Федеральное государственное образовательное бюджетное учреждение высшего образования

«Финансовый университет при правительстве Российской Федерации»

(Финансовый университет)

Колледж информатики и программирования

МДК.05.02 Разработка кода информационных систем

Группа: 3ИСИП-723

Лабораторная работа 2

На тему: Взаимодействие с DOM

Выполнил:

_____ Е.Д. Сидоров

Принял:

Р.Р. Абзалимов _____

Москва

2026

Содержание

Практическая часть 4
Шаг 1: Создание структуры HTML 4
Шаг 2: Написание функций для увеличения и уменьшения счетчиков 5
Шаг 3: Завершение HTML-документа 6
Практическое задание с использованием всех методов: 7
Итоговый результат: 9
Контрольные вопросы 10
Самостоятельная работа 12
..... 15
Заключение 19

ВВЕДЕНИЕ

DOM (Document Object Model) — это программный интерфейс для веб-документов, который представляет структуру документа в виде дерева объектов. Каждый элемент HTML, текст, атрибуты и даже комментарии становятся объектами в этом дереве. DOM позволяет разработчикам взаимодействовать с содержимым, структурой и стилем документов.

Как DOM связан с HTML:

1. Структурное представление: когда браузер загружает HTML-документ, он создает DOMдерево, где каждый элемент HTML становится узлом. Например, теги , <p>, <h1> и другие представляют собой узлы в этом дереве.

2. Динамическое взаимодействие: С помощью JavaScript разработчики могут изменять содержимое и структуру HTML-документа, взаимодействуя с DOM. Это позволяет добавлять, удалять или изменять элементы на странице без необходимости перезагрузки.

3. События: DOM поддерживает обработку событий, что позволяет реагировать на действия пользователя, такие как клики, наведение мыши или ввод данных. Это делает веб страницы более интерактивными.

4. Стиль и представление: DOM также позволяет изменять CSS-стили элементов, что дает возможность динамически изменять внешний вид страницы в ответ на действия пользователя или другие события

Сравнение и важные замечания:

- Уникальность: getElementById() и querySelector() с селектором ID (#id) возвращают один элемент, так как id должен быть уникальным на странице.
- Коллекции: Методы getElementsByName, getElementsByClassName, getElementsByTagName и querySelectorAll возвращают коллекции. К их элементам можно

обращаться по индексу (например, elements[0]).

- "Живые" и "статические" коллекции:
  - Коллекции, возвращаемые методами getElementsByClassName, getElementsByTagName, getElementsByName являются "живыми" (live collections) — они автоматически обновляются при изменении структуры DOM.
  - Коллекция, возвращаемая querySelectorAll, является "статической" (staticcollection) — она не обновляется автоматически при изменении DOM.
- Гибкость: querySelector() и querySelectorAll() являются наиболее предпочтительными и гибкими методами, так как позволяют использовать всю мощь CSS-селекторов для точного нацеливания на нужные элементы.

Цель работы

Познакомиться с методом document.getElementById, который позволяет получить доступ к элементам HTML-документа по их идентификатору (ID) и изменять их содержимое. В этой лабораторной работе мы создадим простую веб-страницу с несколькими счетчиками, которые можно увеличивать и уменьшать с помощью кнопок.

Практическая часть

Шаг 1: Создание структуры HTML

Мы начнем с создания базовой структуры HTML-документа. Внутри <body> мы добавим несколько элементов, которые будут представлять счетчики и кнопки для их изменения.

```
<!DOCTYPE html>
<html>
<body>
<span id="counter1">0</span>
<button onclick="incrementCounter(1)">+</button>
<div id="counter2">0</div>
<button onclick="decrementCounter(2)">-</button>
<p id="counter3">0</p>
<button onclick="incrementCounter(3)">+</button>
<h1 id="counter4">0</h1>
<button onclick="decrementCounter(4)">-</button>
```

- Мы создаем 4 счетчика с уникальными идентификаторами (counter1, counter2, counter3, counter4).
- Каждому счетчику соответствует кнопка, которая будет увеличивать или уменьшать его значение.

Шаг 2: Написание функций для увеличения и уменьшения счетчиков

Теперь мы добавим JavaScript-код, который будет обрабатывать нажатия кнопок.

Мы создадим две функции: incrementCounter для увеличения значения счетчика и decrementCounter для его уменьшения.

```
<script>
function incrementCounter(counterNumber) {
let counterElement = document.getElementById(`counter${counterNumber}`); let currentValue = parseInt(counterElement.innerHTML);
counterElement.innerHTML = currentValue + 1;
}

function decrementCounter(counterNumber) {
let counterElement = document.getElementById(`counter${counterNumber}`); let currentValue = parseInt(counterElement.innerHTML);
counterElement.innerHTML = currentValue - 1;
```

Шаг 3: Завершение HTML-документа

Завершим наш HTML-документ, добавив закрывающий тег </body> и </html>.

```
</body>
</html>
```

