МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ «ПОЛОЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ ЕВФРОСИНИИ ПОЛОЦКОЙ»

Факультет информационных технологий

Кафедра технологий программирования

ЛАБОРАТОРНАЯ РАБОТА №4

по дисциплине «Веб - технологии»

Тема: «Создание адаптивного макета с CSS Flexbox»

|  |  |
| --- | --- |
| Выполнил: | студент группы 23-ИТ-1  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В.Ю. Страпко |
| Проверил: | ассистент кафедры ТП  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.А. Макарычева |

|  |
| --- |
|  |

**Цель работы:**Освоить структуру HTML-документа, изучить применение основных тегов и атрибутов для построения полноценной веб-страницы.

1. **Краткая теория**

CSS Flexbox (от flexible box, что переводится как «гибкий контейнер») – это мощный инструмент для создания адаптивной и гибкой верстки веб-страниц. В отличие от традиционных способов позиционирования элементов, таких как float или inline-block, Flexbox предоставляет разработчику простые, интуитивные и более мощные средства для управления выравниванием, распределением пространства и порядком элементов внутри контейнера.

Одной из ключевых особенностей Flexbox является его способность автоматически подстраиваться под доступное пространство, обеспечивая удобное создание адаптивных интерфейсов. Это делает его особенно полезным для современных веб-приложений, где важно корректное отображение элементов на экранах разных размеров.

Основные преимущества Flexbox:

− Простое выравнивание элементов: Flexbox упрощает процесс центрирования, выравнивания и распределения элементов как по горизонтали, так и по вертикали. − Адаптивность: элементы автоматически подстраиваются под размеры контейнера, что позволяет легко реализовывать сложные макеты без необходимости задавать точные размеры для каждого элемента.

− Удобство управления пространством: Flexbox позволяет равномерно распределять свободное место или сжимать элементы, чтобы они помещались в ограниченном пространстве.

− Поддержка большинства браузеров: Flexbox стабильно работает во всех современных браузерах, включая мобильные версии. Flexbox основан на двух основных сущностях:

− Flex-контейнер (flex-container) – это элемент, к которому применяется свойство display: flex или display: inline-flex. После применения одного из этих свойств к элементу, он становится «гибким контейнером» и получает возможность управлять расположением и поведением своих дочерних элементов.

− Flex-элементы (flex-items) – это прямые дочерние элементы flex-контейнера. Только они подчиняются правилам Flexbox. Элементы второго и более глубокого уровня вложенности не затрагиваются, если не задать для них отдельный flex-контейнер.

1. **Практическая часть**

**Задание:** Сконструируйте адаптивный макет, состоящий из:

− Верхнего горизонтального меню, выровненным с использованием Flexbox.

− Основного содержимого, включающего две колонки (главный контент и боковую панель).

− Нижнего колонтитула с выравниванием текста и элементов.

− Настройте поведение элементов при изменении ширины экрана:

− При ширине экрана менее 768px все элементы должны располагаться вертикально.

Дополнительно:

− Используйте медиазапросы для адаптации шрифтов и отступов.

Код главной HTML – страницы представлен в листинге 2.1.

Листинг 2.1 – главная страница.

1. <!DOCTYPE html>
2. <html lang="en">
3. <head>
4. <link rel="stylesheet" href="styles.css">
5. <meta charset="UTF-8">
6. <meta name="viewport" content="width=device-width, initial-scale=1.0">
7. <title>First</title>
8. </head>
9. <body>
10. <header>
11. <a href="second.html">Вторая</a>
12. <a href="third.html">Третья</a>
13. <a href="fouth.html">Четвертая</a>
14. </header>
15. <main>
16. <div class="main-conteiner">
17. <div class="main-content">
18. <h1>Футбол</h1>
19. <p>Футбол — один из самых популярных видов спорта в мире,
20. объединяющий миллионы людей. Игра, в которой две команды
21. сражаются за контроль над мячом, требует не только физической
22. силы, но и стратегического мышления.</p>
23. </div>
24. </div>
25. <div class="s-conteiner">
26. <div class="s-content">
27. <h2>История футбола</h2>
28. <p>Футбол имеет долгую историю, уходящую корнями в древние
29. цивилизации. Современные правила были установлены в 1863 году в
30. Англии, когда была основана Футбольная ассоциация. С тех пор
31. игра распространилась по всему миру, став олимпийским видом
32. спорта и предметом страсти миллионов фанатов.</p>
33. </div>
34. </div>
35. </main>
36. <footer>
37. <div class="footer">
38. <div>&copy; Страпко Вадим Юрьевич</div>
39. <div>
40. <a href="#">Политика конфиденциальности</a>
41. <a href="#">Условия использования</a>
42. <a href="#">Контакты</a>
43. </div>
44. </div>
45. </footer>
46. </body>
47. </html>

