УДК 004.738.5 https://doi.org/10.33619/2414-2948/126/23

СОВРЕМЕННЫЕ ПОДХОДЫ К ОПТИМИЗАЦИИ CSS В RESPONSIVE-ДИЗАЙНЕ ВЕБ-ПРИЛОЖЕНИЙ

©Омаралиева Г. А., *ORCID: 0000-0003-1862-2142, SPIN-код: 4741-5012, канд. физ.-мат. наук, Ошский государственный университет, г. Ош, Кыргызстан, gulya@oshsu.kg*
©Абдумиталип уулу К., *ORCID: 0009-0000-5208-0741, SPIN-код: 4476-7149, канд. физ.-мат. наук, Ошский государственный университет, г. Ош, Кыргызстан, kuba@oshsu.kg*
©Эргешов Т. А., *ORCID: 0009-0001-6077-170X, Ошский государственный университет, г. Ош, Кыргызстан, tilek120783@gmail.com*
©Исламбек кызы А., *ORCID: 0009-0006-1360-8367, Ошский государственный университет, г. Ош, Кыргызстан, aiturgan07072002@gmail.com*

MODERN APPROACHES TO CSS OPTIMIZATION IN RESPONSIVE WEB APPLICATION DESIGN

©Omaralieva G., *ORCID: 0000-0003-1862-2142, SPIN code: 4741-5012, Ph.D., Osh State University, Osh, Kyrgyzstan, gulya@oshsu.kg*
© Abdumitalip uulu K., *ORCID: 0009-0000-5208-0741, SPIN code: 4476-7149, Ph.D., Osh State University, Osh, Kyrgyzstan, kuba@oshsu.kg*
©Ergeshov T., *ORCID: 0009-0001-6077-170X, Osh State University, Osh, Kyrgyzstan, tilek120783@gmail.com*
©Islambek kyzy A., *ORCID: 0009-0006-1360-8367, Osh State University, Osh, Kyrgyzstan, aiturgan07072002@gmail.com*

Аннотация. Исследуются современные подходы к оптимизации каскадных таблиц стилей (CSS) в контексте responsive-дизайна веб-приложений. Рассматривается эволюция CSS-подходов от классических медиа-запросов до технологий CSS Grid Layout, Flexbox, CSS Custom Properties и Container Queries. Проведён анализ методов повышения производительности CSS, включающих выделение критических стилей, минификацию, удаление неиспользуемого кода и асинхронную загрузку стилей. Показано, что комплексное применение современных CSS-технологий позволяет сократить время загрузки веб-страниц на 40–60 % и существенно улучшить пользовательский опыт. Сформулированы практические рекомендации по выбору оптимизационных стратегий для проектов различного масштаба.

Abstract. The article examines modern approaches to CSS optimization in responsive web application design. The evolution from classical media queries to CSS Grid Layout, Flexbox, CSS Custom Properties, and Container Queries is explored. CSS performance enhancement methods are analyzed, including Critical CSS extraction, minification, unused code removal, and asynchronous style loading. It is demonstrated that the comprehensive application of modern CSS technologies can reduce web page loading time by 40–60% and significantly improve user experience. Practical recommendations for optimization strategy selection are formulated.

Ключевые слова: CSS, responsive-дизайн, оптимизация производительности, адаптивная вёрстка, медиа-запросы, CSS Grid, Flexbox, Container Queries, Critical CSS, Core Web Vitals.

Т. 12. №5 2026
https://doi.org/10.33619/2414-2948/126

Keywords: CSS, responsive design, performance optimization, adaptive layout, media queries, CSS Grid, Flexbox, Container Queries, Critical CSS, Core Web Vitals.

Стремительное развитие мобильных технологий привело к фундаментальному пересмотру принципов проектирования веб-приложений. К началу 2025 года доля мобильных устройств в глобальном интернет-трафике превысила 60%, а количество уникальных разрешений экранов исчисляется тысячами. Каскадные таблицы стилей (CSS) занимают центральное место в обеспечении корректного отображения и производительности веб-приложений на различных устройствах. Концепция responsive-дизайна, сформулированная Итаном Маркоттом в 2010 году, предполагает создание веб-интерфейсов, автоматически адаптирующихся к характеристикам устройства. С момента её появления технологический ландшафт CSS претерпел существенные изменения: появились модули Grid Layout и Flexbox, CSS Custom Properties и Container Queries, однако расширение возможностей привело и к усложнению кодовых баз, проблемам избыточности стилей и блокирующей загрузки ресурсов.

