

## **Разработка интерактивного веб-опросника на основе HTML и JavaScript для анализа социологических данных**

*Голубева Евгения Павловна*

*Приамурский государственный университет имени Шолом-Алейхема*

*Студент*

### **Аннотация**

Цель данной статьи – создание интерактивного веб-опросника для сбора и анализа социологических данных с использованием современных веб-технологий. Для создания веб-опросника использовались язык программирования JavaScript, язык разметки HTML и CSS для оформления дизайна. В результате работы был создан веб-опросник, который представляет собой удобный и эффективный инструмент для проведения опросов, анализа собранных данных и их наглядного представления.

**Ключевые слова:** HTML, JavaScript, CSS, веб-опросник.

### **Development of an interactive web-based HTML and JavaScript questionnaire for analyzing sociological and political science data**

*Golubeva Evgeniya Pavlovna*

*Sholom-Aleichem Priamursky State University*

*Student*

### **Abstract**

The purpose of this article is to create an interactive web-based questionnaire for collecting and analyzing sociological data using modern web technologies. The JavaScript programming language, HTML markup language, and CSS were used to create the web questionnaire. As a result of the work, a web-based questionnaire was created, which is a convenient and effective tool for conducting surveys, analyzing the collected data and visualizing them.

**Keywords:** HTML, JavaScript, CSS, web questionnaire.

## **1 Введение**

### **1.1 Актуальность**

Разработка интерактивного веб-опросника на основе HTML и JavaScript для анализа социологических данных является крайне актуальной в современном обществе, где информация и мнение граждан играют ключевую роль в принятии решений. В условиях быстрого развития технологий и увеличения объемов данных, собранных через онлайн-опросы, необходимость в эффективных инструментах для их анализа становится все более очевидной.

Интерактивные веб-опросники позволяют не только собирать данные, но и обеспечивают удобный интерфейс для респондентов, что способствует

повышению качества и точности получаемых ответов. Использование современных веб-технологий, таких как HTML и JavaScript, позволяет создавать динамичные и адаптивные решения, которые могут быть легко интегрированы в различные исследовательские проекты.

Кроме того, актуальность данной разработки также обусловлена растущим интересом к социологическим исследованиям, которые помогают понять общественные настроения и выявить ключевые проблемы. Веб-опросники становятся важным инструментом для исследователей, позволяя им быстро и эффективно получать информацию, необходимую для анализа и принятия обоснованных решений.

## **1.2 Обзор исследований**

М.М. Насретдинова в данной статье рассматривала особенности организации онлайн опроса с помощью онлайн панели [1]. Представили результаты проектной документации и прототипа автоматизированной системы для проведения социологических опросов Е.Д. Рогожников, О.Ф. Абрамова, Д.Н. Лясин [2]. Е.Ю. Мальцева и Е.В. Молчанова в статье рассматривали особенности применения информационных технологий на этапах социологического исследования [3]. Продемонстрировали методику разработки web-анкет для интернет-опросов, позволяющая сформировать анкету для оперативного сбора и анализа информации о квалификации преподавателей Г.Д. Волкова, О.В. Новоселова, В.В. Попов, А.И. Башмаков, Д.Н. Жедяевский. [4]. С.Д. Отбоева статью посвятила особенностям применения информационных технологий на разных этапах социологического исследования. [5].

## **1.3 Цель исследования**

Цель исследования - создание интерактивного веб-опросника для сбора и анализа социологических данных с использованием языка программирования JavaScript, языка гипертекстовой разметки HTML и языка описания внешнего вида документа CSS для оформления дизайна.

## **2 Материалы и методы**

Для создания приложения используется язык программирования JavaScript, язык гипертекстовой разметки HTML, язык описания внешнего вида документа CSS. В качестве среды разработки используется текстовым редактор Notepad++, оптимизированным для языков программирования.

## **3 Результаты и обсуждения**

Перед началом работы требуется скачать Notepad++ с официального сайта и установить.

Создадим файл index.html. Для этого необходимо открыть Notepad++, и для того, чтобы создать файл в меню необходимо выбрать файл -> новый (или нажмите Ctrl + N). В меню выбираем файл -> Сохранить как... В поле "Имя файла" введите название файла index с расширением .html, и сохраняем.