Итоговый код:

```
html
<!DOCTYPE html>
<html>
<head> <title>Лабораторная работа №2:
  Счетчики</title>
  <style> body { font-family:
    Arial, sansserif; margin:
    20px;
      padding: 20px;
    }
    .counter { margin:
      10px; padding:
      10px; border:
      1px solid #ccc;
      display:
      inline-block;
    } button { margin:
    5px; padding: 5px
    15px; cursor:
    pointer;
    }
  </style>
</head>
<body>
  <h2>Счетчики с использованием
  document.getElementById()</h2>

  <div class="counter">
    <span>Счетчик 1: </span>
    <span id="counter1">0</span>
    <button onclick="incrementCounter(1)">+</button>
  </div>

  <div class="counter">
    <span>Счетчик 2: </span>
    <div id="counter2">0</div>
    <button onclick="decrementCounter(2)">-</button>
  </div>
```

```
  <div class="counter">
    <span>Счетчик 3: </span>
    <p id="counter3">0</p>
    <button onclick="incrementCounter(3)">+</button>
  </div>

  <div class="counter">
    <span>Счетчик 4: </span>
    <h3 id="counter4">0</h3>
    <button onclick="decrementCounter(4)">-</button>
  </div>

  <script>
  function incrementCounter(counterNumber) { let counterElement =
    document.getElementById(`counter${counterNumber}`); let currentValue
    = parseInt(counterElement.innerHTML); counterElement.innerHTML =
    currentValue + 1;
  }

  function decrementCounter(counterNumber) { let counterElement =
    document.getElementById(`counter${counterNumber}`); let currentValue
    = parseInt(counterElement.innerHTML); counterElement.innerHTML =
    currentValue - 1;
  }
  </script>
</body>
</html>
```

Практическое задание с использованием всех методов:

```
<!-- Пример getElementById -->
 <div class="demo-section">
 <h2>1. getElementById()</h2>
 <div id="main-title">Оригинальный заголовок</div>
 <button onclick="changeTitle()">Изменить заголовок</button>
</div>

 <!-- Пример getElementsByName -->
<div class="demo-section">
 <h2>2. getElementsByName()</h2>
 <input type="checkbox" name="skill" value="js"> JavaScript
 <input type="checkbox" name="skill" value="html"> HTML
 <input type="checkbox" name="skill" value="css"> CSS
 <button onclick="showSkills()">Показать выбранные</button>
 <p id="skills-result"></p>
 </div>
```



```

 <!-- Пример getElementsByClassName -->
 <div class="demo-section">
 <h2>3. getElementsByClassName()</h2>
 <p class="important">Важный пункт 1</p>
 <p class="important">Важный пункт 2</p>
 <p>Обычный пункт</p>
 <p class="important">Важный пункт 3</p>
 <button onclick="highlightImportant()">Подсветить важное</button>
</div>

 <!-- Пример getElementsByTagName -->
 <div class="demo-section">
 <h2>4. getElementsByTagName()</h2>
 <div>Первое сообщение</div>
 <div>Второе сообщение</div>
<div>Третье сообщение</div>
 <button onclick="formatAllDivs()">Форматировать все div</button>
</div>

 <!-- Пример querySelector -->
 <div class="demo-section">
 <h2>5. querySelector()</h2>
 <div class="container">
 <p class="first">Первый абзац</p>
 <p>Второй абзац</p>
 <p class="first">Третий абзац</p>
 </div>
 <button onclick="selectFirst()">Выбрать первый</button>
</div>

 <!-- Пример querySelectorAll -->
 <div class="demo-section">
 <h2>6. querySelectorAll()</h2>
 <ul id="products">
```

```
<li class="product">Ноутбук</li
```

Итоговый результат:

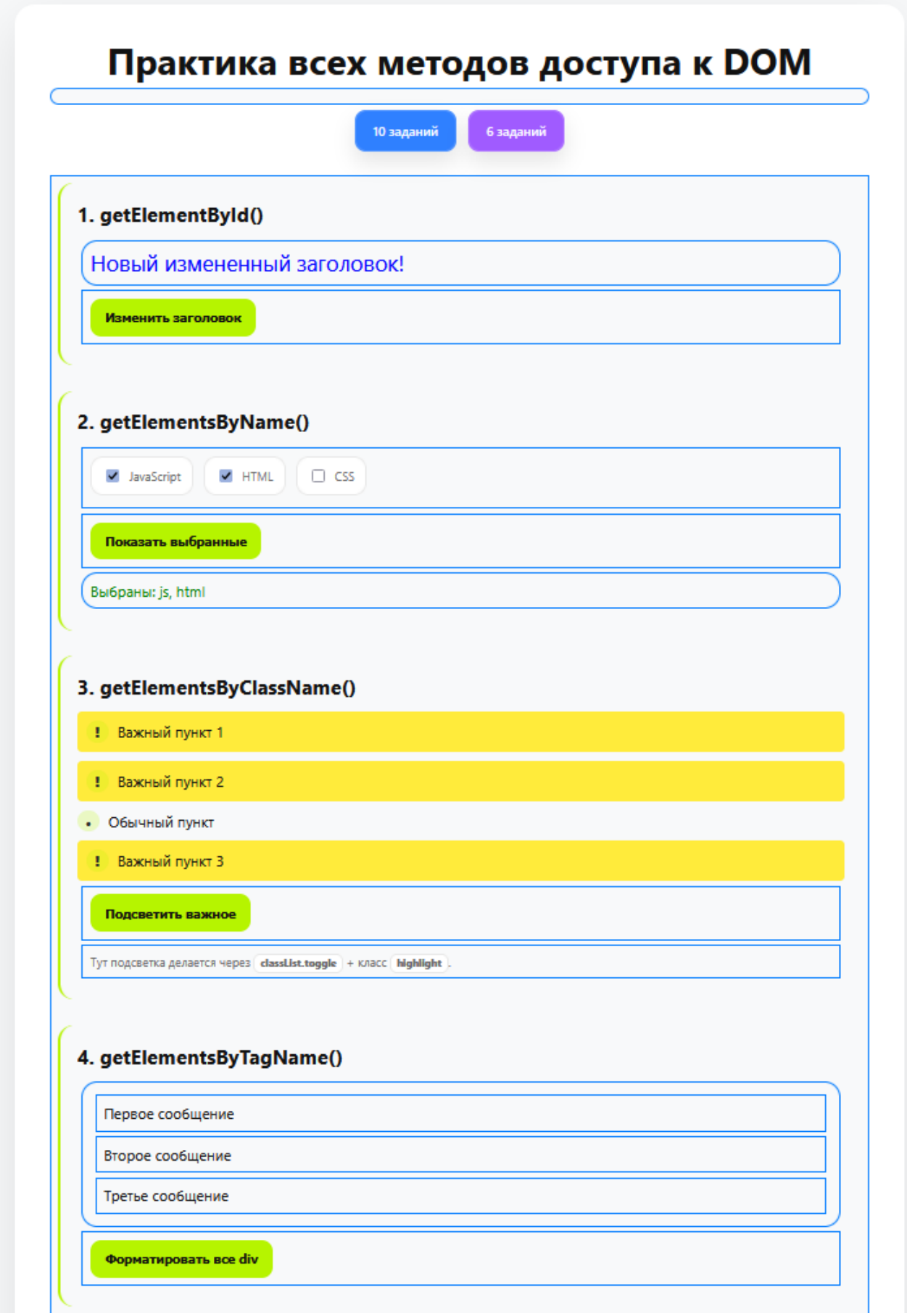


Рисунок 1 Итоговый результат (часть 1)

