http://dx.doi.org/10.35596/1729-7648-2023-29-4-58-65

Оригинальная статья
Original paper

УДК 004.4

МЕТОД И АЛГОРИТМ ДЛЯ ОБРАБОТКИ ИЗОБРАЖЕНИЙ БОЛЬШИХ ОБЪЕМОВ

О. Н. АНДРЕЙЧУК, Н. И. ЛИСТОПАД

Белорусский государственный университет информатики и радиоэлектроники
(г. Минск, Республика Беларусь)

Поступила в редакцию 17.09.2023

Аннотация. Предложены метод и алгоритм на его основе для обработки изображений больших объемов. Новизна в данном случае – это метод реализации программного кода с использованием объекта App, а также алгоритм специфической реализации фильтров для обработки графических элементов. В ходе разработки были учтены различные проблемные ситуации, такие как использование современного синтаксиса объявления переменных, применение стрелочных функций, разделение кода на более мелкие функции или методы класса, добавление комментариев к сложным участкам кода, проверка наличия ошибок и исключений. Кроме того, были соблюдены стандарты и рекомендации по стилю кодирования. Реализованный алгоритм на языке программирования JavaScript представляет собой эффективное веб-приложение, использующее библиотеку EaselJS для работы с графикой и отличающееся специфической логикой работы с графическими слоями, инструментами рисования и взаимодействия с пользователем. Это, в конечном счете, позволило, по сравнению с известными решениями, улучшить производительность кода.

Ключевые слова: веб-приложение, алгоритм обработки изображений, эффективность и оптимизация кода.

Конфликт интересов. Авторы заявляют об отсутствии конфликта интересов.

Для цитирования. Андрейчук, О. Н. Метод и алгоритм для обработки изображений больших объемов / О. Н. Андрейчук, Н. И. Листопад // Цифровая трансформация. 2023. Т. 29, № 4. С. 58–65. http://dx.doi.org/10.35596/1729-7648-2023-29-4-58-65.

METHOD AND ALGORITHM FOR PROCESSING LARGE VOLUME IMAGES

OLGA N. ANDREICHUK, NIKOLAI I. LISTOPAD

Belarusian State University of Informatics and Radioelectronics (Minsk, Republic of Belarus)

Submitted 17.09.2023

Abstract. The article proposes a new method and an algorithm based on it for processing large-volume images. What is new in the implementation of image processing is the method of implementing program code using the App object. The novelty of the proposed method also lies in the algorithm for the specific implementation of filters for processing graphic elements. During development, various problematic situations were taken into account, such as using modern variable declaration syntax, using arrow functions, dividing code into smaller functions or class methods, adding comments to complex sections of code, checking for errors and exceptions, and following standards and guidelines by coding style. The implemented algorithm in the JavaScript programming language is an effective web application that uses the EaselJS library to work with graphics and has specific logic for working with graphic layers, drawing tools and user interaction, which ultimately made it possible to improve code performance compared to known solutions.

Keywords: web application, image processing algorithm, code efficiency and optimization.

Conflict of interests. The authors declare no conflict of interests.

For citation. Andreichuk O. N., Listopad N. I. (2023) Method and Algorithm for Processing Large Volume Images. *Digital Transformation.* 29 (4), 58–65. http://dx.doi.org/10.35596/1729-7648-2023-29-4-58-65 (in Russian).

Введение

Специфика обработки изображений больших объемов является важным аспектом, который необходимо учитывать для повышения эффективности и оптимизации функционирования соответствующих веб-приложений, базирующихся на новых алгоритмах, позволяющих произвести оптимизацию кода и сократить время выполнения, затрачиваемое на саму обработку. Оптимальное решение – реализация алгоритма обработки изображений с помощью языка программирования JavaScript, поскольку он является клиентским и не требует установки дополнительного программного средства на персональный компьютер для запуска и работы [1].

Веб-приложение имеет хорошую, рациональную конечную структуру файлов, что позволяет системе функционировать достаточно быстро и стабильно. Имена файлов и папок приведены к нижнему регистру и не имеют пробелов в имени, что обеспечивает гарантию работоспособности веб-приложения на любых веб-сервисах.

