http://doi.org/10.35596/2522-9613-2022-28-2-52-60

Оригинальная статья
Original paper

УДК 004.4

РАЗРАБОТКА АЛГОРИТМОВ ОБРАБОТКИ ИЗОБРАЖЕНИЙ БОЛЬШИХ ОБЪЕМОВ

О. Н. ВИНИЧУК, В. И. ДРАВИЦА

Белорусский государственный университет информатики и радиоэлектроники,
филиал «Минский радиотехнический колледж»
(г. Минск, Республика Беларусь)

Поступила в редакцию 20 июня 2022

Аннотация. В последние годы существенно возрос интерес к цифровой обработке изображений, поэтому совсем не случайно, что цифровая обработка – одно из интенсивно развиваемых направлений исследования. При работе с компьютерной системой достаточно важным фактором является качественное отображение изображений, вследствие чего не менее важным фактором являются и методы обработки и улучшения изображений, которые не только отвечают за качественное отображение, но и позволяют сделать интересующие детали на изображении более заметными. Сегодня достаточно сложно найти приложение или веб-приложение с простым и удобным интерфейсом, а также с относительно низкими характеристиками по энергозатратам на операционную систему и устройство в целом. В данной статье представлены новые алгоритмы, которые позволяют повысить эффективность обработки изображений за счет сокращения времени загрузки приложения и самой обработки, а также снижения нагрузки на операционную систему.

Ключевые слова: веб-приложение, обработка изображений, обработка изображений больших объемов.

Конфликт интересов. Автор заявляет об отсутствии конфликта интересов.

Для цитирования. О. Н. Виничук, В. И. Дравица. Разработка алгоритмов обработки изображений больших объемов. *Цифровая трансформация.* 2022; 28(2): 52-60.

DEVELOPMENT OF ALGORITHMS FOR PROCESSING IMAGES OF LARGE VOLUMES

OLGA N. VINICHUK, VIKTOR I. DRAVITSA

Belarusian State University of Informatics and Radioelectronics,
Branch "Minsk Radio Engineering College" (Minsk, Republic of Belarus)

Submitted 20 June 2022

Abstract. In recent years, interest in digital image processing has increased significantly, so it is no coincidence that digital processing is one of the intensively developed areas of research. When working with a computer

system, a rather important factor is the high-quality display of images, as a result of which the methods of processing and improving images are no less important factors, which are not only responsible for the high-quality display of the image, but also allow to increase the visibility of interesting details in the image. Today it is quite difficult to find an application or a web application with a simple and user-friendly interface, as well as with relatively low characteristics in terms of energy consumption needed to supply the operating system and the device in general. This article presents new algorithms that improve the efficiency of image processing by reducing application loading and processing time, as well as by reducing the load on the operating system.

Keywords: web application, image processing, large volume image processing.

Conflict of interests. The author declares no conflict of interests.

For citation. Vinichuk O. N., Dravitsa V. I. Development of Algorithms for Processing Images of Large Volumes. *Digital Transformation.* 2022; 28(2): 52-60.

Введение

Основой создания любого приложения являются алгоритмы работы различных его частей. При выборе алгоритма важно проанализировать основные функции, которыми будет обладать разрабатываемое приложение.

Основной функционал, который отвечает за обработку изображений, хранится в библиотеке EaselJS [1]. Библиотека EaselJS была выбрана неслучайно: она облегчает работу с элементом Canvas – элемент HTML5, предназначенный для создания растрового двухмерного изображения при помощи JavaScript. На рис. 1 представлена блок-схема подключения и настройки библиотеки EaselJS.