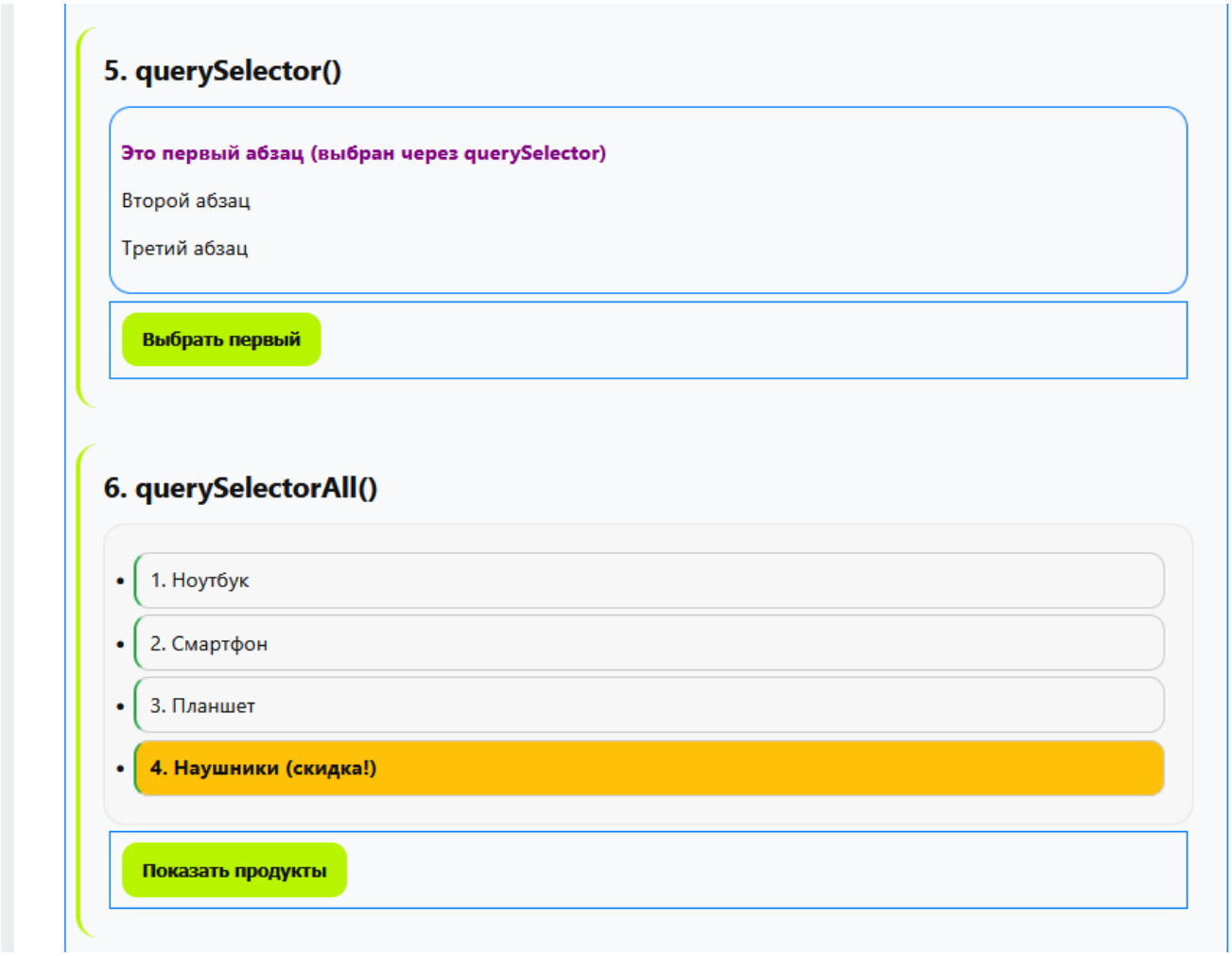


Рисунок 2 Итоговый результат (часть 2)

Контрольные вопросы

1. DOM – объектная модель документа. Это представление HTML в виде дерева объектов, которое JavaScript может изменять.

2. Получить элемент по ID:

```javascript
let element = document.getElementById("myId");
```

3. Событие – действие на веб-странице (например, клик, нажатие клавиши, загрузка). Примеры: `click`, `keydown`, `load`.

4. Добавить обработчик события на кнопку:

```html
<button id="myButton">Нажми</button>
```

```
  <script>
    document.getElementById("myButton").addEventListener("click", function() {
alert("Кнопка нажата!");
    });
  </script>
  ```
```

5. Изменить текст элемента:*

```
  ```javascript
  document.getElementById("myElement").textContent = "Новый текст";
  ```
```

6. Массивы – структуры для хранения наборов данных. Используются для работы с DOM, например, через `querySelectorAll`, который возвращает коллекцию элементов.

7. Удалить элемент из DOM:

```
  ```javascript
  let element = document.getElementById("myElement");
element.parentNode.removeChild(element);
  // Или современный способ:
element.remove();
  ```
```

8. Разница между `getElementById` и `querySelector`:

- `getElementById` находит элемент только по `id`.
- `querySelector` использует CSS-селекторы, работает с любыми элементами.

9. Изменить стиль элемента:

```
  ```javascript
  document.getElementById("myElement").style.color = "red";
  ```
```

10. Добавить новый элемент в DOM:

```
  ```javascript
   let newElement = document.createElement("p");
newElement.textContent = "Новый параграф";
document.body.appendChild(newElement);
  ```
```

Самостоятельная работа

Задание 1: getElementById()
```

```
<section>
      <h2>Задание 1: getElementById()</h2>
      <h1 id="my-title">Привет, мир!</h1>
      <button onclick="changeText()">Изменить текст</button>
   </section>
//////////////////////////////
function changeText() {
         let element = document.getElementById('my-title');
element.textContent = 'Привет, JavaScript!';
element.style.color = 'red';        }
```

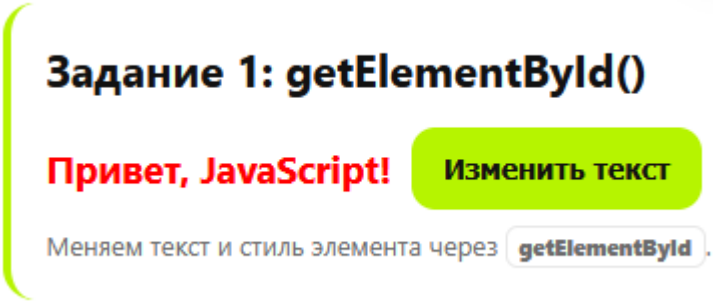


Рисунок 3 getElementByld()

Задание 2: getElementsByName()

```
<section>
      <h2>Задание 2: getElementsByName()</h2>
      <input type="checkbox" name="fruit"> Яблоко<br>
<input type="checkbox" name="fruit"> Банан<br>
      <input type="checkbox" name="fruit"> Апельсин<br>
      <button onclick="countFruits()">Сколько выбрано?</button>
<p id="result"></p>
   </section>
//////////////////////////////
// Задание 2
      function countFruits() {
         let checkboxes = document.getElementsByName('fruit');
let count = 0;

         for (let checkbox of checkboxes) {
if (checkbox.checked) {
               count++;
            }
         }

         document.getElementById('result').textContent = `Выбрано фруктов: ${count}`;
```