Листинг 2.2 – файл styles.css

1. body {
2. margin: 0;
3. font-family: Arial, sans-serif;
4. }
5. .footer{
6. display: flex;
7. flex-direction: column;
8. background-color: #333;
9. color: white;
10. padding: 20px;
11. align-items: center;
12. gap: 10px;
13. }
14. .footer a{
15. text-decoration: none;
16. color: white
17. }
18. .footer a:hover{
19. color: rgb(19, 121, 92);
20. }
21. header {
22. background-color: #273d53;
23. padding: 15px;
24. display: flex;
25. flex-direction: row;
26. justify-content: space-around;
27. align-items: center;
28. }
29. header a {
30. color: #ecf0f1;
31. text-decoration: none;
32. padding: 10px 15px;
33. border-radius: 5px;
34. }
35. header a:hover {
36. background-color: #4d6780;
37. }
38. @media (max-width: 768px) {
39. .main-container, .s-container {
40. flex-direction: column;
41. padding: 10px;
42. }
43. header {
44. flex-direction: column;
45. align-items: flex-start;
46. padding: 10px;
47. }
48. h1 {
49. font-size: 1.5em;
50. }
51. h2 {
52. font-size: 1.3em;
53. }
54. p {
55. font-size: 1em;
56. line-height: 1.5em;
57. }
58. .footer {
59. padding: 15px;
60. }
61. .footer a {
62. padding: 5px;
63. font-size: 0.9em;
64. }
65. }
66. .main-conteiner{
67. display: flex;
68. flex-wrap: wrap;
69. padding: 20px;
70. }
71. .s-conteiner{
72. display: flex;
73. flex-wrap: wrap;
74. padding: 20px;
75. }

В данном листинге были использованы следующие CSS стили:

* font-family: Arial, sans-serif; – **Значение**, указывающее предпочитаемый шрифт и запасной шрифт;
* .footer a:hover – определяет стили для ссылок (<a>) внутри элемента с классом .footer при наведении курсора мыши на эти ссылки;
* flex-direction: row; – Устанавливает направление расположения flex-элементов внутри контейнера;
* justify-content: space-around; – Устанавливает выравнивание flex-элементов вдоль главной оси (в данном случае — горизонтально);
* @media (max-width: 768px) – используется для создания медиа-запросов, которые позволяют применять стили только при определенных условиях, связанных с размерами экрана устройства;
* flex-direction: column; – устанавливает вертикальное направление для элементов внутри flex-контейнера;
* flex-wrap: wrap; – используется для управления тем, как дочерние элементы внутри flex-контейнера будут вести себя, если они не помещаются в одну строку (или столбец, в зависимости от направления);
* flex-direction: row; – в CSS используется для управления направлением, в котором располагаются дочерние элементы внутри flex-контейнера;

**3.Контрольные вопросы**

**1. Определение CSS Flexbox**

**Flexbox** (или Flexbox Layout) — это метод компоновки для CSS, который позволяет легко управлять размещением и выравниванием элементов в контейнере, обеспечивая более эффективное распределение пространства внутри контейнеров, даже когда их размеры неизвестны или динамически изменяются.

**2. Какие основные преимущества Flexbox?**

* **Удобное выравнивание**: Легко выравнивать элементы по горизонтали и вертикали.
* **Автоматическое распределение пространства**: Элементы могут автоматически расширяться или сжиматься, чтобы заполнить доступное пространство.
* **Упрощенное размещение**: Легко изменять порядок элементов и их направление.
* **Поддержка адаптивного дизайна**: Элементы могут плавно адаптироваться к различным размерам экрана.

**3. На каких сущностях основан Flexbox?**

Flexbox основан на двух сущностях:

* **Flex-контейнер**: Элемент, которому присвоено свойство display: flex; или display: inline-flex;.
* **Flex-элементы**: Прямые дочерние элементы флекс-контейнера.

**4. Какие CSS-свойства для расположения Flex-контейнера?**

Основные свойства для размещения флекс-контейнера:

* display
* flex-direction
* flex-wrap
* justify-content
* align-items
* align-content

**5. Определение осей Flexbox**

Flexbox использует две оси для размещения элементов:

* **Главная ось (main axis)**: Направление, в котором располагаются флекс-элементы (по умолчанию — горизонтально).
* **Поперечная ось (cross axis)**: Направление, перпендикулярное главной оси (по умолчанию — вертикально).