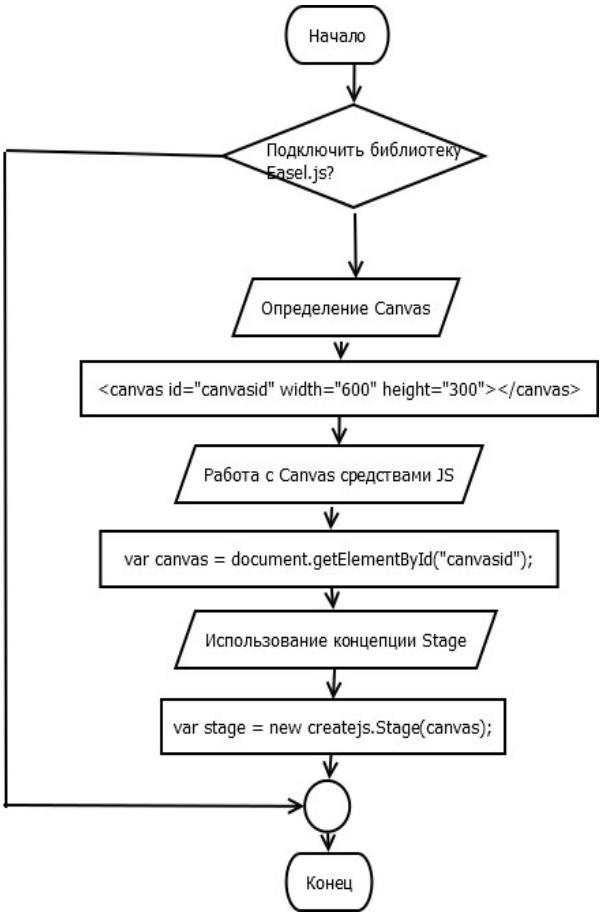

Рис. 1. Подключение библиотеки EaselJS.js
Fig. 1. Connecting the EaselJS.js library

У библиотеки EaselJS имеется класс Graphics, который предоставляет легкий в использовании API (программный интерфейс приложения) для создания инструкций векторного рисунка и прорисовывания их на указанный контекст [1, 2]. Команды EaselJS очень похожи на те, что используются в обычном Canvas HTML5. Вместе с тем, в EaselJs имеются и некоторые отличия для работы с Canvas.

Приложение для обработки изображений

В разработанном приложении реализованы функции посредством класса Filter. Как известно, класс Filter – это базовый класс, от которого наследуются свойства всех остальных фильтров [1]. Фильтры применяются к объектам, которые были помещены в кеш компьютера при помощи метода кеширования (cache). Если объект изменяется, необходимо поместить данный объект в кеш снова или использовать updateCache(). Необходимо также отметить, что фильтры должны быть применены перед помещением изображения в кеш. В этой связи использование библиотеки EaselJS является оправданным, так как появляется возможность использовать (применить) ряд предварительно созданных фильтров. Вышеописанный процесс представлен на рис. 2.

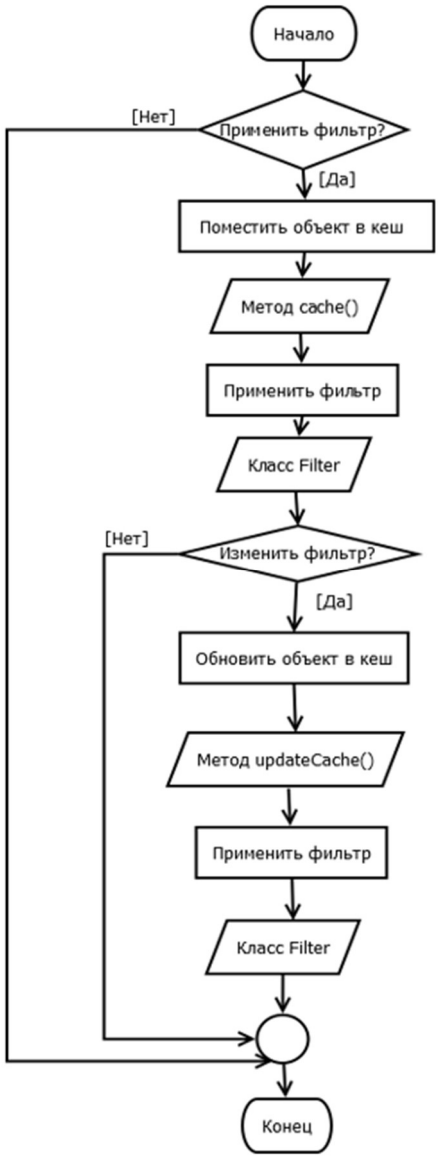

Рис. 2. Процесс применения фильтра к изображению
Fig. 2. The process of applying the filter to an image

На рис. 3 представлена схема алгоритма работы приложения. Основные функции веб-приложения здесь отображены в виде блоков алгоритма.