Актуальность исследования обусловлена тем, что неоптимизированный CSS-код непосредственно влияет на метрики Core Web Vitals — LCP, FCP и CLS, учитываемые поисковыми системами. Увеличение времени загрузки на одну секунду может снижать конверсию на 7% и сокращать повторные посещения на 50% [6, 8].

Целью настоящей работы является систематизация и сравнительный анализ современных подходов к оптимизации CSS в responsive-дизайне, включая эволюцию CSS-подходов, оптимизационный потенциал новых технологий и методы повышения производительности доставки стилевых ресурсов.

Основополагающей работой в области адаптивного веб-дизайна является монография “Responsive Web Design” (2011), впервые сформулировавшая целостную концепцию адаптивного подхода на основе гибких сеток, гибких изображений и медиа-запросов [1].

Практическое развитие этих идей нашло отражение в работе “Responsive Web Design with HTML5 and CSS” (2020), демонстрирующей применение Flexbox и Grid Layout для построения адаптивных макетов с акцентом на производительности. Фундаментальный вклад в изучение производительности внесли С. Саудерс, систематизировавший правила оптимизации фронтенда в работах “High Performance Web Sites” (2007) и “Even Faster Web Sites” (2009), а также Grigorik I., описавший модель критического пути рендеринга в монографии “High Performance Browser Networking” (2013), где CSS-ресурсы играют ключевую роль, блокируя построение дерева рендеринга [2].

На раннем этапе существования Всемирной паутины веб-страницы проектировались с расчётом на фиксированное разрешение экрана — 800×600 или 1024×768 пикселей. Макеты строились на абсолютных единицах и табличной разметке, что исключало адаптацию к устройствам с иными характеристиками. Появление мобильных браузеров обнажило несостоятельность данного подхода [1].

Первым шагом к адаптивности стало введение медиа-запросов в CSS3, позволяющих применять различные стили в зависимости от характеристик устройства:

```
@media screen and (max-width: 768px) {
  .container { flex-direction: column; padding: 1rem; }
  .sidebar { display: none; }
}
```

Медиа-запросы позволили определять контрольные точки (breakpoints) — пороговые значения ширины экрана, при которых макет перестраивается. Параллельно происходил переход от абсолютных единиц к относительным (проценты, em, rem, vw, vh), что позволило

Т. 12. №5 2026
https://doi.org/10.33619/2414-2948/126

создавать гибкие сетки с пропорциональным изменением размеров элементов. Существенное влияние оказал подход mobile-first, при котором базовые стили разрабатываются для мобильных устройств, а для крупных экранов добавляются через медиа-запросы с min-width. Этот подход обеспечивает приоритизацию контента для мобильных пользователей и оптимизацию производительности за счёт минимальной начальной загрузки стилей.

Несмотря на достоинства, классический подход обнаружил ряд ограничений. Медиа-запросы ориентированы на глобальные характеристики viewport, а не на размер конкретного компонента, что препятствует созданию переиспользуемых модулей. Каскадный характер CSS при большом числе медиа-запросов усложняет сопровождение, а фрагментация устройств (от смарт-часов до телевизоров) делает фиксированный набор breakpoints недостаточным [3].

Данные ограничения послужили стимулом для разработки новых CSS-технологий. В период с 2017 по 2024 год появился ряд технологий, принципиально изменивших подход к построению адаптивных макетов. Модуль CSS Flexible Box Layout (Flexbox, рекомендация W3C 2018) предназначен для одномерного распределения пространства между элементами [4].

