УДК 004

Е. А. Шевченко[1]✉, *Е. Ю. Воронкин*[1]

Разработка расширения браузера для создания коллекции образцов элементов веб-страницы с использованием JavaScript

[1]Сибирский государственный университет геосистем и технологий,
г. Новосибирск, Российская Федерация
e-mail: shea2002@mail.ru

Аннотация. В статье рассматривается процесс разработки расширения браузера для создания коллекции образцов элементов веб-страницы с использованием JavaScript. В ходе исследования была проанализирована предметная область, выделены существующие подходы к извлечению компонентов интерфейса, а также выявлены их ограничения. Целью работы стало создание расширения, позволяющего выделять элементы прямо на странице, очищать их от лишних стилей, просматривать в изолированной среде и сохранять в формате JSON для последующего использования в редакторах интерфейсов. При реализации использовался нативный стек технологий (HTML, CSS, JavaScript) и API Chrome Extension. Архитектура построена на модульной структуре с изолированными компонентами. В статье кратко изложены этапы рабработки, а также продемонстрирована работа расширения на сайте «METANIT».

Ключевые слова: браузерное расширение, JavaScript, предпросмотр, взаимодействие с DOM, извлечение данных, выбор стилей, хранение контента

E. A. Shevchenko[1]✉, *E. Yu. Vononkin*[1]

Developing a browser extension to create a collection of sample web page elements using JavaScript

[1]Siberian State University of Geosystems and Technologies, Novosibirsk, Russian Federation
e-mail: shea2002@mail.ru

Abstract. The article discusses the process of developing a browser extension to create a collection of sample elements of a web page using JavaScript. The study analyzed the subject area, highlighted existing approaches to extracting interface components, and identified their limitations. The purpose of the work was to create an extension that allows you to select elements directly on the page, clear them of unnecessary styles, view them in an isolated environment and save them in JSON format for future use in interface editors. The implementation used a native technology stack (HTML, CSS, JavaScript) and the Chrome Extension API. The architecture is based on a modular structure with isolated components. The article summarizes the stages of development, as well as demonstrates how the extension works on the «METANIT» website.

Keywords: browser extension, JavaScript, preview, DOM interaction, data extraction, style selection, content storage

Введение

В веб-разработке визуальная составляющая сайта играет не менее важную роль, чем функциональная. Заказчики зачастую приходят без четкого технического задания, что в разы усложняет процесс разработки. Разработчику приходится самостоятельно подбирать примеры с других сайтов и демонстрировать

возможные решения. При этом использование скриншотов или графических редакторов только повышает энергозатраты, что оказывается нерациональным в условиях малых или средних проектов. Работа через DevTools требует времени на анализ структуры и стилей, которые могут зависеть от контекста страницы. Все это замедляет процесс и усложняет взаимодействие с заказчиком.

Целью настоящей работы является разработка браузерного расширения, позволяющего визуально выбирать элементы с веб-страниц, просматривать их в интерактивном режиме и сохранять в формате, пригодном для дальнейшей разработки.

Работа обладает практической значимостью как готовый инструмент для веб-разработчиков, упрощающий процесс подготовки интерфейсных решений.

Методы и материалы

При разработоке расширения был проведен анализ существующих решений [1, 2], это позволило выделить их ограничения и сформулировать требования к собсвенной разработке.

Для построения архитектуры использовался модульный подход, обеспечивающий изоляцию компонентов и гибкость при масштабировании [3]. Проектирование интерфейсов велось с учетом принципов минимализма и быстрого взаимодействия. Логика приложения реализована на JavaScript, с использованием стандартов ECMAScript и API расширений Chrome, включая chrome.runtime, chrome.storage, chrome.tabs [4].

В качестве среды разработки использовался Visual Studio Code с набором расширений для форматирования, автодополнения и документации коду через JSDoc [5]. Для тестирования и отладки использовались Chrome DevTools и режим разработчика браузера Chrome. Хранение пользовательских данных осуществлялось через chrome.storage.local, что обеспечило изоляцию данных и независимость от сторонних серверов. Все модули реализованы с учетом ограничений Manifest V3.

Результаты

В результате было разработано расширение, позволяющее выделять HTML-элементы на веб-странице, просматривать их визуальное представление и структуру, а также сохранять в формате JSON для дальнейшего использования.

После запуска расширения открывается окно с кнопкой активации захвата (рис. 1).

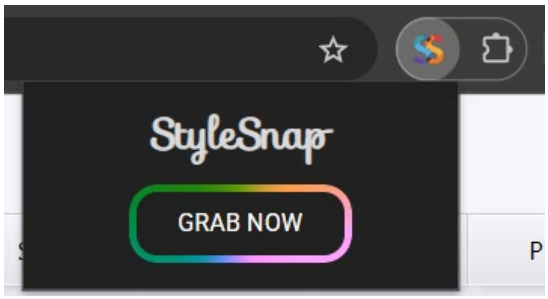

Рис. 1. Интерфейс расширения

После активации необходимо выбрать элемент на странице и кликнуть по нему, после этого появится список всех родительских и вложенных элементов выбранного блока. При наведении на любой из них в интерфейсе расширения нужный элемент на странице будет автоматически подсвечиваться, чтобы пользователь легко мог сориентироваться в структуре (рис. 2).