Для начала реализуем код с базовой структуры HTML-документа (рис.1). Внутри тега <head> задаются метаданные, такие как кодировка (UTF-8) и заголовок страницы. В разделе <style> определены CSS-стили для оформления страницы: шрифты, отступы, выравнивание заголовка по центру, а также скрытие формы опроса и контейнера для результатов по умолчанию. В теле документа (<body>) размещены элементы интерфейса: заголовок, кнопки для начала опроса, просмотра результатов, скачивания данных и сброса результатов. Контейнеры surveyForm и resultsDisplay используются для динамического отображения формы и результатов соответственно. Все функциональные элементы изначально скрыты, кроме кнопки "Пройти опрос". Для того, чтобы просмотреть файл index.html, необходимо нажать правой кнопкой мыши на файл, и выбрать открыть с помощью, и выбираем браузер.

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Опрос по социологии/политологии</title>
  <style>
    /* CSS Styles */
    body {
      font-family: Arial, sans-serif;
      margin: 20px;
    }
    h1 {
      text-align: center;
    }
    #surveyForm {
      margin-top: 20px;
      display: none;
    }
    .question {
      margin-bottom: 10px;
    }
    button {
      margin-top: 10px;
      padding: 10px 20px;
      cursor: pointer;
    }
    #resultsDisplay {
      margin-top: 20px;
      border: 1px solid #ccc;
      padding: 10px;
      display: none;
    }
  </style>
</head>
<body>
  <h1>Опрос по социологии/политологии</h1>
  <button id="takeSurveyBtn">Пройти опрос</button>
  <button id="viewResultsBtn" style="display:none;">Просмотр
результатов</button>
  <div id="surveyForm"></div>
  <div id="resultsDisplay"></div>
```

```

<button id="downloadCSVBtn" style="display:none;">Скачать результаты в CSV</button>
<button id="downloadJSONBtn" style="display:none;">Скачать результаты в JSON</button>
<button id="resetCountsBtn" style="display:none;">Сбросить результаты</button>

<script>
    // JavaScript код будет здесь
</script>
</body>
</html>

```



Рисунок 1. Базовая структура HTML-документа

Далее необходимо реализовать весь функционал опроса. Для этого в теге `<script>` последовательно вставляем коды функционала веб опросника.

В JavaScript-коде создаем объект `questionMap`, который содержит вопросы опроса. Каждый вопрос имеет уникальный ключ (например, `q1`, `q2`) и текстовое описание. Этот объект служит для удобного хранения и доступа к вопросам при создании формы. Вопросы охватывают различные темы, такие как доверие к правительству, роль СМИ, участие в политической жизни и другие. Использование объекта позволяет легко добавлять или изменять вопросы без необходимости переписывать логику отображения формы. В теге `<script>` прописываем следующий код:

```

const questionMap = {
  q1: "Вы доверяете правительству?",
  q2: "Считаете ли вы, что СМИ играют важную роль в обществе?",
  q3: "Вы активно участвуете в политической жизни?",
  q4: "Считаете ли вы, что молодежь интересуется политикой?",
  q5: "Вы считаете, что свобода слова важна в демократическом обществе?",
  q6: "Вы считаете, что коррупция является серьезной проблемой в стране?",
  q7: "Вы верите в возможность изменений в политике?",
  q8: "Вы считаете, что социальные сети влияют на общественное мнение?",
  q9: "Вы считаете, что независимость судебной системы важна?",
  q10: "Вы довольны уровнем образования в стране?"
};

```

Результаты опроса хранятся в объекте `results`, где каждый вопрос представлен в виде вложенного объекта с двумя свойствами: `да` и `нет`. Эти свойства используются для подсчета количества ответов. Далее после объекта `questionMap` создаем следующий объект `results`, который будет хранить количество ответов "да" и "нет" для каждого вопроса:

```
let results = {
  q1: { да: 0, нет: 0 },
  q2: { да: 0, нет: 0 },
  q3: { да: 0, нет: 0 },
  q4: { да: 0, нет: 0 },
  q5: { да: 0, нет: 0 },
  q6: { да: 0, нет: 0 },
  q7: { да: 0, нет: 0 },
  q8: { да: 0, нет: 0 },
  q9: { да: 0, нет: 0 },
  q10: { да: 0, нет: 0 }
};
```

Функция `displaySurveyForm` отвечает за отображение формы опроса. Она очищает контейнер `surveyForm`, чтобы избежать дублирования, и делает его видимым. Затем в цикле создаются элементы формы: для каждого вопроса добавляется текст вопроса и два варианта ответа ("да" и "нет") в виде радиокнопок. После создания всех вопросов добавляется кнопка "Отправить", которая вызывает функцию `submitSurvey` при нажатии. Эта функция динамически создает интерфейс опроса, обеспечивая гибкость и удобство.

Создаем функцию `displaySurveyForm` после объекта `results`:

```
function displaySurveyForm() {
  const formContainer = document.getElementById('surveyForm');
  formContainer.innerHTML = '';
  formContainer.style.display = 'block';

  for (let i = 1; i <= 10; i++) {
    const div = document.createElement('div');
    div.className = 'question';
    const questionKey = 'q' + i;
    div.innerHTML = `
      ${questionMap[questionKey]}<br>
      <input type="radio" name="${questionKey}" value="да"
id="${questionKey}_да">
      <label for="${questionKey}_да">Да</label>
      <input type="radio" name="${questionKey}" value="нет"
id="${questionKey}_нет">
      <label for="${questionKey}_нет">Нет</label><br>`;
    formContainer.appendChild(div);
  }

  const submitBtn = document.createElement('button');
  submitBtn.textContent = 'Отправить';
  submitBtn.addEventListener('click', submitSurvey);
  formContainer.appendChild(submitBtn);
}
```

Далее прописываем код для обработчика события для кнопки "Пройти опрос":

```
document.getElementById('takeSurveyBtn').addEventListener('click',
function() {
  displaySurveyForm();
  document.getElementById('resultsDisplay').style.display = 'none'; //
Скрыть результаты
});
```

После того, как прописали код для кнопки "Пройти опрос", обновляем страницу файла `index.html` в браузере, и проверяем функционал кнопки

«Пройти опрос». После нажатия на кнопку появляется страница с опросом (см.рис.2).

**Опрос по социологии/политологии**

Пройти опрос

Вы доверяете правительству?  
☐ Да ☐ Нет

Считаете ли вы, что СМИ играют важную роль в обществе?  
☐ Да ☐ Нет

Вы активно участвуете в политической жизни?  
☐ Да ☐ Нет

Считаете ли вы, что молодежь интересуется политикой?  
☐ Да ☐ Нет

Вы считаете, что свобода слова важна в демократическом обществе?  
☐ Да ☐ Нет

Вы считаете, что коррупция является серьезной проблемой в стране?  
☐ Да ☐ Нет

Вы верите в возможность изменений в политике?  
☐ Да ☐ Нет

Вы считаете, что социальные сети влияют на общественное мнение?  
☐ Да ☐ Нет

Вы считаете, что независимость судебной системы важна?  
☐ Да ☐ Нет

Вы довольны уровнем образования в стране?  
☐ Да ☐ Нет

Отправить

Рисунок 2. Страница с опросом

Создаем функцию `submitSurvey`. Функция `submitSurvey` обрабатывает отправку формы. Она предотвращает стандартное поведение формы (перезагрузку страницы) с помощью `event.preventDefault()`. Затем проверяется, ответил ли пользователь на все вопросы. Если хотя бы один вопрос остался без ответа, выводится предупреждение. Если все вопросы заполнены, результаты обновляются: для каждого вопроса увеличивается счетчик выбранного ответа ("да" или "нет"). Обновленные результаты сохраняются в `localStorage`, после чего форма скрывается, а кнопка "Просмотр результатов" становится видимой. Пользователь получает уведомление об успешном завершении опроса. После `document.getElementById` прописываем код функции `submitSurvey`:

```
function submitSurvey(event) {
    event.preventDefault(); // Предотвращаем перезагрузку страницы