Flexbox заменяет целые группы правил, ранее необходимых для центрирования и выравнивания, что сокращает размер стилевого файла и нагрузку на парсер браузера. Модуль CSS Grid Layout (рекомендация W3C 2017) предоставляет инструментарий для двумерной компоновки. Существенным преимуществом Grid является возможность создания адаптивных сеток без медиа-запросов — посредством функций repeat(), minmax() и ключевого слова auto-fit:

```
.product-grid {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(280px, 1fr));
  gap: 1.5rem;
}
```

Данная конструкция полностью устраняет необходимость в медиа-запросах для компонента: столбцы автоматически добавляются или удаляются в зависимости от ширины контейнера. Переход с классических grid-систем на основе float на CSS Grid Layout позволяет сократить объём стилевого кода макета на 40–60%.

Таблица 1

СРАВНИТЕЛЬНАЯ ХАРАКТЕРИСТИКА СОВРЕМЕННЫХ CSS-ТЕХНОЛОГИЙ

Технология	*Тип адаптации*	*Сокращение объёма CSS, %*	*Необходимость медиа-запросов*	*Поддержка браузерами (2024), %*
Flexbox	Одномерная	20–30	Требуются для макета	99+
CSS Grid Layout	Двумерная	40–60	Частично устраняет	97+
CSS Custom Properties	Параметрическая	15–25	Упрощают использование	98+
Container Queries	Компонентная	30–50	Заменяет для компонентов	91+

CSS Custom Properties (переменные) позволяют определять именованные значения, вычисляемые на этапе выполнения и участвующие в каскаде, что делает их инструментом для создания адаптивных дизайн-систем с централизованным управлением параметрами через медиа-запросы. Наиболее значимым нововведением стала спецификация Container Queries (CSS Containment Level 3, поддержка во всех браузерах к 2023 году), решающая ключевую проблему классических медиа-запросов — привязку к viewport. Вместо этого стили компонента определяются размером его контейнера, что позволяет создавать по-настоящему

Т. 12. №5 2026
https://doi.org/10.33619/2414-2948/126

переиспользуемые модули. Команда Netflix отметила переход к восходящему подходу к компонентному дизайну и устранение JavaScript-зависимостей, а redBus объединила мобильную и десктопную кодовые базы в единую систему.

Совместное применение технологий создаёт синергетический эффект: переход от float-сетки к комбинации CSS Grid, Custom Properties и Container Queries сокращает объём кода макета более чем в 2,5 раза.

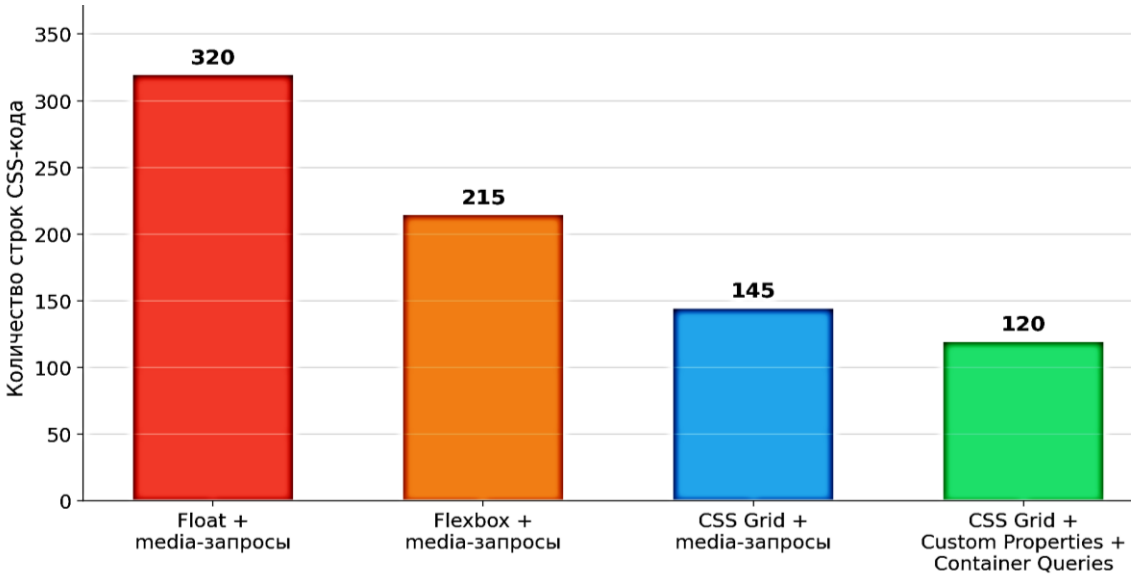

Рисунок 1. Сравнение объёма CSS-кода при различных подходах к построению адаптивного макета