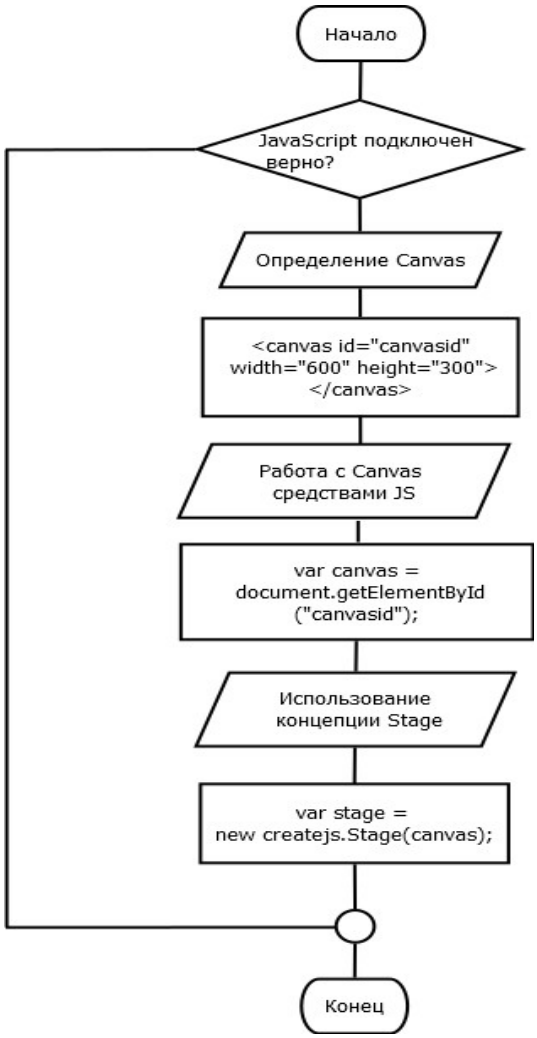

Рис. 3. Схема алгоритма работы приложения
Fig. 3. Scheme of the algorithm of the application

Новизна представленного алгоритма состоит в том, что работа над изображением реализована посредством HTML элемента, который используется для представления графики средствами языков программирования, в частности JavaScript.

При работе с веб-приложением для обработки изображений большого объема необходимо учитывать тот факт, что вся работа таких приложений построена на скриптах языка JavaScript. Ввиду этого при построении общего алгоритма идет проверка на подключение скриптов. Т.е. если скрипты в веб-приложении отсутствуют, то можно завершать работу с приложением и обработка изображений будет недоступна. Если же все скрипты подключены корректно, то следующим шагом будет определение рабочей области, а именно области Canvas, на которую помещаются изображения для последующей обработки. После того, как инициализирована рабочая область и помещено изображение, происходит основная работа веб-приложения. С рабочей области Canvas средствами JavaScript происходит получение объекта-изображения и, в зависимости от действий пользователя, производится его обработка [2].

Для применения фильтров в веб-приложении реализована универсальная функция applyFilter(), что, несомненно, сокращает время обработки изображения. Данная функция используется во всех применениях фильтров к цифровому изображению. Блок-схема алгоритма работы функции applyFilter() представлена на рис. 4 [1].