    // Проверяем, что все вопросы заполнены
    let allAnswered = true;
    for (let i = 1; i <= 10; i++) {
        const questionKey = 'q' + i;
        const selected = document.querySelector(`input[name="${questionKey}"]:checked`);
        if (!selected) {
            allAnswered = false;
            break;
        }
    }

    if (!allAnswered) {
        alert('Пожалуйста, ответьте на все вопросы.');
```

=

```
        return;
    }

    // Обновляем результаты
```

```

        for (let i = 1; i <= 10; i++) {
            const questionKey = 'q' + i;
            const selected = document.querySelector(`input[name="${questionKey}"]:checked`).value;
            results[questionKey][selected]++;
        }

        // Сохраняем результаты в localStorage
        localStorage.setItem('surveyResults', JSON.stringify(results));

        // Скрываем форму и показываем кнопку "Просмотр результатов"
        document.getElementById('surveyForm').style.display = 'none';
        document.getElementById('viewResultsBtn').style.display = 'inline-block';
        alert('Спасибо за участие в опросе!');
    }

```

Далее создаем функцию `showResultsAndDownloads`, которая отображает результаты опроса и кнопки для скачивания данных (см.рис.3). Функция делает видимыми контейнер `resultsDisplay` и кнопки "Скачать результаты в CSV", "Скачать результаты в JSON" и "Сбросить результаты". Затем вызывается функция `displayResults`, которая выводит текущие результаты в контейнер `resultsDisplay`. Эта функция обеспечивает удобный доступ к результатам и дополнительным функциям после завершения опроса. Добавляем код функции `showResultsAndDownloads` после функции `submitSurvey`:

```

function showResultsAndDownloads() {
    document.getElementById('resultsDisplay').style.display = 'block';
    document.getElementById('downloadCSVBtn').style.display = 'inline-block';
    document.getElementById('downloadJSONBtn').style.display = 'inline-block';
    document.getElementById('resetCountsBtn').style.display = 'inline-block';
    displayResults();
}

```

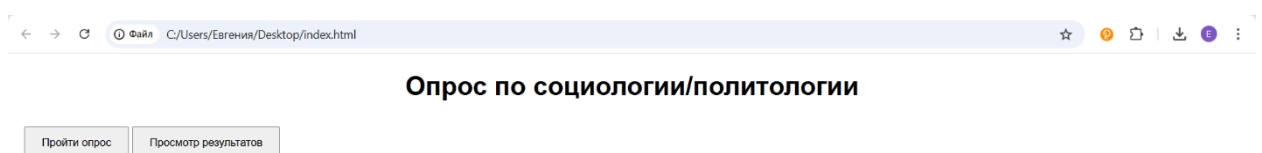


Рисунок 3. Страница после прохождения опроса

Далее создаем код обработчика события для кнопки "Просмотр результатов":

```

document.getElementById('viewResultsBtn').addEventListener('click',
function() {
    showResultsAndDownloads();
});

```

Добавляем функцию `displayResults`, которая отвечает за отображение текущих результатов опроса на странице (см.рис.4). Функция получает

контейнер `resultsDisplay`, очищает его и добавляет заголовок "Текущие результаты:". Затем в цикле перебираются все вопросы (от `q1` до `q10`), и для каждого вопроса формируется строка с текстом вопроса и количеством ответов "да" и "нет", которая добавляется в контейнер. Это позволяет пользователю видеть итоги опроса в удобном и читаемом формате. После кода `document.getElementById` прописываем код функции `displayResults`:

```
function displayResults() {
    const resultsDisplay =
document.getElementById('resultsDisplay');
    resultsDisplay.innerHTML = '<h2>Текущие результаты:</h2>';
    for (let i = 1; i <= 10; i++) {
        const q = `q${i}`;
        resultsDisplay.innerHTML += `<p>${questionMap[q]}: Да -
${results[q]['да']} | Нет - ${results[q]['нет']}</p>`;
    }
}
```

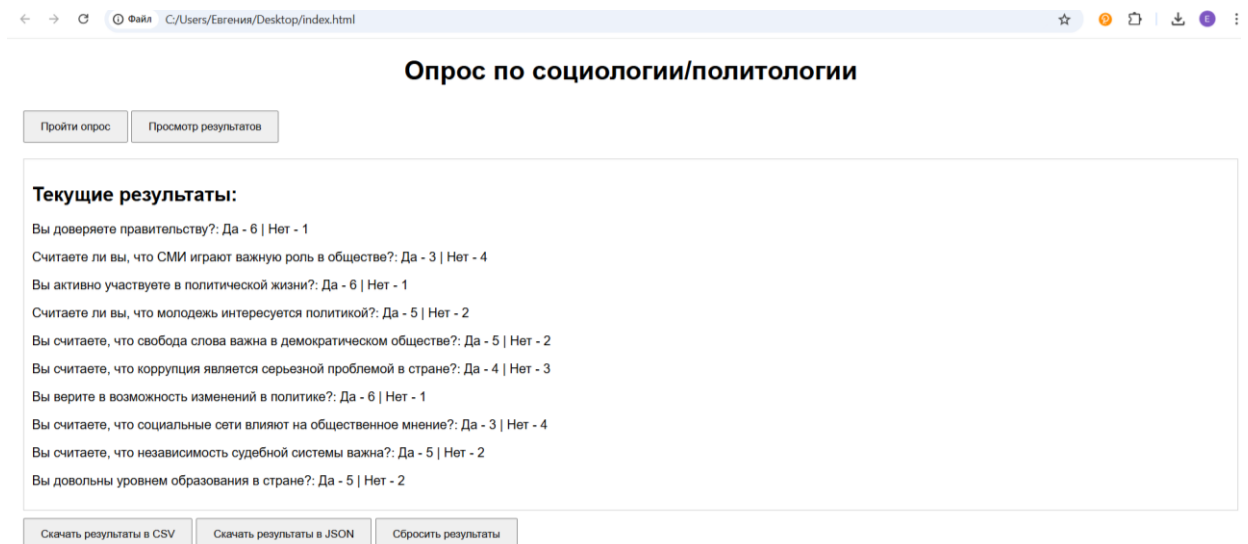


Рисунок 4. Просмотр результатов

Также необходимо добавить функцию `getResultsForDownload`. Данная функция подготавливает данные для скачивания, создавая объект, где ключами являются тексты вопросов, а значениями — количество ответов "да" и "нет". Этот объект используется функциями `downloadCSV` и `downloadJSON` для экспорта результатов в форматы CSV и JSON, обеспечивая возможность скачивания данных пользователем. Обе функции работают вместе, предоставляя удобный интерфейс для просмотра и сохранения результатов опроса. Прописываем код функции `getResultsForDownload`:

```
function getResultsForDownload() {
    const resultsForDownload = {};
    for (let i = 1; i <= 10; i++) {
        const questionKey = 'q' + i;
        resultsForDownload[questionMap[questionKey]] = {
            да: results[questionKey].да,
            нет: results[questionKey].нет
        };
    }
}
```

```

    };
  }
  return resultsForDownload;
}

```

Далее создадим функции `downloadCSV` и `downloadJSON`. Данные функции позволяют пользователю скачать результаты опроса в формате CSV или JSON. Внутри этих функций данные преобразуются в соответствующий формат. Для CSV создается строка с заголовками и данными, разделенными точкой с запятой. Для JSON данные преобразуются в строку с помощью `JSON.stringify`. Затем создается объект `Blob`, который представляет файл, и ссылка для его скачивания. При нажатии на кнопку файлы формата csv (см.рис.5) и файлы формата json (см.рис.6) автоматически загружаются на устройство пользователя.

Добавляем код функции `downloadCSV` для скачивания результатов в формате CSV:

```

function downloadCSV() {
  const resultsForDownload = getResultsForDownload();
  const csvContent = `Вопрос;Да;Нет\n` +
    Object.keys(resultsForDownload).map(question =>
    `${question};${resultsForDownload[question]['да']};${resultsForDownload[question]['нет']}`).join('\n');
  const blob = new Blob([csvContent], { type: 'text/csv;charset=utf-8;' });
  const link = document.createElement('a');
  link.href = URL.createObjectURL(blob);
  link.download = 'survey_results.csv';
  link.click();
}

```

Также добавляем код функции `downloadJSON` для скачивания результатов в формате JSON:

```

function downloadJSON() {
  const resultsForDownload = getResultsForDownload();
  const jsonContent = JSON.stringify(resultsForDownload, null, 2);
  const blob = new Blob([jsonContent], { type:
  'application/json;charset=utf-8;' });
  const link = document.createElement('a');
  link.href = URL.createObjectURL(blob);
  link.download = 'survey_results.json';
  link.click();
}

```

Прописываем код обработчика событий для кнопок скачивания:

```

document.getElementById('downloadCSVBtn').addEventListener('click',
downloadCSV);
document.getElementById('downloadJSONBtn').addEventListener('click',
downloadJSON);

```

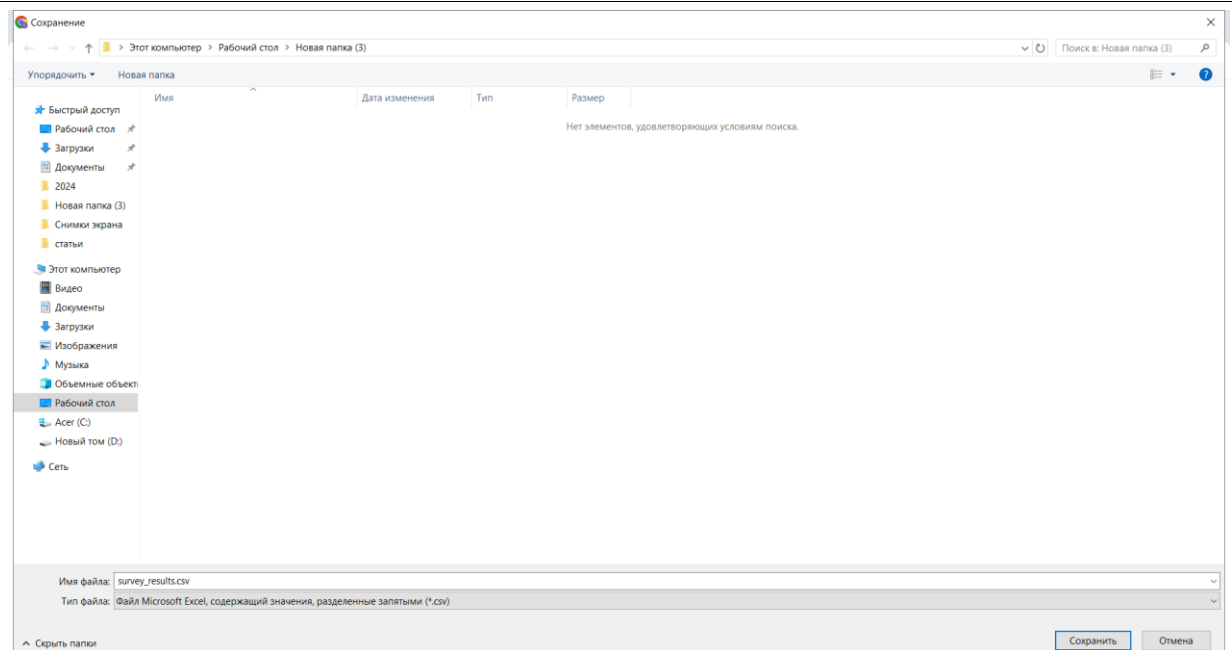