Методы оптимизации производительности CSS. Браузерная модель рендеринга предполагает, что CSS является ресурсом, блокирующим отрисовку: пока все стилевые файлы не загружены и не проанализированы, браузер не приступит к построению дерева рендеринга. Поэтому объём, структура и способ доставки CSS непосредственно влияют на время первой отрисовки контента (FCP) и воспринимаемую скорость загрузки. Выделение критических стилей (Critical CSS) — один из наиболее эффективных методов: минимальный набор правил для отрисовки видимой части страницы встраивается в HTML-документ, а остальные стили загружаются асинхронно. Для автоматизации используются инструменты critical и Penthouse. Применение Critical CSS сокращает время FCP на 25–40%. Минификация CSS (cssnano, clean-css) — удаление пробелов, комментариев, сокращение значений — обеспечивает уменьшение объёма на 15–25%, а в сочетании с GZIP/Brotli-сжатием — до 80–90%. Удаление неиспользуемого CSS (PurgeCSS, UnCSS) — наиболее радикальный метод: для проектов с CSS-фреймворками он позволяет сократить объём стилей на 70–95%. Разделение стилей по маршрутам (route-based CSS splitting) — формирование отдельного стилевого файла для каждой страницы — сокращает начальную загрузку на 50–70% для крупных одностраничных приложений. Платформа PostCSS с плагинами Autoprefixer и postcss-preset-env стала стандартом автоматизированной обработки стилей. Эффективность оптимизации оценивается через метрики Core Web Vitals: Largest Contentful Paint (LCP, целевое значение — менее 2,5 с) зависит от скорости загрузки CSS; Cumulative Layout Shift (CLS, менее 0,1) связан с корректностью стилей и стабильностью макета; Interaction to Next Paint (INP) измеряет отзывчивость страницы, на которую могут влиять сложные CSS-анимации и частый пересчёт стилей.

Т. 12. №5 2026
https://doi.org/10.33619/2414-2948/126

Таблица 2

ВЛИЯНИЕ МЕТОДОВ ОПТИМИЗАЦИИ CSS НА МЕТРИКИ CORE WEB VITALS

Метод оптимизации	*Влияние на LCP*	*Влияние на CLS*	*Влияние на INP*	*Сложность внедрения*
Critical CSS	5	2	1	Средняя
Минификация CSS	3	1	1	Низкая
Удаление неиспользуемого CSS	4	1	2	Средняя
Разделение по маршрутам	4	2	1	Высокая
Переход на CSS Grid	2	4	3	Средняя
Container Queries	2	4	2	Средняя
Комплексный подход	5	5	4	Высокая

Для количественной оценки использован корпоративный веб-сайт (12 страниц) на Bootstrap 4 с float-сеткой. Оптимизированная версия использует CSS Grid, Flexbox, Custom Properties, Critical CSS, PurgeCSS и cssnano. Измерения выполнены Google Lighthouse в режиме эмуляции мобильного устройства (4G).