JavaScript как один из наиболее популярных языков программирования для веб-разработки предоставляет множество возможностей для создания функциональных и интерактивных приложений. Однако, чтобы обеспечить оптимальные пользовательские запросы и повысить производительность, необходимо применять новые методы, алгоритмы и современные стили написания кода, учитывать различные проблемные ситуации.

Методы проведения исследований и их результаты

Обработка изображений, как известно [2], представлена набором функций, методов, объектов на языке программирования JavaScript. Все файлы js имеют корректное синтаксическое оформление кода. Каждое действие, которое реализовано в веб-приложении, представлено в отдельных функциях, что ускоряет выполнение необходимых действий. Например, посредством функций можно вызывать определенные действия несколько раз по ходу выполнения веб-приложения.

Предлагается следующий подход при разработке веб-приложения. Прежде всего, необходимо применять исключительно современные стили написания кода с целью его оптимизации, при этом нужно учитывать следующие проблемные ситуации:

– использовать современный синтаксис объявления переменных с помощью let или const вместо var. Например: for (let i = 0; let layer = this.layers[i]; i++);

– использовать стрелочные функции вместо функций обратного вызова. Например: this.layers. forEach((v) => { if (v.active) ret = v; });

– код должен быть разделен на более мелкие функции или методы класса для повышения читаемости и поддерживаемости;

– в коде должны присутствовать комментарии к сложным участкам кода, чтобы облегчить его понимание;

– код должен быть проверен на наличие потенциальных ошибок и исключений, должны быть добавлены обработка ошибок и проверка входных данных, где это необходимо;

– необходимо применить современный стандарт и рекомендации по стилю кодирования, такие как использование отступов, правильное именование переменных и функций, использование одинарных или двойных кавычек для строковых литералов.

Разработанный код на языке программирования JavaScript представляет собой реализацию веб-приложения с использованием библиотеки EaselJS для работы с графикой. Новизна разработки заключается в применении оригинального подхода, заключающегося в специфической логике работы с графическими слоями, инструментами рисования и взаимодействия с пользователем. Специфика логики работы – в функциональном подходе, что значительно сокращает количество строк кода, делает его более читабельным, повышает производительность.

Файловая структура приложения, реализующего новый подход, представлена на рис. 1. Веб-приложение имеет рациональную конечную структуру файлов, что позволяет ему работать достаточно стабильно с хорошим быстродействием. Имена файлов и папок приведены к соответствующему регистру, согласно спецификации CamelCase, и не имеют пробелов в имени, что обеспечивает гарантию работоспособности веб-приложения на любых веб-сервисах.