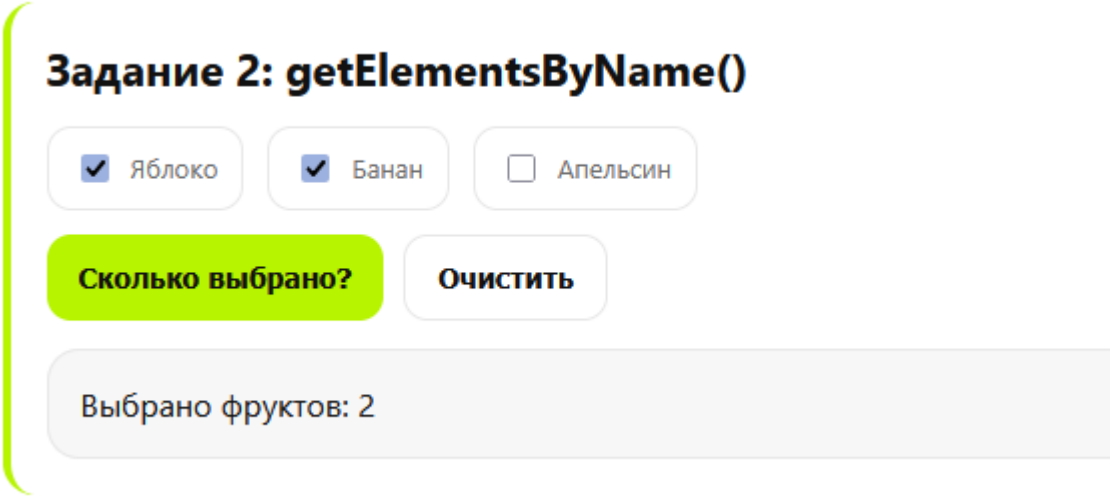


Рисунок 4 getElementsByName()

Задание 3: getElementsByClassName()

```
<section>
      <h2>Задание 3: getElementsByClassName()</h2>
      <p class="important">Первая важная новость</p>
      <p class="important">Вторая важная новость</p>
      <p>Обычная новость</p>
      <button onclick="highlightImportant()">Выделить важное</button>
</section>
////////////////////////////////
function highlightImportant() {
        let elements = document.getElementsByClassName('important');

        for (let element of elements) {
           element.style.backgroundColor = 'yellow';
element.style.fontWeight = 'bold';
element.style.padding = '10px';
        }
      }
```

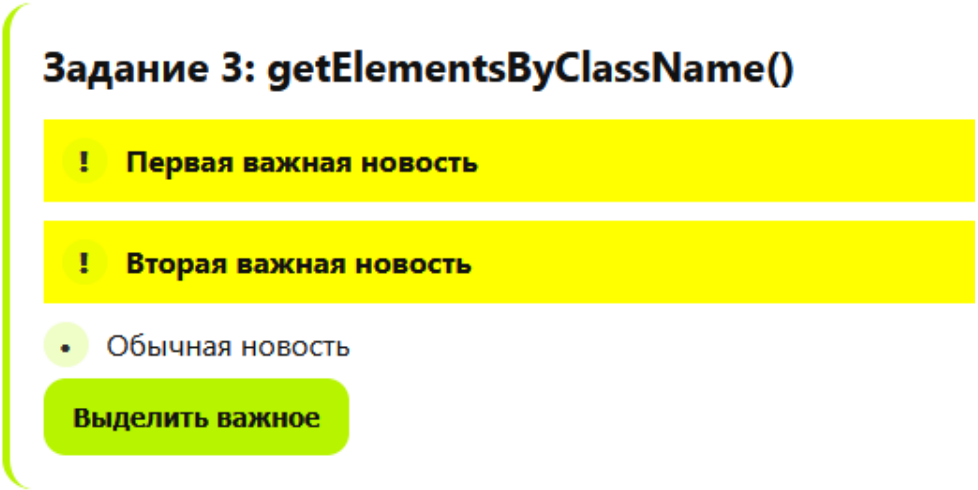


Рисунок 5 getElementsByClassName()

Задание 4: getElementsByTagName()

```
// Задание 4
   <section>
      <h2>Задание 4: getElementsByTagName()</h2>
      <button>Кнопка 1</button>
      <button>Кнопка 2</button>
      <button>Кнопка 3</button>
      <button onclick="styleAllButtons()">Стилизовать все кнопки</button>
   </section>    function styleAllButtons() {
         let buttons = document.getElementsByTagName('button');
   ////////////////////////////


         for (let button of buttons) {
            button.style.backgroundColor = 'green';
            button.style.color = 'white';
button.style.padding = '10px';
button.style.margin = '10px';
         }
      }
```

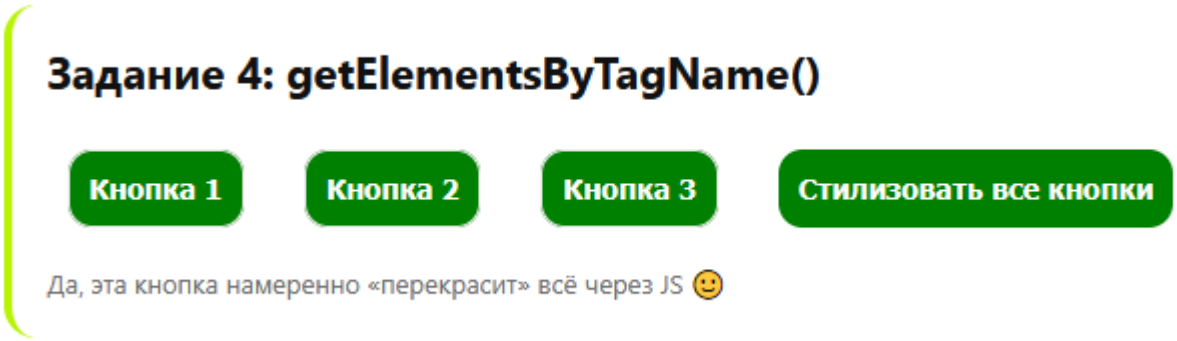


Рисунок 6 getElementsByTagName()