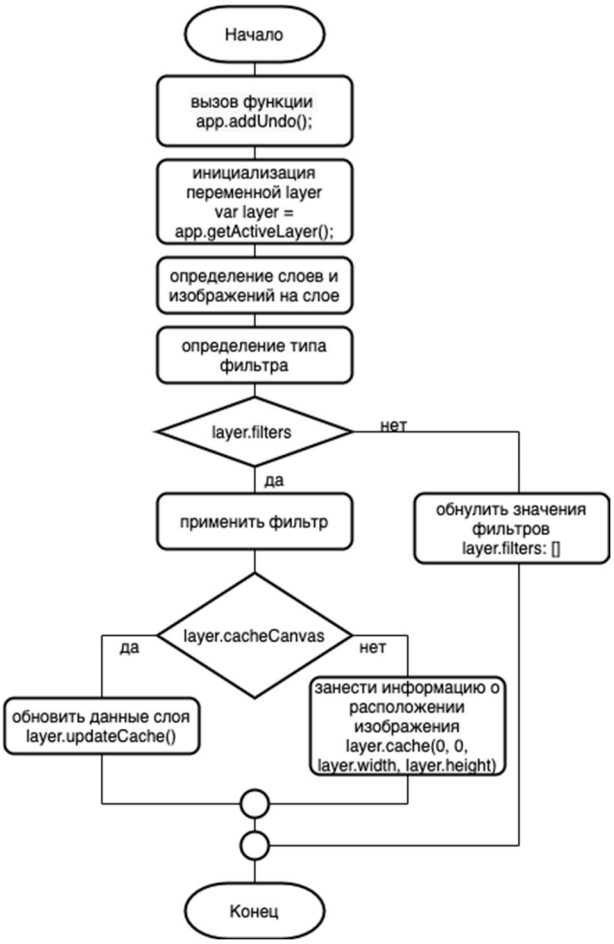

Рис. 4. Алгоритм работы функции applyFilter()
Fig. 4. The algorithm of the applyFilter() function

Для работы функции applyFilter() необходима инициализация и вызов функции addUndo(), которая отвечает за преобразование данных слоя и сохранение данных об изображении в буфере.

Ввиду того, что на рабочую область можно добавить несколько изображений, а каждое из изображений добавляется в качестве отдельного слоя, который можно скрыть, необходимо было реализовать функцию, которая определяет, скрыт ли слой – функция getActiveLayer(). При написании алгоритма для обработки изображений был использован механизм функций-конструкторов.

В качестве примера рассмотрим функцию-конструктор для добавления цветового фильтра на изображение ColorFilter. Для добавления цветового фильтра на изображение используется функция filterColorify, в которую обязательно передаются три параметра, а именно значения цветов red, green, blue в диапазоне от 0 до 255. Функция filterColorify представлена на рис. 5 [1].

```
filterColorify = function (r, g, b) {
    applyFilter(new ColorFilter(1.0, 1.0, 1.0, 1.0, r, g, b));
    hideDialog('#dialog-filtercolorify');
}
```

Рис. 5. Функция filterColorify
Fig. 5. Function filterColorify

Тело функции filterColorify содержит три функции:

– applyFilter() – функция применения фильтра;
– ColorFilter() – функция-конструктор;
– hideDialog() – функция, которая скрывает диалоговое окно. Блок-схема работы алгоритма функции ColorFilter представлена на рис. 6.

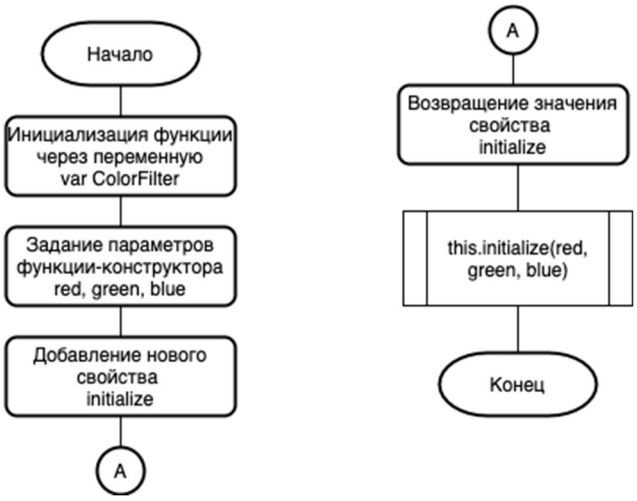

Рис. 6. Алгоритм работы функции ColorFilter()
Fig. 6. The algorithm of the ColorFilter() function

В веб-приложении реализована достаточно удобная функция сохранения изображения. Сохраняется полностью рабочая область: если, например, на рабочую область добавлено несколько изображений, их не придется объединять в один слой – объединение происходит автоматически при сохранении. Блок-схема алгоритма работы сохранения изображения представлена на рис. 7 [1].