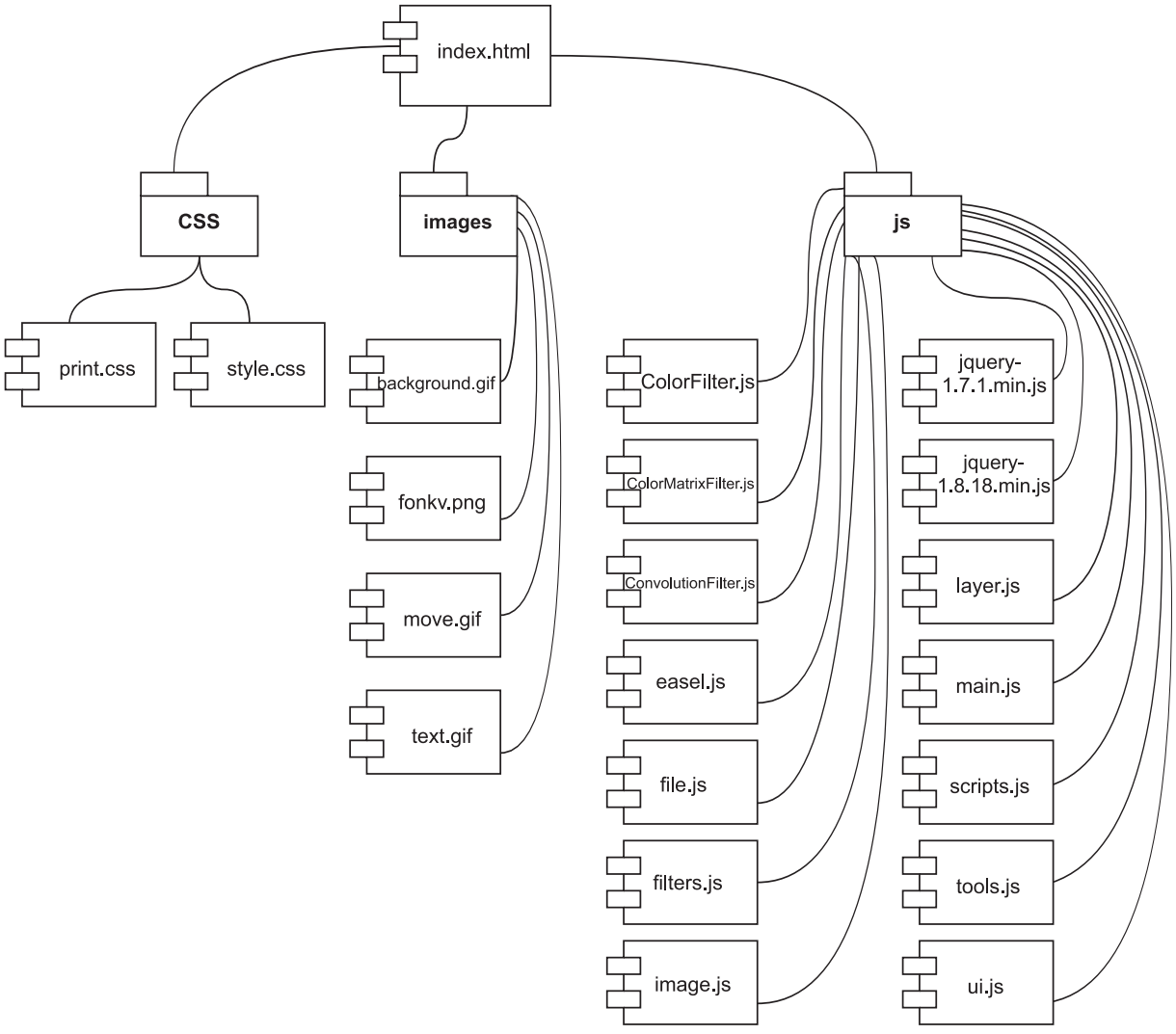

Рис. 1. Файловая структура приложения
Fig. 1. Application file structure

Основная структура приложения построена в виде объекта App, что позволяет оперировать данными как единым целым. Новым в реализации обработки изображений большого объема как раз и является метод реализации программного кода с использованием объекта App. Вся файловая структура js-файлов имеет доступ к данному объекту и его свойствам. Такой способ позволяет ускорить работу веб-приложения и как следствие – повысить эффективность обработки изображения за счет уменьшения временных затрат.

Представление объекта App в каком-то смысле уникально, поскольку эта структура содержит в себе функции для работы со слоями при обработке изображений. Данные функции наследуются во всей структуре веб-приложения. Блок-схема работы объекта App представлена на рис. 2.

Рассмотрим код на примере метода загрузки слоев loadLayers объекта App, который хранится в файле main.js. Код выполняет функцию загрузки слоев из JSON-объекта в веб-приложение. Метод loadLayers принимает два параметра – from и to, являющиеся массивами слоев. Вначале извлекается последний элемент из массива from, который сохраняется в переменной jsonString. Затем проверяется, является ли jsonString неопределенным. Если переменная jsonString – неопределенная, то функция возвращает false, и вся процедура завершается.

