* Выравнивание элементов, justify-content – 1
* Выравнивание элементов, align-items и align-self – 1
* Выравнивание строк и столбцов, align-content – 1
* Управление элементам, flex-basis, flex-shrink и flex-grow – 1
* Многоколоночный дизайн на flexbox – 2

**Flexbox** - это общее название для модуля **Flexible Box Layout**, который имеется в CSS3. Данный модуль определяет особый режим компоновки/верстки пользовательского интерфейса, который называется **flex layout**. В этом плане Flexbox предоставляет иной подход к созданию пользовательского интерфейса, который отличается от табличной или

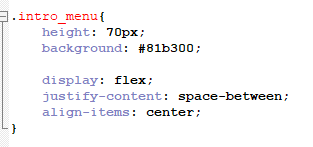
блочной верстки.

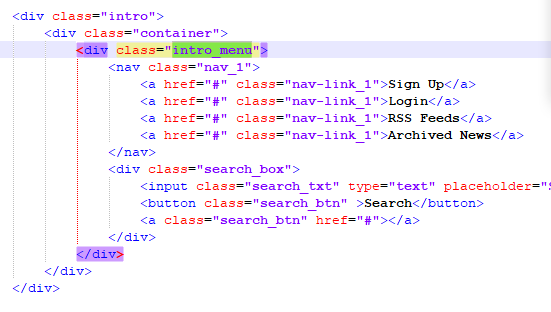
Благодаря Flexbox проще создавать сложные, комплексные интерфейсы, где мы с легкостью можем переопределять направление и выравнивание элементов, создавать адаптивные табличные представления. Кроме того, Flexbox довольно прост в использовании. Единственная проблема, которая может возникнуть при его применении, - это кроссбраузерность. Например, в Internet Explorer поддержка Flexbox и то частичная появилась только в последней версии - IE11. В то же время все современные браузеры, в том числе Microsoft Edge, Opera, Google Chrome, Safari, Firefox, имеют полную поддержку данного модуля.

Основными составляющими компоновки flexbox являются flex-контейнер (flex container) и flex-элементы (flex items). **Flex container** представляет некоторый элемент, внутри которого размещены flex-элементы.

# Flex container

Для создания flex-контейнера необходимо присвоить его стилевому свойству display одно из двух значений: flex или inline-flex.





Результат:



# Направление, flex-direction

Flex-элементы во flex-контейнере могут иметь определенное направление, а именно они могут располагаться в виде строк или в виде столбцов. Для управления направлением элементов CSS3 предоставляет свойство flex- direction. Оно определяет направление элементов и может принимать следующие значения:

* + row: значение по умолчанию, при котором элементы располагаются в виде строки слева направо
  + row-reverse: элементы также располагаются в виде стоки только в обратном порядке справа налево
  + column: элементы располагаются в столбик сверху вниз
  + column-reverse: элементы располагаются в столбик в обратном порядке снизу вверх



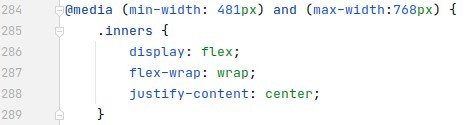
Результат:



# Flex-wrap

Свойство flex-wrap определяет, будет ли flex-контейнер несколько рядов элементов (строк или столбцов) в случае если его размеры недостаточны, чтобы вместить в один ряд все элементы. Это свойство может принимать следующие значения:

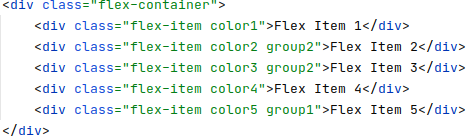
* + nowrap: значение по умолчанию, которое определяет flex-контейнер, где все элементы раполагаются в одну строку (при расположении в виде строк) или один столбец (при расположении в столбик)
  + rap-reverse: то же самое, что и значение wrap, только элементы располагаются в обратном порядке.
  + wrap: если элементы не помещаются во flex-контейнер, то создает дополнительные ряды в контейнере для размещения элементов. При расположении в виде строки содаются дополнительные строки, а при расположении в виде столбца добавляются дополнительные столбцы.