Задание 5: querySelector()

```
<section>
      <h2>Задание 5: querySelector()</h2>
      <ul>
         <li class="item">Пункт 1</li>
         <li class="item">Пункт 2</li>
         <li class="item special">Особый пункт</li>
</ul>
      <button onclick="findFirst()">Найти первый элемент</button>
   </section>
////////////////////////////////
function findFirst() {
         let element = document.querySelector('.item');
if (element) {
            element.textContent = 'Первый найденный!';
            element.style.color = 'blue';
element.style.fontWeight = 'bold';
         }
      }
```

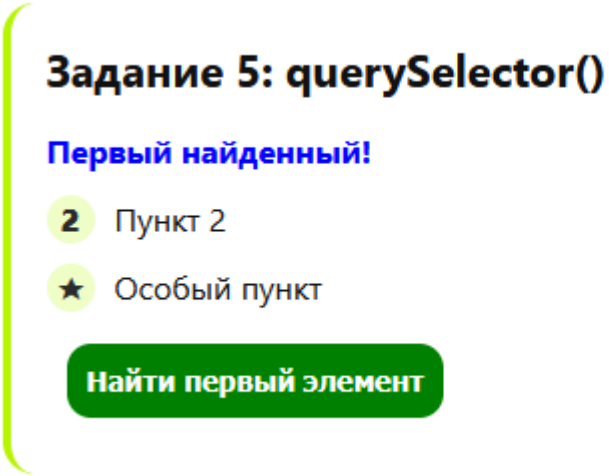


Рисунок 7 querySelector()

Задание 6: querySelectorAll()

```
// Задание 6
        function styleAllBoxes() {
            let boxes = document.querySelectorAll('.box');

            boxes.forEach(box => {
                box.style.border = '2px solid black';
box.style.padding = '15px';
box.style.margin = '10px';
box.style.display = 'inline-block';
box.style.backgroundColor = '#e0e0e0';
            });
        }
////////////////////////////////////////////////
<section>
        <h2>Задание 6: querySelectorAll()</h2>
        <div class="box">Коробка 1</div>
        <div class="box">Коробка 2</div>
        <div class="box">Коробка 3</div>
        <button onclick="styleAllBoxes()">Оформить все коробки</button>
    </section>
```

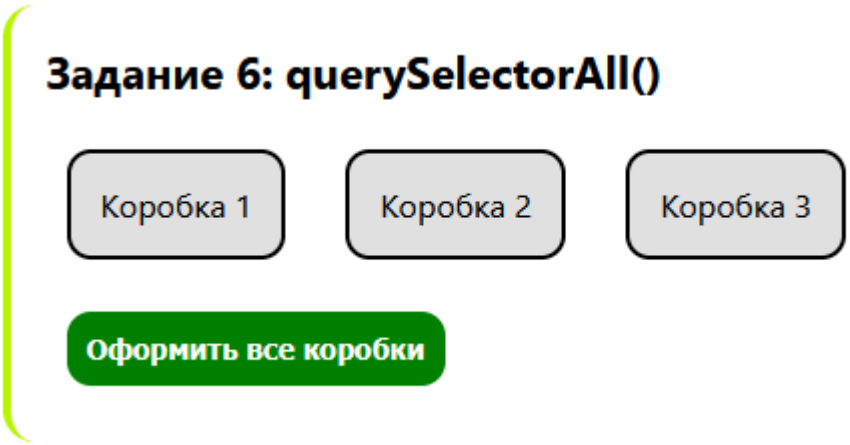


Рисунок 8 querySelectorAll()

Задание 7: изменить цвет фона страницы

```
<section>
        <h2>Задание 7: Изменить цвет фона страницы</h2>
        <button onclick="changeToRed()">Красный</button>
        <button onclick="changeToBlue()">Синий</button>
        <button onclick="changeToGreen()">Зеленый</button>
        <button onclick="resetBackground()">Сбросить фон</button>
    </section>
//////////////////////////////////////////
```

```
function changeToRed() {
        document.body.style.backgroundColor = 'red';
    }
    function changeToBlue() {
        document.body.style.backgroundColor = 'blue';
document.body.style.color = 'white';
    }
    function changeToGreen() {
        document.body.style.backgroundColor = 'green';
    }
    function resetBackground() {
        document.body.style.backgroundColor = '#f5f5f5';
        document.body.style.color = 'black';
    }
```