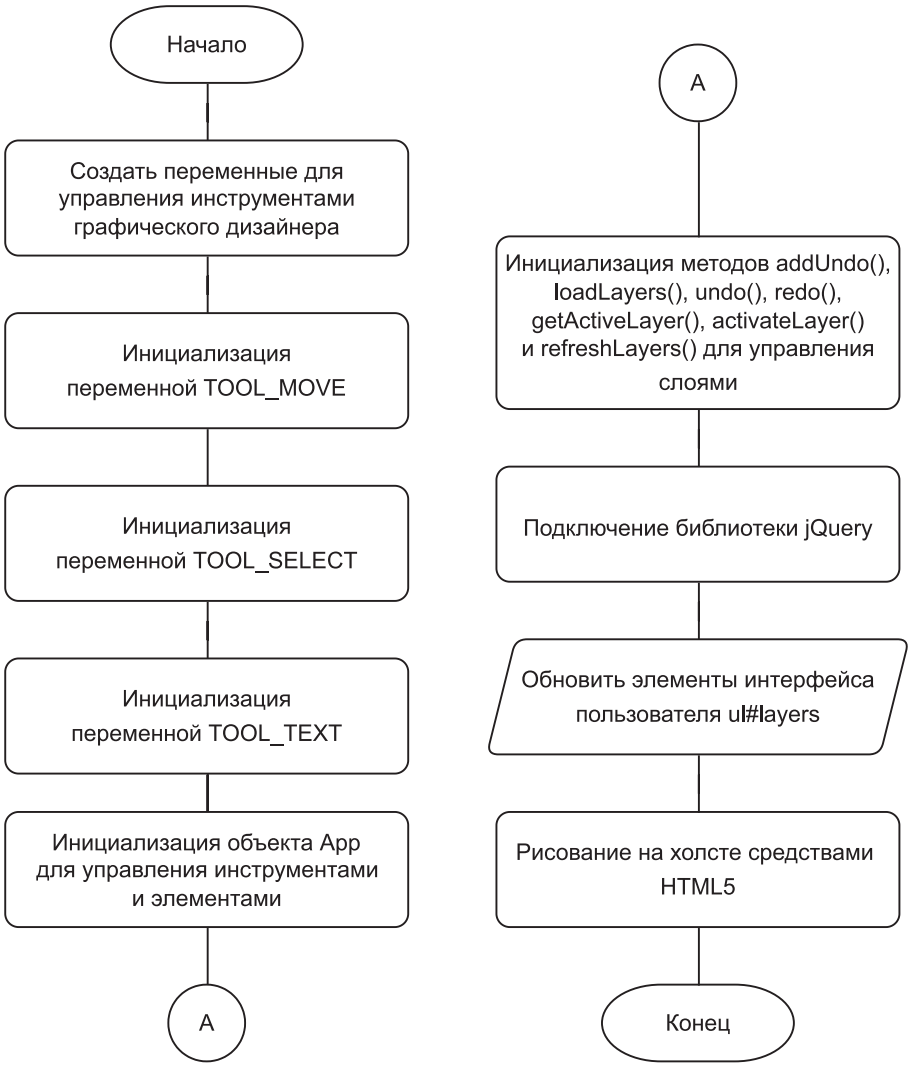

Рис. 2. Блок-схема работы объекта App
Fig. 2. Block diagram of the operation of the App object

После этого текущие слои (this.layers) преобразуются в строку и добавляются в массив посредством оператора to. Далее происходит парсинг JSON-строки jsonString в объект json. Затем выполняется цикл по индексам слоев (i). Для каждого слоя происходит цикл по значениям объекта jsonLayer, который представляет собой слой из объекта json. Если значение не является фильтром (value != ‘filters’), оно присваивается соответствующему свойству слоя, а если является фильтром, то создается новый массив layer.filters. Далее происходит цикл по фильтрам слоя из объекта jsonLayer. Для каждого фильтра создается новый экземпляр класса, имя которого берется из массива jsonLayer.filters.names. Потом значения фильтра присваиваются соответствующим свойствам этого экземпляра.

После применения всех фильтров проверяется, были ли у слоя предыдущие фильтры (hadFilters). Если это так, то обновляется кэш слоя. Наконец, вызывается функция refreshLayers(), которая обновляет отображение слоев в графическом приложении. Описанный выше алгоритм можно представить в виде блок-схемы, приведенной на рис. 3.

Новым в предлагаемом методе является алгоритм специфической реализации фильтров для обработки графических элементов. В коде представлены различные функции фильтрации, такие как filterBrightness, filterColorify, filterDesaturation, filterBlur, filterGaussianBlur, filterEdgeDetection, filterEdgeEnhance, filterEmboss и filterSharpen. Каждая функция фильтрации принимает определенные параметры (например, яркость, цветовую коррекцию, радиус размытия и т. д.) и применяет соответствующие им значения.