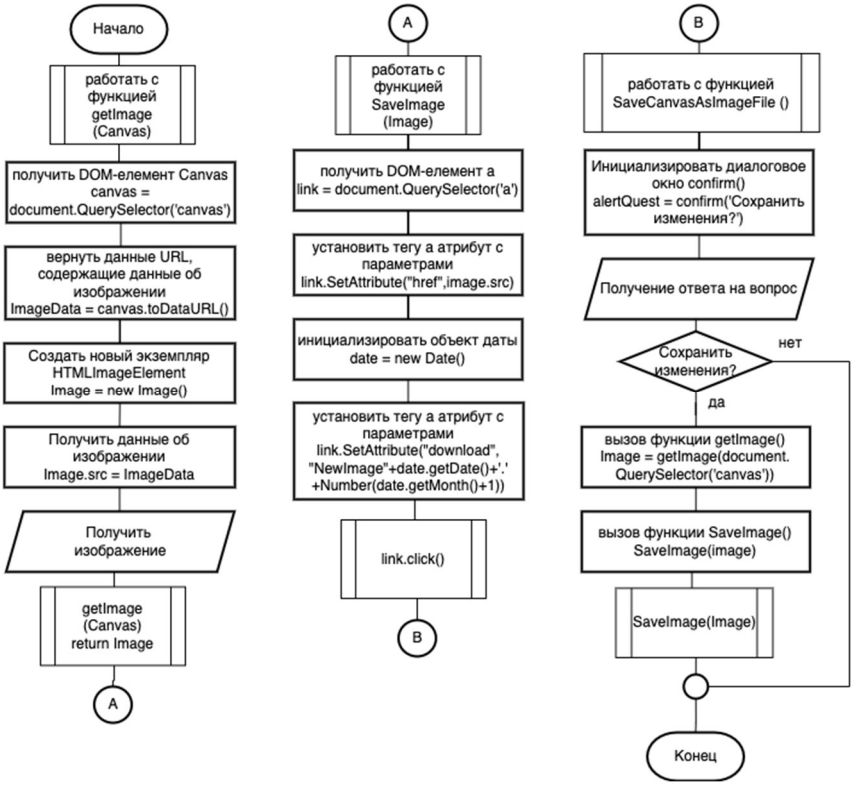

Рис. 7. Алгоритм сохранения изображения
Fig. 7. Algorithm for saving an image

Как следует из рис. 7, процесс сохранения изображения состоит из трех функций:

- getImage(Canvas);
- SaveImage(Image);
- SaveCanvasAsImageFile().

Функция SaveCanvasAsImageFile() отвечает за непосредственное сохранение данных, полученных с Canvas, поэтому включает в себя еще две функции getImage(Canvas) и SaveImage(Image) [1].

Анализ эффективности разработанного приложения

На примере веб-сайта «Photoshop Online» проведем сравнительный анализ эффективности разработанного приложения. В качестве примера проанализируем загрузку вышеназванного веб-сайта и разработанного приложения. Полная загрузка главной страницы веб-сайта «Photoshop Online» завершилась за 1,41 с (рис. 8).

