ISSN 2786-5371 print
ISSN 2786-538X online
Технології та інжиніринг, № 4(9), 2022

Інформаційні технології, електроніка, механічна та електрична інженерія
Information technologies, electronics, mechanical and electrical engineering

https://doi.org/10.30857/2786-5371.2022.4.1

УДК 004.51

БАШОВИЙ В. М., СТАЦЕНКО В. В., СТАЦЕНКО Д. В.
Київський національний університет технологій та дизайну, Україна

ВИЗНАЧЕННЯ ШВИДКОСТІ РОБОТИ СУЧАСНИХ ФРЕЙМВОРКІВ ДЛЯ СТВОРЕННЯ WEB-ІНТЕРФЕЙСІВ

Мета. *Дослідження швидкості формування списків інтерактивних елементів сучасними фреймворками, що використовуються для створення web-інтерфейсів.*

Методика. *Використовуються методи опису та аналізу ефективності програмних засобів, методи та інструменти дослідження web-додатків.*

Результати. *Визначено найбільш популярні JavaScript фреймоврки, що використовуються для створення web-інтерфейсів. До них відносяться фреймворки React, Angular та Vue. Визначено час, що витрачають зазначені фреймоврки на виконання операцій зі списком інтерактивних елементів. Встановлено, що найкращі результати показав фреймворк Angular: загальний час на виконання найбільш складної операції (повного формування сторінки) у нього на 45% краще ніж у React та на 44% краще ніж у Vue. Встановлено, що фреймворки мають різну архітектуру, що може змінювати час роботи в інших додатках. Зокрема, тестування швидкості видалення задачі показало, що фреймоврк Angular значно менше часу витрачає на етапі Scripting ніж React та Vue, але значно більше – на етапі рендерінгу. Це свідчить про інший принцип підготовки даних для елементів web додатку.*

Наукова новизна. *Представлено результати дослідження швидкості роботи JavaScript фреймворків для створення web-додатків. Проаналізовано час виконання операцій формування списків інтерактивних елементів, їх зміни та видалення.*

Практична значимість. *Представлена інформація дозволяє зробити обґрунтований вибір фреймворку для створення web-додатків.*

Ключові слова: *web-додаток; web-інтерфейс; JavaScript; фреймворк; Angular; React; Vue.*

Вступ. Веб-додатки сьогодні є одним з найпоширеніших видів програмного забезпечення, що використовуються для вирішення різноманітних задач. Їх стрімке розповсюдження зумовлено передусім низькими вимогами до налаштувань оточення користувача, оскільки вони потребують лише наявності браузера та доступу до мережі на комп'ютері користувача. Таким чином, середовищем для виконання таких додатків є браузер в якому відбувається формування інтерфейсу користувача. Обробка та збереження даних в переважній кількості випадків здійснюється на сервері. Це дозволяє уникнути процедур встановлення та налаштування програмного забезпечення на комп'ютері користувача, дозволяє використовувати один й той самий додаток на стаціонарних комп'ютерах та мобільних пристроях. Водночас, використання браузера в якості платформи для запуску додатків вносить свої обмеження. Передусім браузер суттєво обмежує доступ до ресурсів операційної системи, що робить неможливим створення web-додатків, які можуть замінити системне програмне забезпечення. Інша проблема пов'язана з відсутністю стандартних компонентів для створення інтерфейсів користувача. Інтерфейс формується розробником з використанням мов HTML, JavaScript та CSS [1]. Вони дозволяють створювати достатньо складні інтерфейси, але їх безпосереднє використання призводить до надмірних витрат часу, особливо при створенні додатків зі схожими елементами управління та відображення інформації. Це зумовило появу великої кількості бібліотек та фреймворків, які дозволяють суттєво скоротити час розробки web-додатків за рахунок використання готових компонентів. З іншої сторони, розробники фреймворків намагаються зробити їх універсальними, що зумовлює розширення кодової бази та додаткові витрати ресурсів. Це зумовлює актуальність дослідження швидкості роботи розповсюджених фреймворків для створення web-інтерфейсів.

Постановка завдання. Сьогодні існує декілька фреймворків, які широко використовуються для розробки web-додатків, та, водночас, побудовані за різними архітектурними принципами.

Метою дослідження є визначення швидкості їх роботи при формуванні списків інтерактивних елементів. Такі списки використовуються у багатьох web-додатків, зокрема, для відображення інформації, що надходить з бази даних, з якою користувач може виконувати певні дії (редагування, видалення тощо). Кожен елемент списку містить певні дані та елементи керування (кнопки, текстові поля, перемикачі тощо). Web-додаток має сформувати сторінку зі списком та забезпечити обробку взаємодії користувача з елементами керування.