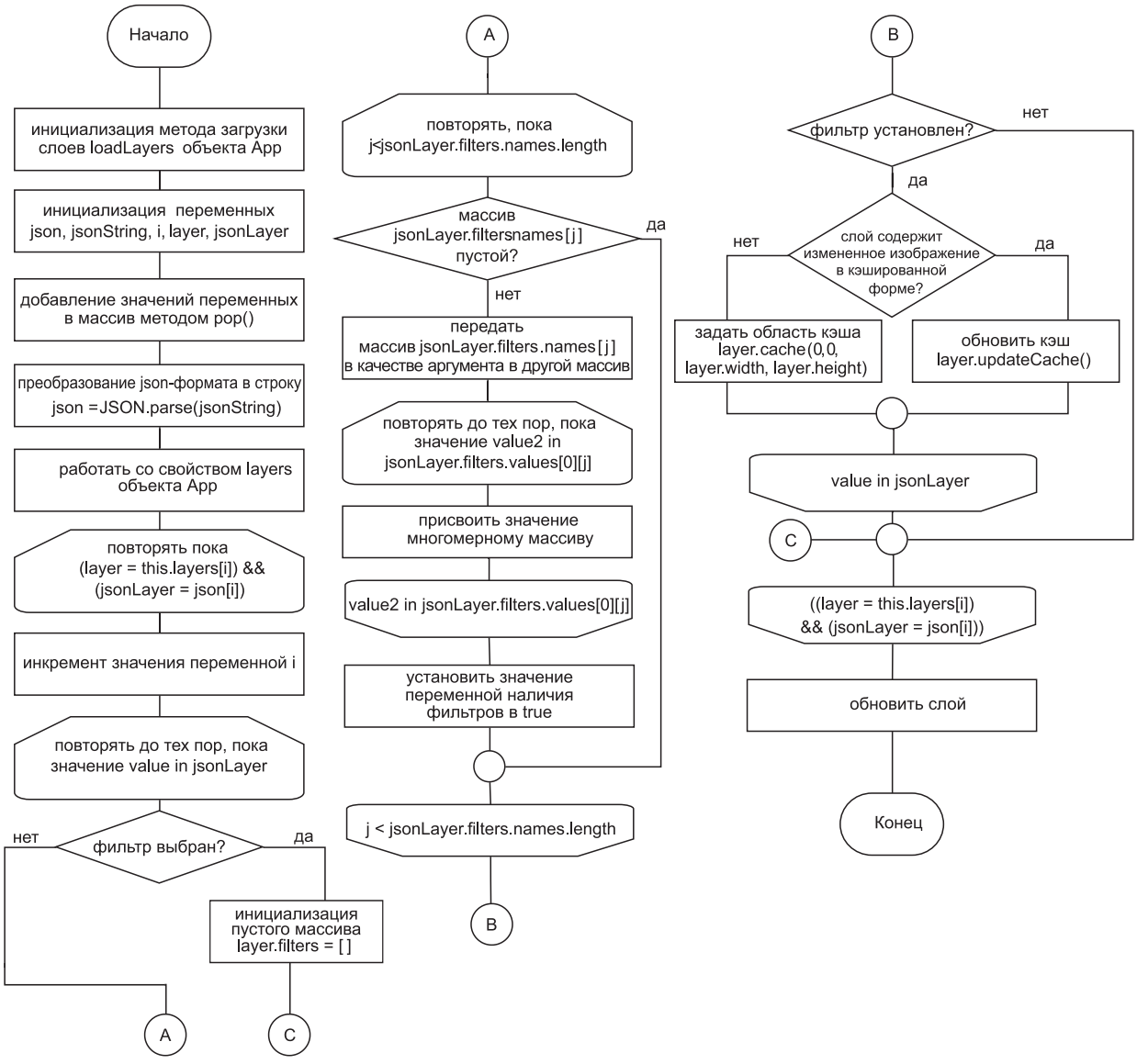

Рис. 3. Алгоритм работы метода loadLayers
Fig. 3. Algorithm for the loadLayers method

Рассмотрим работу кода на примере функции applyFilter, которая является универсальной для всех фильтров в веб-приложении. Данный способ реализации функции позволяет сократить количество строк кода и увеличить скорость работы веб-приложения. Первоначально функция applyFilter вызывает метод addUndo() объекта App, который добавляет текущее состояние слоев в историю изменений для возможности отмены операций.