# Flex-flow, порядок элементов

Свойство flex-flow позволяет установить значения сразу для обоих свойств flex-direction и flex-wrap. Оно имеет следующий формальный синтаксис:

flex-flow: [flex-direction] [flex-wrap]



Результат:



# Выравнивание элементов, justify-content

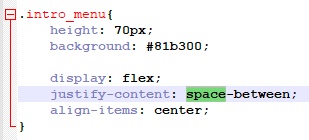
Иногда мы можем сталкиваться с тем, что пространство flex-контейнеров по размеру отличается от пространства, необходимого для flex-элементов. Например:

* + flex-элементы не используют все пространство flex-контейнера
  + flex-элементам требуется большее пространство, чем доступно во flex- контейнере. В этом случае элементы выходят за пределы контейнера.

Для управления этими ситуациями мы можем применять свойство justify- content. Оно выравнивает элементы вдоль основной оси - main axis (при расположении в виде строки по горизонтали, при расположении в виде столбца - по вертикали) и принимает следующие значения:

* + flex-start: значение по умолчанию, при котором первый элемент выравнивается по левому краю контейнера(при расположении в виде строки) или по верху (при расположении в виде столбца), за ним располагается второй элемент и так далее.
  + flex-end: последний элемент выравнивается по правому краю (при расположении в виде строки) или по низу (при расположении в виде столбца) контейнера, за ним выравнивается предпоследний элемент и так далее
  + center: элементы выравниваются по центру
  + space-between: если в стоке только один элемент или элементы выходят за границы flex-контейнера, то данное значение аналогично flex-start. В остальных случаях первый элемент выравнивается по левому краю (при расположении в виде строки) или по верху (при расположении в виде столбца), а последний элемент - по правому краю контейнера (при расположении в виде строки) или по низу (при расположении в виде столбца). Все оставшееся пространство между ними равным образом распределяется между остальными элементами
  + space-around: если в строке только один элемент или элементы выходят за пределы контейнера, то его действие аналогично значению center. В ином случае элементы равным образом распределяют пространство между левым и правым краем контейнера, а расстояние между первым

и последним элементом и границами контейнера составляет половину расстояния между элементами.





Результат:



# Выравнивание элементов, align-items и align-self

*Свойство align-items* также выравнивает элементы, но уже по поперечной оси (cross axis) (при расположении в виде строки по вертикали, при расположении в виде столбца - по горизонтали). Это свойство может принимать следующие значения:

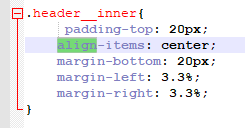
* + stretch: значение по умолчанию, при котором flex-элементы растягиваются по всей высоте (при расположении в строку) или по всей ширине (при расположении в столбик) flex-контейнера
  + baseline: элементы выравниваются в соответствии со своей базовой линией
  + flex-start: элементы выравниваются по верхнему краю (при расположении в строку) или по левому краю (при расположении в столбик) flex-контейнера
  + flex-end: элементы выравниваются по нижнему краю (при расположении в строку) или по правому краю (при расположении в столбик) flex-контейнера
  + center: элементы выравниваются по центру flex-контейнера

*Свойство align-self*

Свойство align-self позволяет переопределить значение свойства align-items для одного элемента. Оно может принимать все те же значения плюс значение "auto":

auto: значение по умолчанию, при котором элемент получает значение от свойства align-items, которое определено в flex-контейнере. Если в контейнере такой стиль не определен, то применяется значение stretch.

* + stretch
  + flex-start
  + flex-end
  + center
  + baseline



Результат:



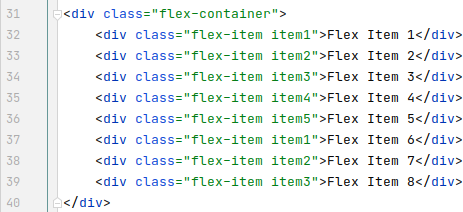
# Выравнивание строк и столбцов, align-content