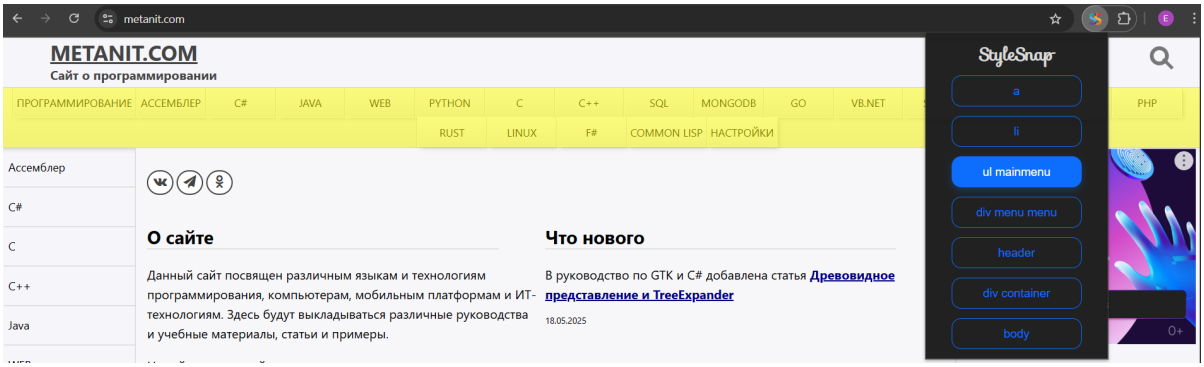

Рис. 2. Подсветка элементов на странице

Клик по элементу в расширении открывает предпросмотр, в котором отображается как внешний вид, так и структура DOM с возможностью перехода к дочерним узлам и их детального просмотра (рис. 3).

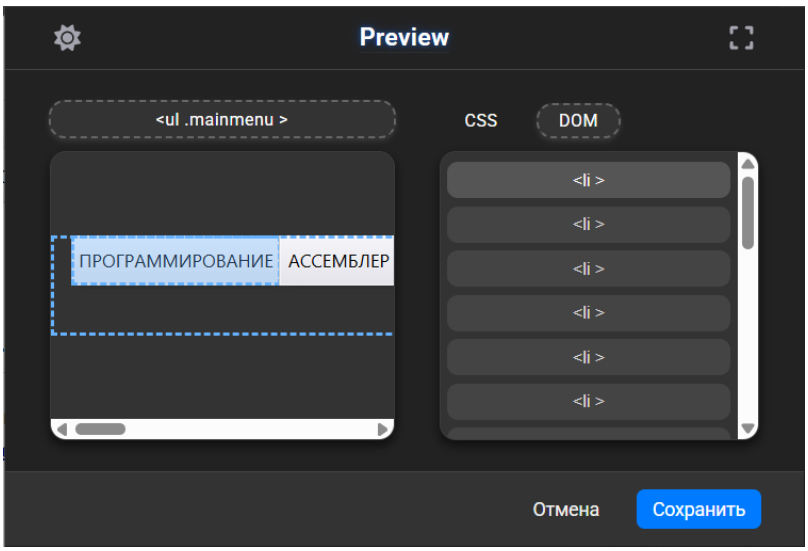

Рис. 3. Структура DOM и дочерние элементы

Пользователь может не только увидеть дочерние элементы, но и управлять стилями каждого из них с помощью интерфейса чекбоксов (рис. 4).

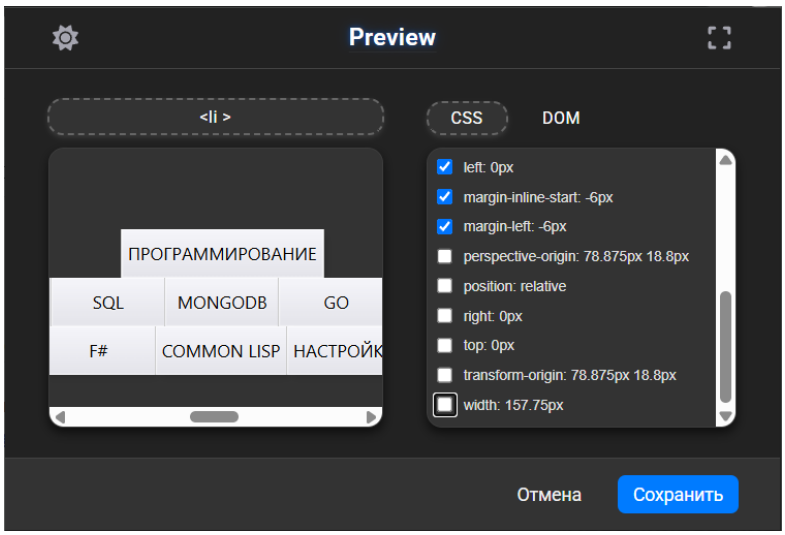

Рис. 4. Управление стилями через чекбоксы

Для более точной оценки предусмотрен полноэкранный предпросмотр, включающий в себя те же функции, которые были описаны выше (рис. 5).