Затем функция получает активный слой с помощью метода getActiveLayer() объекта App и сохраняет его в переменную layer. Далее проверяется, существует ли у слоя массив фильтров (layer.filters). Если массив не существует, то создается пустой массив. Затем фильтр добавляется в массив фильтров слоя с помощью метода push(). После добавления фильтра происходит проверка, есть ли у слоя кэш (layer.cacheCanvas). Если кэш существует, вызывается метод updateCache(), который обновляет кэшированное изображение слоя с учетом примененного фильтра. Если кэш не существует, вызывается метод cache() для создания нового кэша слоя, охватывающего всю его ширину и высоту. Таким образом, функция applyFilter добавляет фильтр в массив фильтров активного слоя и обновляет его кэш для отображения изменений. Алгоритм работы функции applyFilter представлен на рис. 4.

В отличие от существующих решений [3] в приложении улучшена производительность кода примерно в 9,8 раза (рис. 5) за счет:

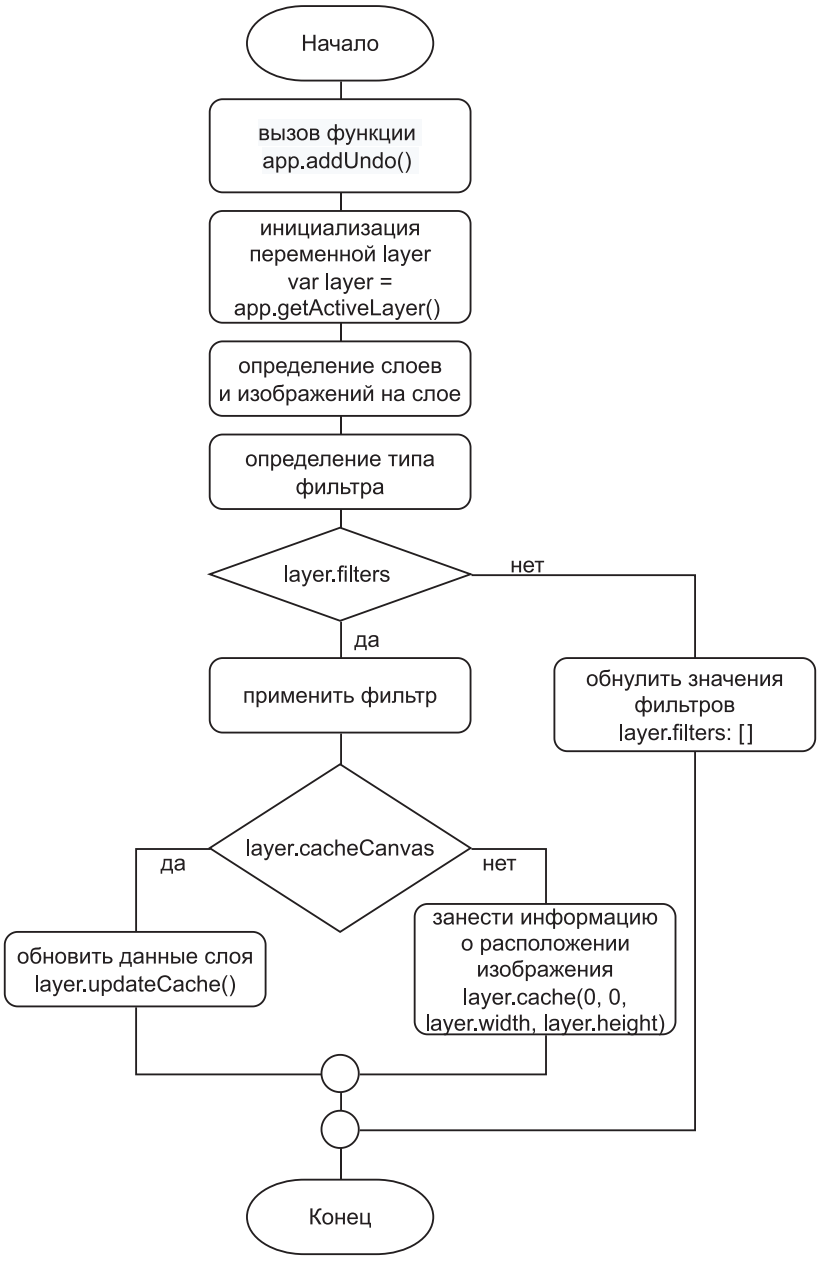

Рис. 4. Алгоритм работы функции applyFilter
Fig. 4. Algorithm for the applyFilter function