Рисунок 5. Скачивание файла результатов опроса в формате CVS

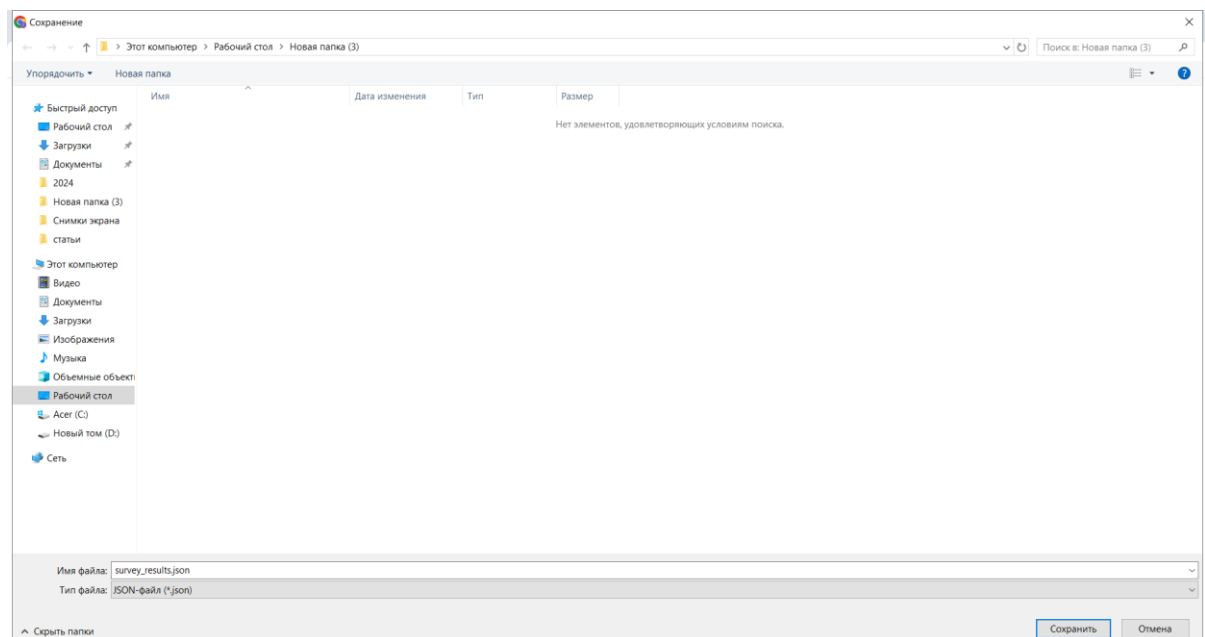


Рисунок 6. Скачивание файла результатов опроса в формате Json

Создаем заключительную функцию `getElementById`. Функция сброса результатов вызывается при нажатии на кнопку "Сбросить результаты" (см.рис.7). Функция запрашивает подтверждение у пользователя, чтобы избежать случайного сброса. Если пользователь подтверждает действие, объект `results` обнуляется, а данные удаляются из `localStorage`. После этого вызывается функция `displayResults`, чтобы обновить отображение результатов на странице. Этот шаг позволяет начать опрос заново, очистив все предыдущие данные. Прописываем код функции `getElementById`:

```
document.getElementById('resetCountsBtn').addEventListener('click', function()
{
    if (confirm('Вы действительно хотите сбросить все результаты?')) {
```

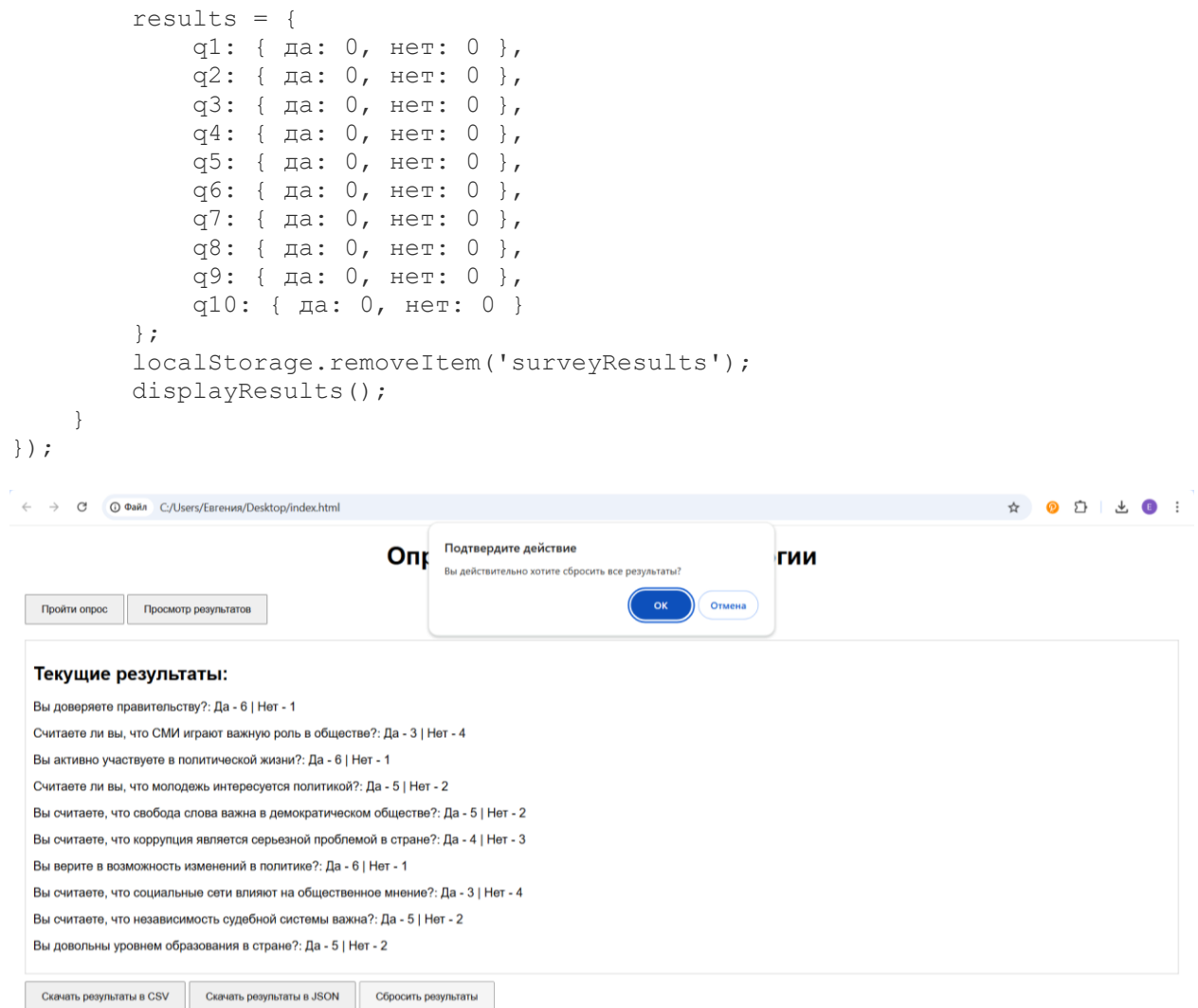


Рисунок 7. Сброс результатов

#### 4 Выводы

В данной работе был представлен процесс создания интерактивного веб-опросника с помощью языка программирования JavaScript, языка гипертекстовой разметки HTML, и языка описания внешнего вида документа CSS. Опросник позволяет пользователям отвечать на вопросы, сохранять результаты в локальном хранилище браузера и предоставляет возможность скачивания данных в форматах CSV и JSON для дальнейшего анализа. Научная статья может служить методическим пособием в учебной деятельности.

#### Библиографический список

1. Насретдинова М. М. Особенности организации опроса в сети интернет // Психология, социология и педагогика. 2015. №. 5. С. 76-77.
2. Рогожников Е. Д., Абрамова О. Ф., Лясин Д. Н. Разработка проектной документации и прототипа автоматизированной системы для проведения социологических опросов // Форум молодых ученых. 2019. №. 6 (34). С. 997-

1005.

3. Мальцева Е. Ю., Молчанова Е. В. Применение информационных технологий при проведении социологического исследования //Концепт. 2015. №. 2. С. 61-65.
4. Методика разработки web-анкет интернет-опросов, для изучения квалификации преподавательского состава вузов / Г. Д. Волкова, О. В. Новоселова, В. В. Попов [и др.] // Межотраслевая информационная служба. 2010. № 2. С. 32-39.
5. Отбоева С.Д. Современные информационные технологии в проведении социологических исследованиях // Современные научные исследования и инновации. 2018. № 9.