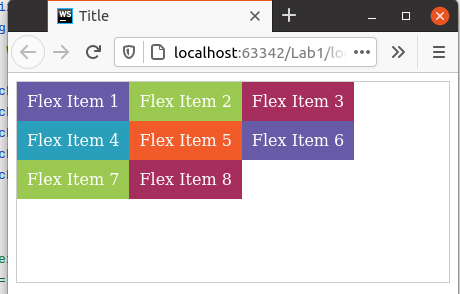
Свойство align-content управляет выравниванием рядов (строк и столбцов) во flex-контейнере и поэтому применяется, если свойство flex-wrap имеет

значение wrap или wrap-reverse. Свойство align-content может иметь следующие значения:

* + stretch: значение по умолчанию, при котором строки (столбцы) растягиваются, занимая все свободное место
  + flex-start: строки (столбцы) выравниваются по началу контейнера (для строк - это верхний край, для столбцов - это левый край контейнера)
  + flex-end: строки (столбцы) выравниваются по концу контейнера (строки - по нижнему краю, столбцы - по правому краю)
  + center: строки (столбцы) позиционируются по центру контейнера
  + space-between: строки (столбцы) равномерно распределяются по контейнеру, а между ними образуются одинаковые отступы. Если же имеющегося в контейнере места недостаточно, то действует аналогично значению flex-start
  + space-around: строки (столбцы) равным образом распределяют пространство контейнера, а растояние между первой и последней строкой (столбцом) и границами контейнера составляет половину расстояния между соседними строками (столбцами).

Результат:

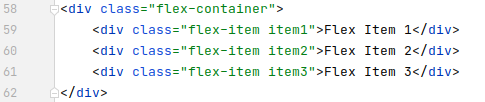




# Управление элементам, flex-basis, flex-shrink и flex-grow

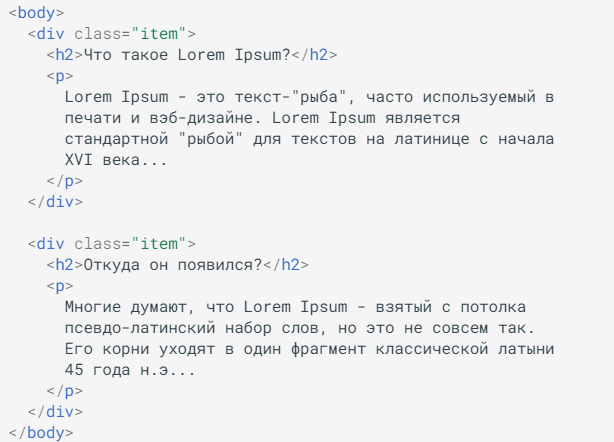
Кроме свойств, устанавливающих выравнивание элементов относительно границ flex-контейнера, есть еще три свойства, которые позволяют управлять элементами:

* + flex-basis: определяет начальный размер flex-элемента
  + flex-shrink: определяет, как flex-элемент будет уменьшаться относительно других flex-элементов во flex-контейнере
  + flex-grow: определяет, как flex-элемент будет увеличиваться относительно других flex-элементов во flex-контейнере

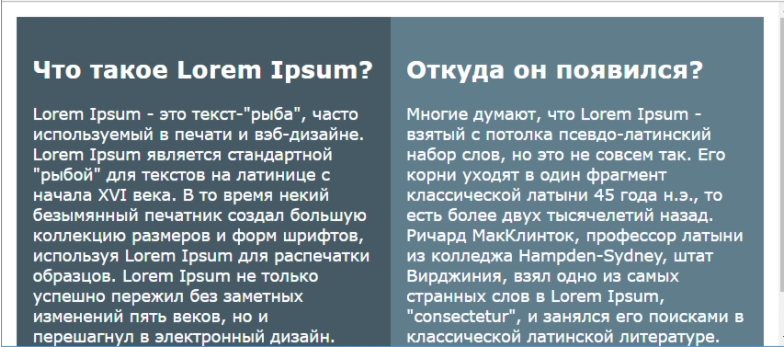


Результат:

# Многоколоночный дизайн на flexbox



Результат:



**Вывод:** в ходе выполнения лабораторной работы был создан дизайн HTML- документа с применением возможностей CSS. В этой работе рассматривались основные блочной верстки. В работе были использованы все перечисленные свойства.

Исходники: <https://github.com/bratyn20/css_html>