1) использования констант: в коде присутствуют три константы TOOL_MOVE, TOOL_SELECT и TOOL_TEXT, что позволяет избегать их повторного определения и помогает ускорить выполнение кода;

2) использования цикла for вместо метода forEach: цикл for работает быстрее, чем метод forEach, особенно при обработке большого количества данных;

3) уменьшения обращений к DOM: создана временная строка HTML вместо обновления списка слоев каждый раз при вызове метода refreshLayers;

4) использования локальных переменных;

5) использования более эффективных алгоритмов: в коде применены более эффективные алгоритмы, чтобы ускорить выполнение кода, например, алгоритм сортировки вместо простого перебора при сортировке списка слоев;

6) оптимизации циклов: циклы оптимизированы, чтобы ускорить выполнение кода, например, в цикле for использован префиксный оператор ++ вместо постфиксного, чтобы избежать лишнего присваивания значения переменной [4].

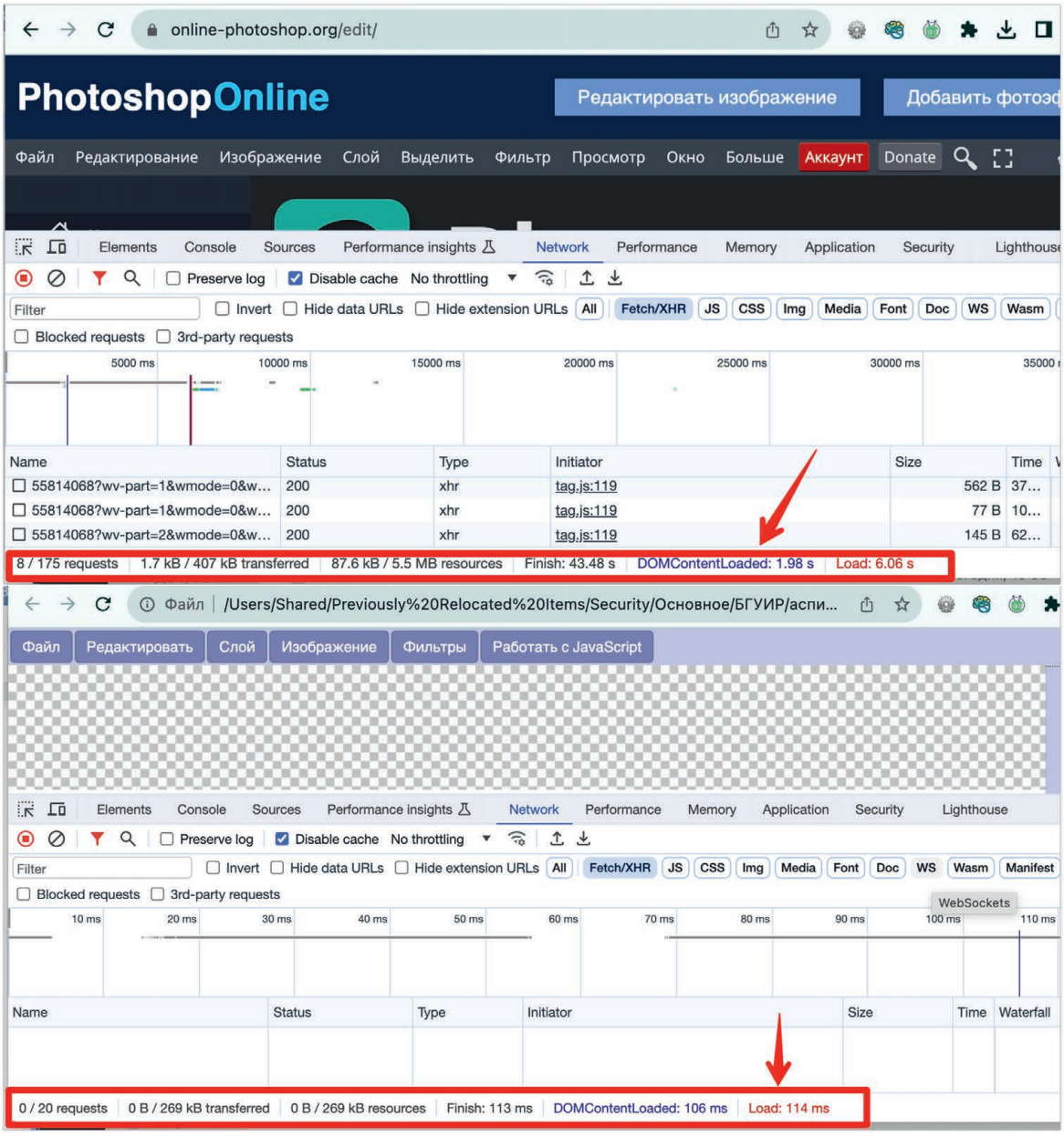

Рис. 5. Загрузка веб-сайта PhotoshopOnline и разработанного веб-приложения
Fig. 5. Uploading the PhotoshopOnline website and the developed web application

Заключение

1. Предложены инновационный метод и соответствующий алгоритм для обработки больших объемов изображений. Уникальность этой методики заключается в разработке программного кода с применением объекта App и в особом алгоритме для фильтрации графических элементов.

2. Разработанный алгоритм на языке программирования JavaScript представляет собой эффективное веб-приложение, использующее библиотеку EaselJS для работы с графикой и отличающееся специфичной логикой взаимодействия с графическими слоями, инструментами рисования и взаимодействия с пользователями. Это позволило значительно повысить производительность кода по сравнению с существующими решениями.

Список литературы

1. Цифровая обработка изображений в информационных системах / И. С. Грузман [и др.]. Новосибирск: Изд-во Новосиб. гос. техн. ун-та, 2000. 168 с.

2. Гонсалес, Р. Цифровая обработка изображений. 3-е изд. испр. и доп. / Р. Гонсалес, Р. Вудс. М.: Техносфера, 2012. 1104 с.