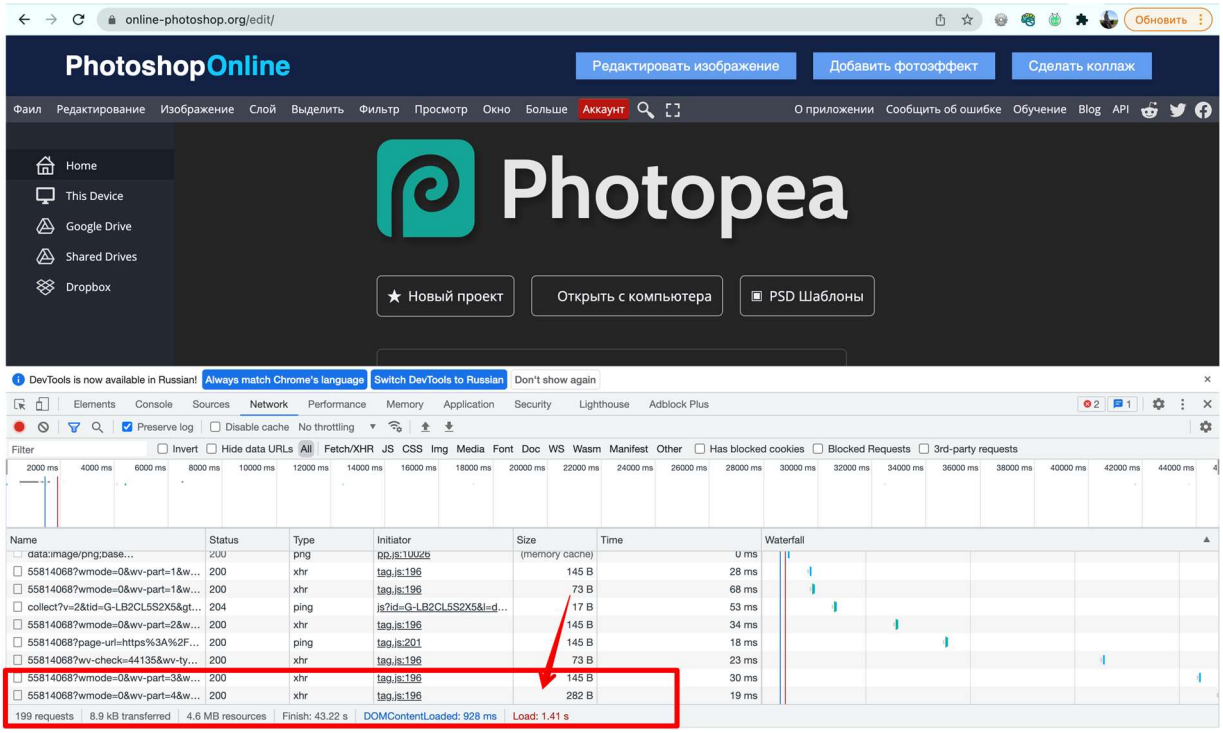

Рис. 8. Загрузка веб-сайта «Photoshop Online»
Fig. 8. Loading the website "Photoshop Online"

Полная загрузка разработанного веб-приложения завершилась за 143 мс (рис. 9).