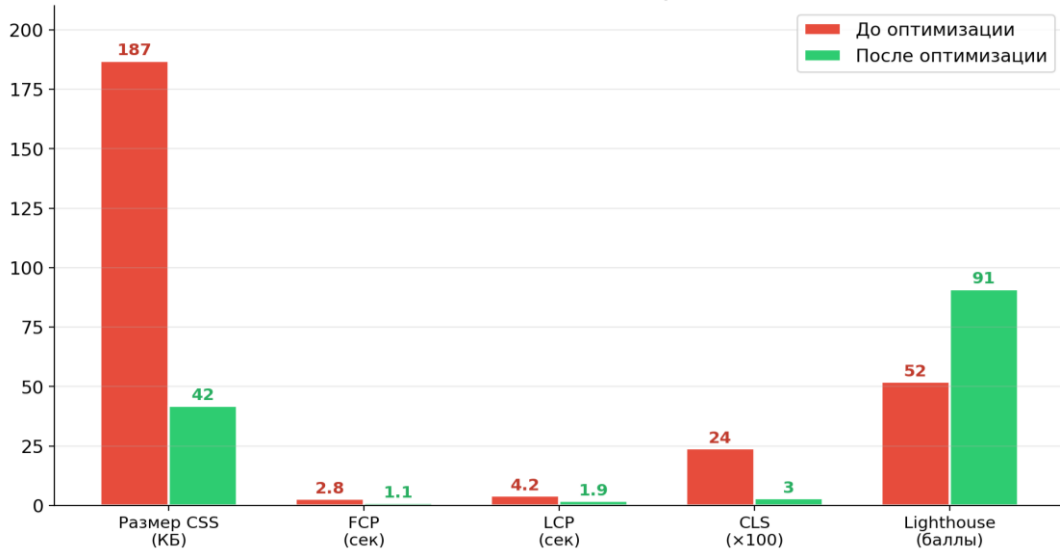

Рисунок 2. Сравнение метрик производительности до и после комплексной оптимизации CSS

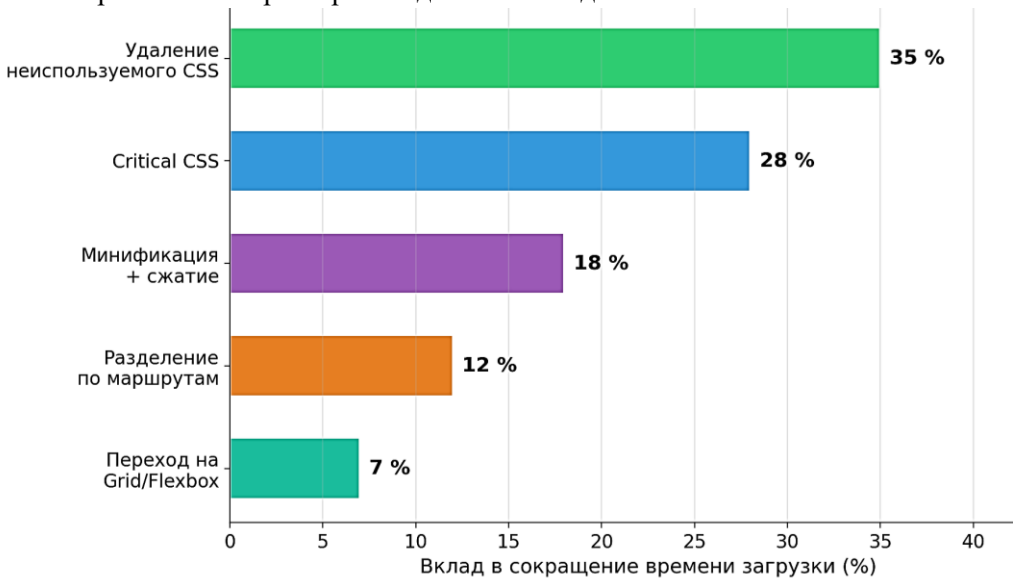

Рисунок 3. Вклад отдельных методов оптимизации CSS в сокращение времени загрузки страницы

Т. 12. №5 2026
https://doi.org/10.33619/2414-2948/126

Объём CSS сократился с 187 КБ до 42 КБ (−77,5%), FCP – с 2,8 до 1,1 с (−60,7%), LCP – с 4,2 до 1,9 с (−54,8%), CLS – с 0,24 до 0,03 (−87,5%). Балл Lighthouse вырос с 52 до 91.

Наибольший вклад вносит удаление неиспользуемого CSS (35%) и Critical CSS (28%). Минификация со сжатием даёт 18%, разделение по маршрутам — 12%, переход на Grid/Flexbox — 7% [6, 7].

Для небольших проектов достаточно минификации и CSS Grid/Flexbox; для средних — дополнительно Critical CSS, PurgeCSS и Custom Properties; для крупных SPA рекомендуется полный комплекс с разделением по маршрутам и мониторингом Core Web Vitals.