3. Кузница IT-решений [Электронный ресурс]. Режим доступа: http://shiftoffproblem.com/what-is-spa-and-mpa/.
4. Флэнаган, Д. JavaScript. Подробное руководство / Д. Флэнаган; пер. с англ. СПб.: Символ-Плюс, 2017. 992 с.

References

1. Gruzman I. S., Kirichuk V. S., Kosykh V. P., Peretyagin G. I., Spector A. A. (2000) *Digital Image Processing in Information Systems*. Novosibirsk: Publishing House of Novosibirsk State Technical University. 168.
2. Gonzalez R., Woods R. (2012) *Digital Image Processing, 3rd ed*. Moscow, Tekhnosphere Publ. 1104.
3. *Forge of IT Solutions*. Available: http://shiftoffproblem.com/what-is-spa-and-mpa/.
4. Flanagan D. (2017) *JavaScript. Detailed Guide, per. from English*. St. Petersburg, Symbol-Plus. 992.

Вклад авторов

Авторы внесли равный вклад в написание статьи.

Authors' contribution

The authors contributed equally to the writing of the article.

Сведения об авторах

Андрейчук О. Н., преподаватель филиала Белорусского государственного университета информатики и радиоэлектроники «Минский радиотехнический колледж»

Листопад Н. И., д. т. н., профессор, заведующий кафедрой информационных радиотехнологий Белорусского государственного университета информатики и радиоэлектроники

Адрес для корреспонденции

220013, Республика Беларусь,
г. Минск, просп. Независимости, 62
Филиал «Минский радиотехнический колледж»
Белорусского государственного университета
информатики и радиоэлектроники
Тел.: +375 29 341-04-17
E-mail: memory1703@gmail.com
Андрейчук Ольга Николаевна

Information about the authors

Andreichuk O. N., Lecturer at the Branch of the Belarusian State University of Informatics and Radioelectronics "Minsk Radiotechnical College"

Listopad N. I., Dr. of Sci. (Tech.), Professor, Head at the Department of Information Radio Technologies of the Belarusian State University of Informatics and Radioelectronics

Address for correspondence

220013, Republic of Belarus,
Minsk, Nezavisimosty Ave., 62
Branch of the Belarusian State University
of Informatics and Radioelectronics
"Minsk Radiotechnical College"
Tel.: +375 29 341-04-17
E-mail: memory1703@gmail.com
Andreichuk Olga Nikolaevna