**6. Какие CSS-свойства для настраивания направления осей?**

Свойства для настройки направления осей:

* flex-direction
* flex-wrap

**7. Опишите свойство flex-direction и его значения. Опишите свойство flex-wrap и его значения.**

* **flex-direction**: Определяет направление главной оси.
  + row: Элементы располагаются по горизонтали (по умолчанию).
  + row-reverse: Элементы располагаются по горизонтали в обратном порядке.
  + column: Элементы располагаются по вертикали.
  + column-reverse: Элементы располагаются по вертикали в обратном порядке.
* **flex-wrap**: Определяет, будут ли флекс-элементы переноситься на новую строку.
  + nowrap: Элементы не переносятся (по умолчанию).
  + wrap: Элементы переносятся на новую строку.
  + wrap-reverse: Элементы переносятся на новую строку в обратном порядке.

**8. Опишите свойство justify-content и его значения.**

**justify-content**: Определяет, как флекс-элементы распределяются вдоль главной оси.

* flex-start: Элементы выравниваются по началу главной оси (по умолчанию).
* flex-end: Элементы выравниваются по концу главной оси.
* center: Элементы выравниваются по центру главной оси.
* space-between: Элементы распределены по главной оси с равными промежутками между ними.
* space-around: Элементы распределены по главной оси с равными промежутками вокруг них.

**9. Опишите свойство align-items и его значения.**

**align-items**: Определяет, как флекс-элементы выравниваются вдоль поперечной оси.

* stretch: Элементы растягиваются, чтобы заполнить контейнер (по умолчанию).
* flex-start: Элементы выравниваются по началу поперечной оси.
* flex-end: Элементы выравниваются по концу поперечной оси.
* center: Элементы выравниваются по центру поперечной оси.
* baseline: Элементы выравниваются по базовой линии текста.

**10. Опишите свойство align-content и его значения.**

**align-content**: Определяет, как строки флекс-элементов распределяются вдоль поперечной оси, когда имеется свободное пространство.

* flex-start: Строки выравниваются по началу поперечной оси.
* flex-end: Строки выравниваются по концу поперечной оси.
* center: Строки выравниваются по центру поперечной оси.
* space-between: Строки распределены с равными промежутками между ними.
* space-around: Строки распределены с равными промежутками вокруг них.
* stretch: Строки растягиваются, чтобы заполнить контейнер.

**11. Опишите свойство align-self и его значения.**

**align-self**: Позволяет переопределить значение align-items для конкретного флекс-элемента.

* auto: Использует значение align-items родительского контейнера (по умолчанию).
* flex-start: Элемент выравнивается по началу поперечной оси.
* flex-end: Элемент выравнивается по концу поперечной оси.
* center: Элемент выравнивается по центру поперечной оси.
* baseline: Элемент выравнивается по базовой линии текста.
* stretch: Элемент растягивается, чтобы заполнить контейнер.

**12. Опишите порядок следования flex-элементов и можно ли его изменить?**

Порядок следования флекс-элементов задается свойством order. Элементы по умолчанию имеют значение order: 0. Элементы с меньшим значением order отображаются первыми.

**13. Перечислите основные свойства флекс-контейнера.**

Основные свойства флекс-контейнера:

* display
* flex-direction
* flex-wrap
* justify-content
* align-items
* align-content

**14. Перечислите основные свойства флекс-элементов.**

Основные свойства флекс-элементов:

* flex-grow
* flex-shrink
* flex-basis
* align-self
* order

**15. Что такое медиазапросы и для чего они нужны?**

**Медиазапросы** — это CSS-техника, позволяющая применять стили в зависимости от характеристик устройства, таких как ширина экрана, высота, ориентация и разрешение. Они используются для создания адаптивного дизайна, который подстраивается под разные устройства.

**16. Опишите orientation и его значения.**

**orientation** — это медиапараметр, который определяет ориентацию устройства:

* portrait: Вертикальная ориентация.
* landscape: Горизонтальная ориентация.

**17. Опишите viewport и его условия.**

**Viewport** — это область просмотра веб-страницы в браузере. Условия для viewport определяются в метатегах.

Это устанавливает ширину viewport равной ширине устройства и устанавливает начальный масштаб.