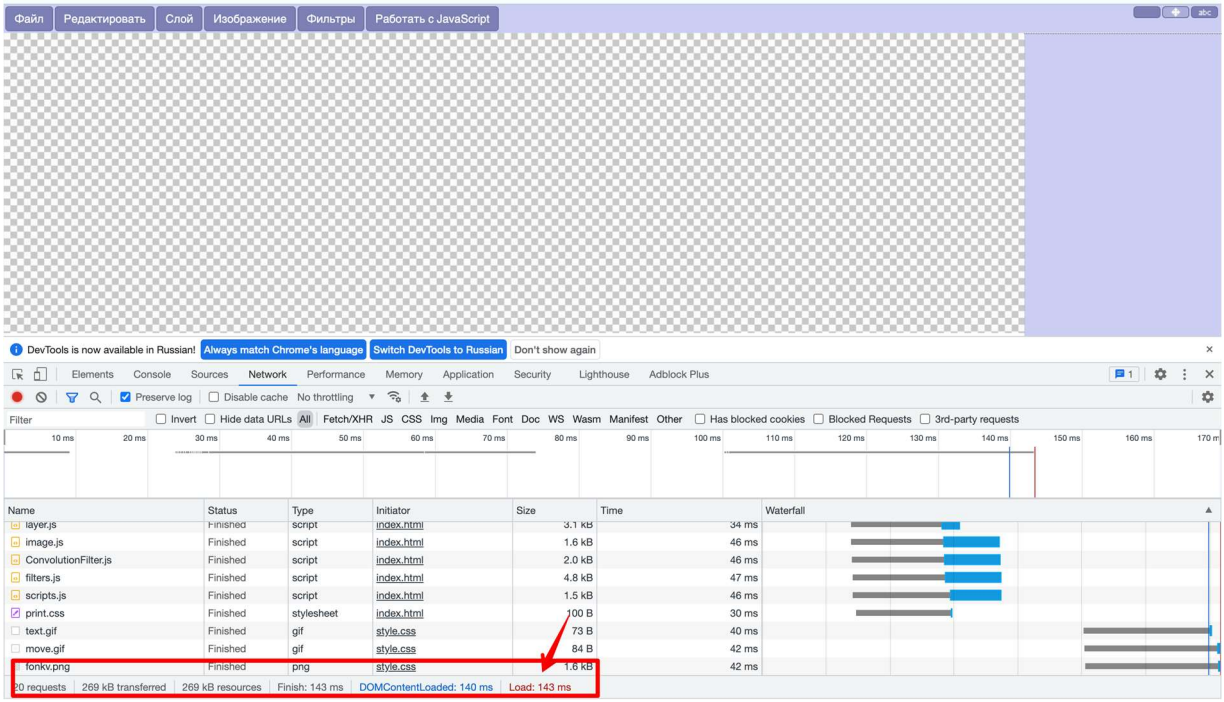

Рис. 9. Загрузка разработанного веб-приложения
Fig. 9. Loading the developed web application

Открытие и загрузка изображений для редактирования также происходит быстрее посредством разработанного веб-приложения (рис. 10). Загрузка изображения на веб-сайте «Photoshop Online» длилась 26 мс, загрузка изображения в разработанном веб-приложении – 8 мс.

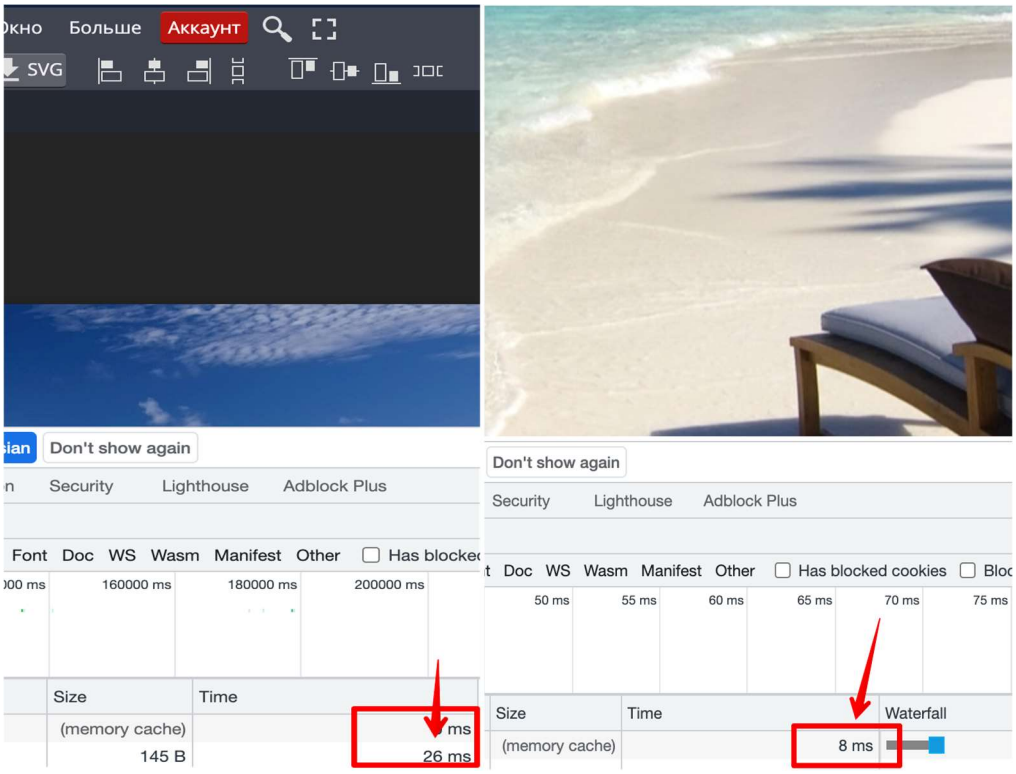

Рис. 10. Загрузка изображения на ресурсы
Fig. 10. Uploading an image to resources

Таким образом, эффективность данного алгоритма доказана на практике. Сравнительный анализ показал, что посредством разработанного алгоритма загрузка веб-приложения и изображения для обработки происходит намного быстрее, чем у приложений-аналогов. Полученные результаты будут особенно актуальны при обработке изображений больших объемов, когда время загрузки и обработки являются важными факторами, а в некоторых случаях и критическими.