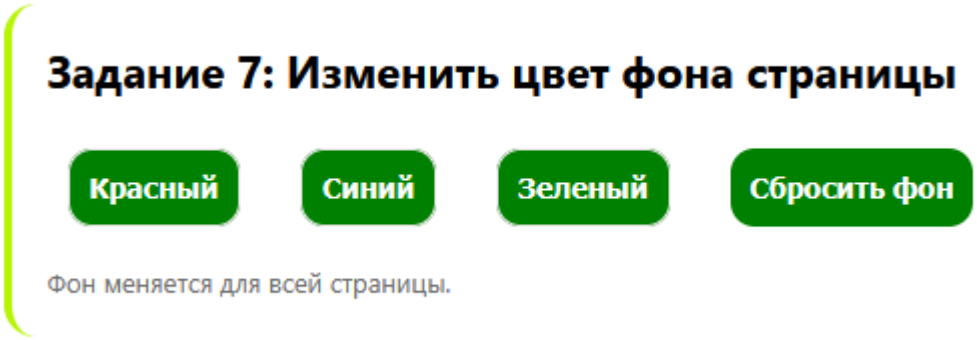


Рисунок 9 Изменение фона страницы

Задание 8: показать/скрыть элемент

```
function showText() {          document.getElementById('secret-text').style.display = 'block';
    }
    function hideText() {
        document.getElementById('secret-text').style.display = 'none';
    }
////////////////////////////// function showText() {
        document.getElementById('secret-text').style.display = 'block';
    }
    function hideText() {
        document.getElementById('secret-text').style.display = 'none';
    }
```

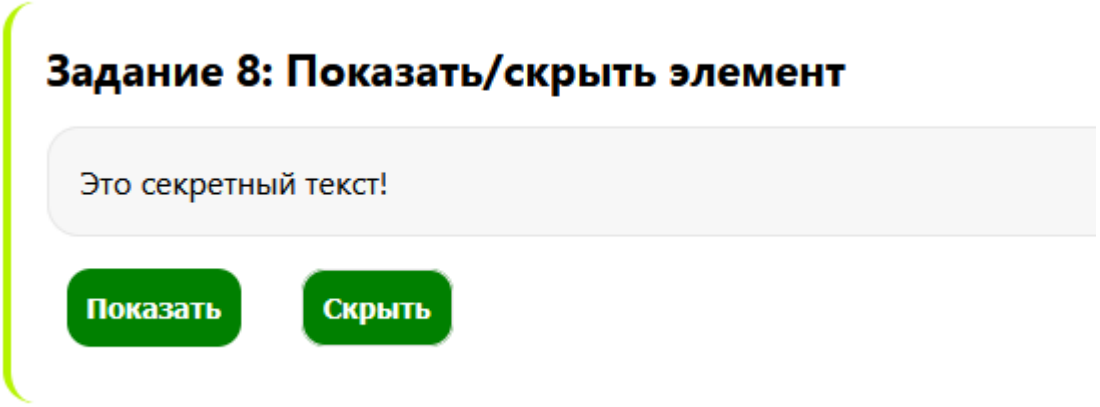


Рисунок 11 показать/скрыть элемент

Задание 9: изменить картинку

```
<section>        <h2>Задание 9: Изменить
картинку</h2>
      <img id="my-image" src="https://www.betalt.lt/wp-content/uploads/2025/01/5179-800x600.jpg"
width="200" alt="Изображение">
      <br>
      <button onclick="changeToDog()">Собака</button>
      <button onclick="changeToCat()">Кошка</button>
      <button onclick="resetImage()">Исходное</button>
   </section>
//////////////////////////////////////////////////
function changeToDog() {
         document.getElementById('my-image').src = 'https://img.freepik.com/free-photo/happy-
dogportrait-looking-front_1194-589243.jpg?semt=ais_hybrid&w=740';
      }
      function changeToCat() {
         document.getElementById('my-image').src =
'https://avatars.mds.yandex.net/i?id=9ce2516902a24b535c209c8189d3bd15_l-8497409-
imagesthumbs&n=13';
      }
      function resetImage() {
         document.getElementById('my-image').src =
'https://www.betalt.lt/wpcontent/uploads/2025/01/5179-800x600.jpg';
      }
```