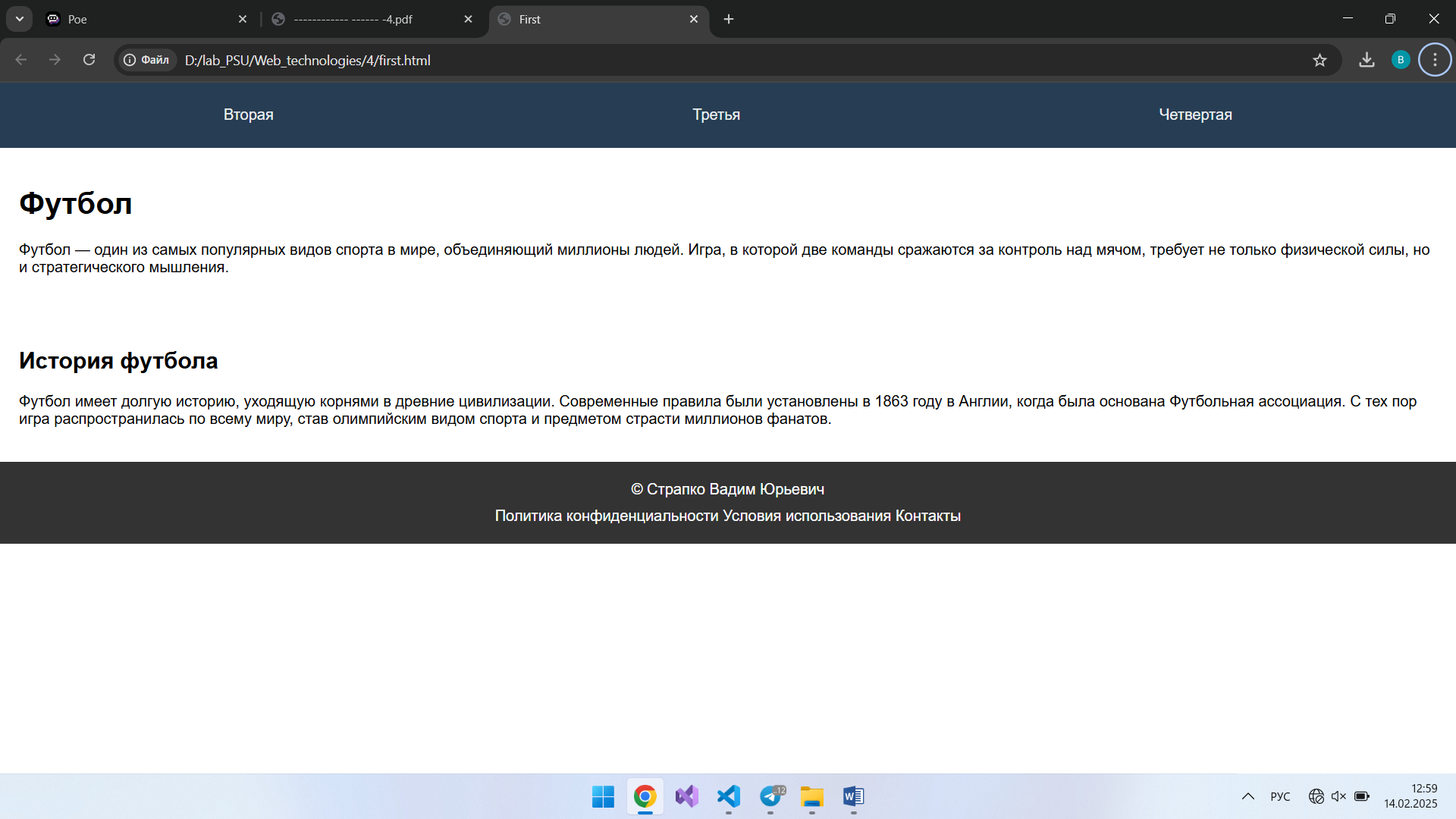
**18. Какие отличия Desktop First и Mobile First?**

* **Desktop First**: Разработка начинается с дизайна для настольных устройств, а затем адаптируется для мобильных. Обычно используются медиазапросы для изменения стилей.
* **Mobile First**: Разработка начинается с дизайна для мобильных устройств, а затем адаптируется для настольных. Используются медиазапросы для добавления стилей для более широких экранов.

**19. Каким образом можно комбинировать медиазапросы?**

Медиазапросы можно комбинировать с помощью логических операторов and, or и not. Также можно использовать несколько медиазапросов, разделяя их запятыми.

**4.Скриншоты результата**



**Рисунок 4.1 – Первая страница**

**Вывод:** в ходе данной лабораторной работы были изучены основы Flexbox для создания адаптивных макетов, совместимых с разными разрешениями экрана.