Заключение

В настоящей работе проведён систематический анализ современных подходов к оптимизации CSS в responsive-дизайне веб-приложений. Показано, что эволюция от медиа-запросов до CSS Grid Layout, Flexbox, Custom Properties и Container Queries позволяет сократить объём стилевого кода на 40–60%, а комплексное применение методов оптимизации производительности (Critical CSS, минификация, удаление неиспользуемого кода, разделение по маршрутам) обеспечивает улучшение метрик FCP на 60,7%, LCP на 54,8% и CLS на 87,5%. Сформулированы дифференцированные рекомендации по выбору стратегий в зависимости от масштаба проекта. Перспективы развития связаны со спецификацией CSS Nesting, позволяющей организовывать правила иерархически без препроцессоров, директивой @layer для управления приоритетом стилей и интеграцией искусственного интеллекта в инструменты оптимизации — автоматическим определением критических стилей и интеллектуальным разделением файлов на основе анализа пользовательского поведения.

Список литературы:

1. Marcotte, E. Responsive Web Design. New York: A Book Apart, 2011. 153 p.
2. Grigorik I. High Performance Browser Networking: What every web developer should know about networking and web performance. O'Reilly Media, Inc., 2013.
3. Souders S. Even faster web sites: performance best practices for web developers. O'Reilly Media, Inc. 2009.
4. World Wide Web Consortium. CSS Flexible Box Layout Module Level 1: W3C Recommendation. 2018. https://www.w3.org/TR/css-flexbox-1/
5. Кыштообаева Ч. Роль компьютерных технологий в повышении качества образования учащихся //Вестник Ошского государственного университета. 2023. Т. 3. С. 51-58.
6. Макфарланд Д. Новая большая книга CSS. СПб.: Питер, 2016. 720 с.
7. Молдояров У. Д., Сейитказыева Г. И., Ажибекова А. Т. Сравнительный анализ фреймворков при разработке динамических адаптивных сайтов // Вестник Ошского государственного педагогического университета имени А. Мырсабекова. 2022. №1-1(19). С. 156-160.
8. Логинова Н. В. Методы оптимизации производительности web-приложений // Наука и образование сегодня. 2024. №1 (78). С. 4.

References:

1. Marcotte, E. Responsive Web Design. New York: A Book Apart, 2011. 153 p.
2. Grigorik, I. (2013). *High Performance Browser Networking: What every web developer should know about networking and web performance*. O'Reilly Media, Inc.
3. Souders, S. (2009). *Even faster web sites: performance best practices for web developers*. O'Reilly Media, Inc.

4. World Wide Web Consortium. CSS Flexible Box Layout Module Level 1: W3C Recommendation. 2018. https://www.w3.org/TR/css-flexbox-1/

5. Kyshtoobaeva, Ch. (2023). Rol′ komp′yuternykh tekhnologij v povyshenii kachestva obrazovaniya uchashchikhsya. *Vestnik Oshskogo gosudarstvennogo universiteta, 3*, 51-58. (in Russian).

6. Makfarland, D. (2016). Novaya bol′shaya kniga CSS. St. Petersburg. (in Russian).

7. Moldoyarov, U. D., Sejitkazyeva, G. I., & Azhibekova, A. T. (2022). Sravnitel′nyj analiz frejmvorkov pri razrabotke dinamicheskikh adaptivnykh sajtov. *Vestnik Oshskogo gosudarstvennogo pedagogicheskogo universiteta imeni A. Myrsabekova,* (1-1(19)), 156-160. (in Russian).

8. Loginova, N. V. (2024). Metody optimizatsii proizvoditel′nosti web-prilozhenij. *Nauka i obrazovanie segodnya,* (1 (78)), 4. (in Russian).

Поступила в редакцию
19.02.2026 г.

Принята к публикации
05.03.2026 г.

Ссылка для цитирования:

Омаралиева Г. А., Абдумиталип уулу К., Эргешов Т. А., Исламбек кызы А. Современные подходы к оптимизации CSS в responsive-дизайне веб-приложений // Бюллетень науки и практики. 2026. Т. 12. №5. С. 190-196. https://doi.org/10.33619/2414-2948/126/23

Cite as (APA):

Omaralieva, G., Abdumitalip uulu, K., Ergeshov, T., & Islambek kyzy, A. (2026). Modern Approaches to CSS Optimization in Responsive Web Application Design. *Bulletin of Science and Practice, 12*(5), 190-196. (in Russian). https://doi.org/10.33619/2414-2948/126/23