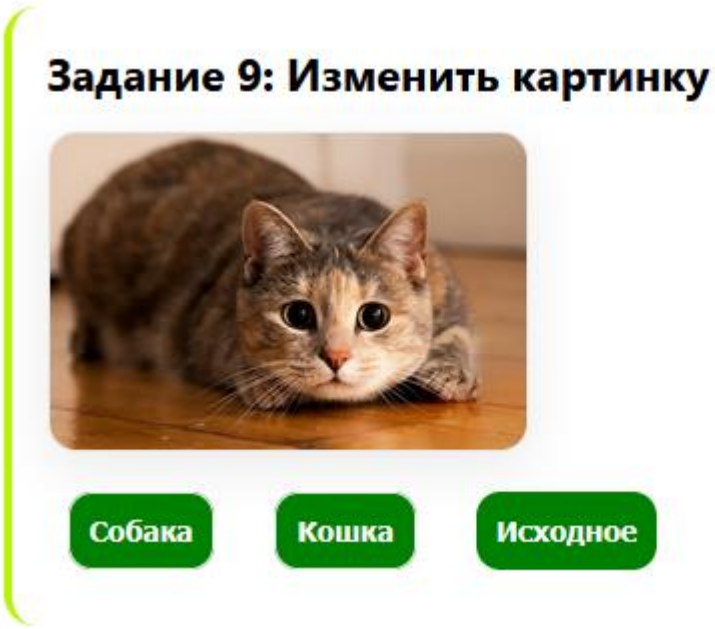


Рисунок 10 изменить картинку

Задание 10: Простой калькулятор

```
<section>
        <h2>Задание 10: Простой калькулятор</h2>
        <p id="counter" style="font-size: 24px; font-weight: bold;">0</p>
        <button onclick="increase()">+1</button>
        <button onclick="decrease()">-1</button>
        <button onclick="resetCounter()">Сбросить</button>
    </section>
//////////////////////////
function increase() {
          let counter = document.getElementById('counter');
let value = parseInt(counter.textContent);
counter.textContent = value + 1;
        }
        function decrease() {
          let counter = document.getElementById('counter');
let value = parseInt(counter.textContent);
counter.textContent = value - 1;
        }
        function resetCounter() {
          document.getElementById('counter').textContent = '0';
        }
```

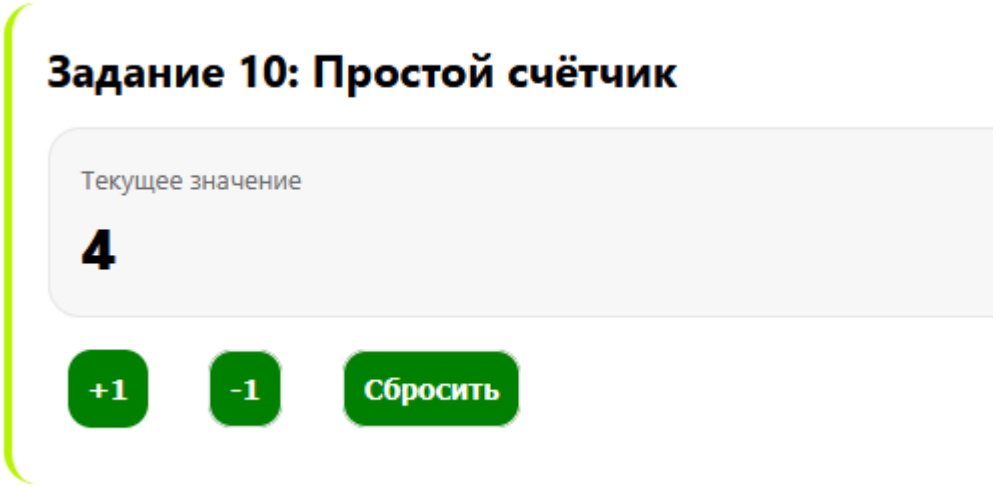


Рисунок 12 простой калькулятор

Заключение

В ходе выполнения лабораторной работы были успешно освоены основные методы работы с объектной моделью документа (DOM) в JavaScript. Практические задания позволили закрепить теоретические знания о структуре DOM и способах взаимодействия с элементами веб-страницы. Были изучены и применены на практике различные методы доступа к элементам DOM: getElementById для поиска по уникальному идентификатору, getElementsByName для работы с группами элементов

форм, getElementsByClassName для манипуляции элементами с определенными классами, getElementsByTagName для выборки по типу тега, а также современные методы querySelector и querySelectorAll, предоставляющие гибкость CSS-селекторов. Каждый метод продемонстрировал свои преимущества и области применения, что позволило понять их различия и выбрать оптимальный подход для различных сценариев.

Особое внимание было уделено обработке событий, что является ключевым аспектом создания интерактивных веб-приложений. Были реализованы обработчики для различных событий: кликов мыши, изменения состояния элементов форм и других пользовательских взаимодействий. Практика показала, что правильная работа с событиями позволяет создавать отзывчивые и удобные интерфейсы.

Важным результатом работы стало освоение методов динамического изменения содержимого страницы: модификации текстового содержимого через textContent и innerHTML, изменения атрибутов элементов, управления CSS-стилями через свойство style, а также добавления и удаления элементов из DOM-дерева. Эти навыки необходимы для создания динамических веб-приложений, которые реагируют на действия пользователя без перезагрузки страницы. Была реализована система навигации между различными заданиями лабораторной работы, что продемонстрировало возможности создания связанных многостраничных приложений. Все задачи были выполнены последовательно, от простых к более сложным, что позволило постепенно наращивать уровень владения технологиями.

В заключение можно констатировать, что лабораторная работа достигла поставленных целей: были сформированы практические навыки работы с DOM, освоены различные методы доступа и манипуляции элементами, изучены принципы обработки событий. Полученные знания составляют фундамент для дальнейшего изучения веб-разработки и создания современных интерактивных веб-приложений. Работа продемонстрировала, что понимание DOM и умение эффективно с ним работать являются необходимыми компетенциями для frontend-разработчика.