Заключение

Представленный выше алгоритм работы веб-приложения является, в некотором смысле, уникальным и позволяет обрабатывать изображения различного объема без нагрузки на операционную систему, т.е. обладает относительно низкими характеристиками по энергозатратам на операционную систему и устройство в целом. Уникальность и новизна алгоритма заключаются в интеграции в разработанный код библиотеки EaselJS. Данная библиотека использует минимизированный файл js, который при загрузке браузера обрабатывается намного быстрее, чем обычный, не минимизированный файл js.

При разработке алгоритма работы веб-приложения были использованы специальные конструкции библиотеки EaselJS, которые сокращают объем реализованных функций и методов, например конструктор ColorFilter, который передает параметры RGB для цветового фильтра.

Таким образом, можно сделать вывод, что представленные алгоритмы позволяют повысить эффективность обработки изображений за счет сокращения времени загрузки приложения и самой обработки, а также за счет снижения нагрузки на операционную систему, что особенно важно, когда речь идет об обработке изображений больших объемов.

Список литературы

1. Виничук, О. Н. Разработка методов и алгоритмов цифровой обработки изображений : автореф. дис. магистра тех наук : 1-39 80 02 / О. Н. Виничук – Минск : БГУИР, 2017. – 14 с.
2. Виничук, О. Н. Веб-приложение для обработки изображений больших объемов / О. Н.Виничук – Минск : ГУ «БелИСА», 2020. – С. 23–32.
3. CoderLessons.com [Электронный ресурс]. – Режим доступа: https://coderlessons.com/articles/veb-razrabotka-articles/ispolzovanie-createjs-easeljs. – Дата доступа: 10.02.2022.
4. Флэнаган, Д. JavaScript. Подробное руководство / Д. Флэнаган., пер. с англ. – СПб : Символ-Плюс, 2008. – С. 122–581.
5. Буч, Г. Язык UML. Руководство пользователя. / Г. Буч, Д. Рамбо, И. Якобсон. 2-е изд.: пер. с англ. Мухин Н. – М. : ДМК Пресс, 2006. – С. 281–300.

References

1. Vinichuk, O. N. Development of methods and algorithms for digital image processing: author. dis. ... master of tech. sciences : 1-39 80 02 / O. N. Vinichuk – Minsk : BSUIR, 2017. – 14 p.
2. Vinichuk, O. N. Web application for large-scale image processing / O. N. Vinichuk – Minsk : GU "BelISA", 2020. – P. 23–32.
3. CoderLessons.com [Electronic resource]. – Access mode: https://coderlessons.com/articles/veb-razrabotka-articles/uselzovanie-createjs-easeljs. – Access date: 02.10.2022.
4. Flanagan, D. JavaScript. Detailed guide / D. Flanagan / trans. from English. – St. Petersburg : Symbol-Plus, 2008. – P. 122–581.
5. Butch, G. Language UML. User guide. / G. Butch, D. Rambo, I. Jacobson. 2nd ed.: tr. from English. Mukhin N. – M .: DMK Press, 2006. – P. 281–300.

Вклад авторов / Authors' contribution

Авторы внесли равный вклад в написание статьи.

All authors have equally contributed to the writing of the article.

Сведения об авторах

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

Дравица В. И., к. ф.-м. н, директор Научно-инженерного республиканского унитарного предприятия «Межотраслевой научно-практический центр систем идентификации и электронных деловых операций».

Information about the authors

Vinichuk O. N., Lecturer at the Branch "Minsk Radio Engineering College" of the Belarusian State University of Informatics and Radioelectronics.

Dravitsa V. I., Cand. of Sci., Head of the Scientific Engineering Republican Unitary Enterprise «Interbranches Research Development Centre for Identification Systems and e-Business Operations».

Адрес для корреспонденции

220013, Республика Беларусь,
Минск, пр. Независимости, 62,
Белорусский государственный университет информатики и радиоэлектроники, филиал «Минский радиотехнический колледж»
тел. +375-29-341-04-17;
e-mail: memory1703@gmail.com
Виничук Ольга Николаевна

Address for correspondence

220013, Republic of Belarus,
Minsk, Independence Ave., 62,
Belarusian State University of Informatics and Radioelectronics, Branch "Minsk radioengineering college"
tel. +375-29-341-04-17;
e-mail: memory1703@gmail.com
Vinichuk Olga Nikolaevna