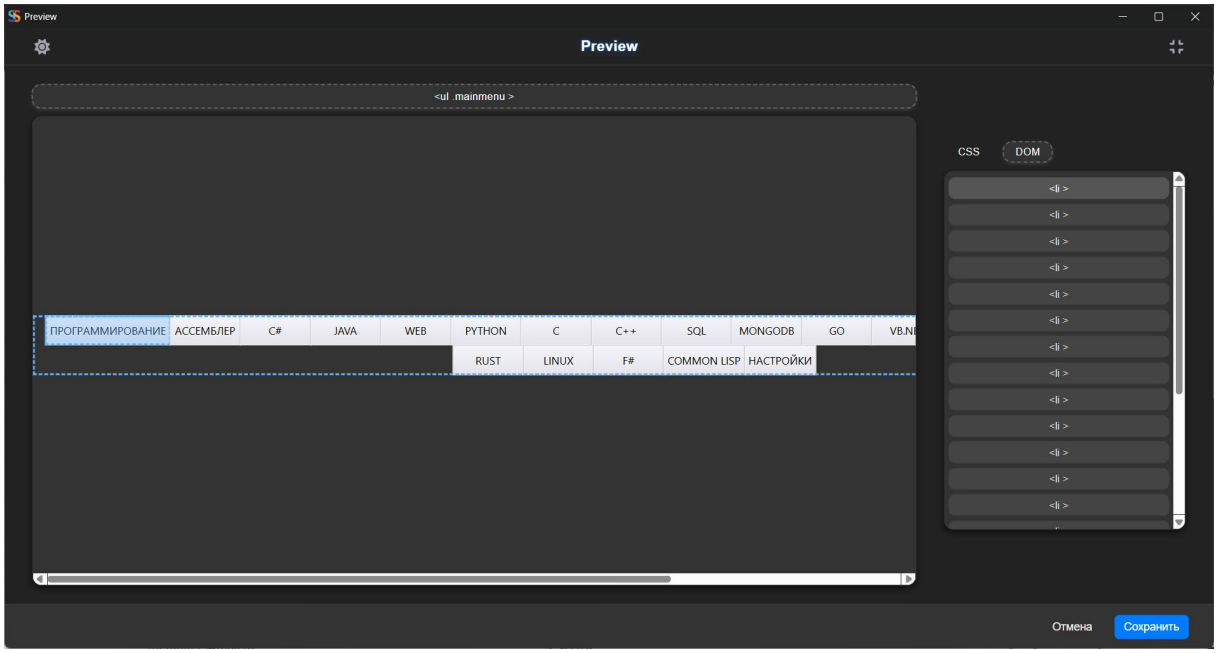

Рис. 5. Полноэкранный режим

Также было предусмотрено сохранение структуры и стилей элемента в формате JSON. При клике на кнопку «Сохранить» открывается окно, в котором необходимо дать наименование коллекции, а также указать в какую папку необходимо сохранить (рис. 6).

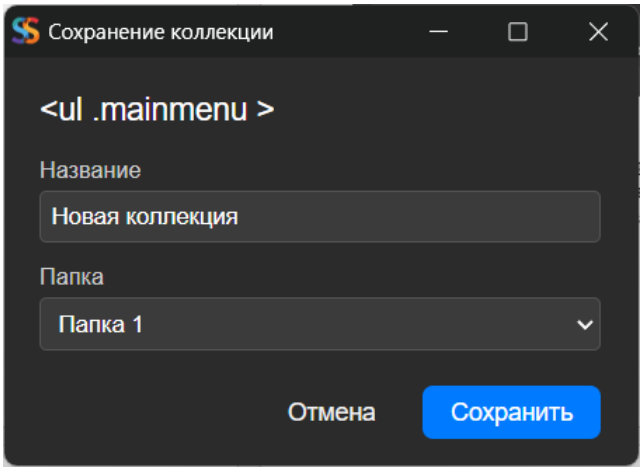

Рис. 6. Окно сохранения

Заключение

В результате выполненной работы было разработано расширение браузера для создания коллекции образцов элементов веб-страницы с использованием JavaScript. Сохраненный файл в формате JSON пригоден для повторного использования и может быть загружен в редактор WISUA.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. CopyCSS. Расширение для копирования HTML и CSS с веб-страницы. – URL: https://chrome.google.com/webstore/detail/copy-css (дата обращения: 21.01.2025).

2. SnipCSS. Инструмент для визуального извлечения стилей с сайтов. – URL: https://snipcss.com (дата обращения: 21.01.2025).

3. Фаулер М. Архитектура корпоративных приложений : пер. с англ. – М. : Вильямс, 2004. – 560 с.

4. Chrome Extensions API. Документация для разработчиков. – URL: https://developer.chrome.com/docs/extensions/ (дата обращения: 21.01.2025).

5. Visual Studio Code. Официальный сайт редактора кода. – URL: https://code.visualstudio.com/ (дата обращения: 21.01.2025).