Результати дослідження. Сьогодні основною мовою для створення додатків, що працюють в браузері, є JavaScript. Ця мова є єдиною, яка підтримується всіма сучасними браузерами, що зумовлює її надзвичайно широке поширення. Згідно рейтингу мов програмування за останні два роки по версії DOU.ua мова JavaScript займає 1-ше місце [13]. Також існує декілька мов, які також використовуються для створення web-інтерфейсів, наприклад, TypeScript та CoffeScript. Вони мають певні переваги, але код написаний на цих мовах в будь-якому випадку компілюється у JavaScript перед виконанням у браузері. Таким чином, ці мови дозволяють використовувати інший синтаксис, але мають ті самі обмеження, що і JavaScript.

Вихідними ресурсами для побудови сторінки web-додатку є HTML розмітка, JavaScript код та таблиці стилів (CSS). Всі ці ресурси браузер завантажує з сервера, після чого запускає процес формування сторінки, що включає наступні етапи:

- Завантаження – відправка запитів на сервер, отримання відповідей та їх парсинг.
- Очікування компіляції скриптів (Idle).
- Виконання JavaScript коду (Scripting).
- Рендерінг (Rendering) – формування сторінки в оперативній пам’яті.
- Відображення сторінки у вікні браузера (Painting).
- Виконання системних операцій (System).

Порядок виконання цих етапів визначається браузером та додатком. Наприклад, JavaScript код може змінювати HTML розмітку, що призводить до повторного виконання етапів Rendering та Painting.

Фреймворки, що розглядаються, в рамках цього дослідження, виконують формування сторінки на основі вихідних даних та шаблонів, що задані розробником. Це зменшує час на розробку додатку оскільки всі операції низького рівня виконує фреймоворк.

На рис. 1 наведено тренди популярності найбільш розповсюджених JavaScript фреймворків. Відповідно до нього, найбільш популярними є React, Vue, Angular, які ми розглядаємо в рамках цього дослідження.

З метою дослідження швидкості їх роботи було обрано відомий додаток TodoMVC [2], реалізований на мові JavaScript з використанням всіх зазначених фреймворков. Цей додаток дозволяє створювати та редагувати список задач, змінювати їх статуси, переглядати списки нових та завершених задач, виконувати інші операції. Інтерфейс TodoMVC показаний на рис. 2.

Не зважаючи на те, що додаток є відносно простим, він включає більшість типових операцій, які виконує користувач з даними, а саме: створення нової задачі, її редагування, зміну статусу, видалення, групування задач за їх статусами, виконання групових операцій (видалення виконаних задач).

Станом на сьогодні створено декілька десятків версій TodoMVC, на основі різних бібліотек та фреймворків, що робить цей додаток надзвичайно зручним для порівняння та вибору JavaScript бібліотек.

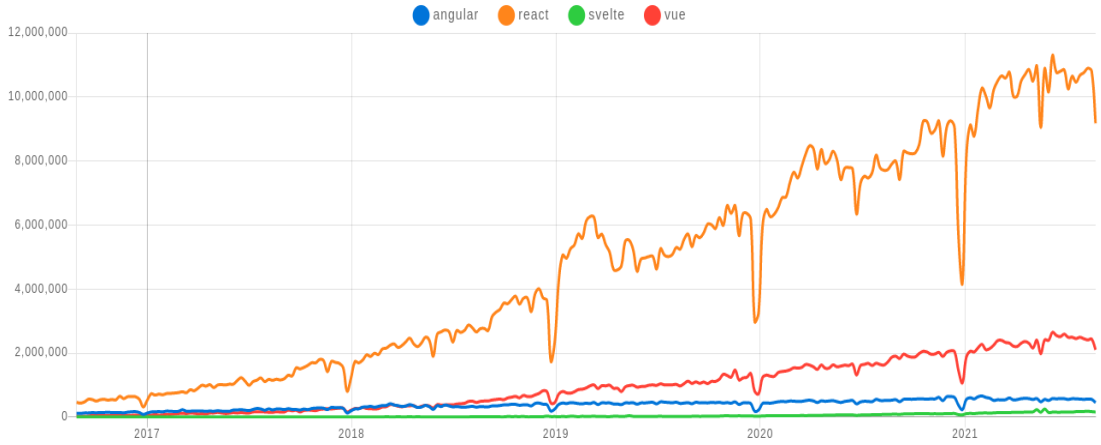

Рис. 1. **Рейтинг використання фреймворків 2021–2022 рр.**

Рис. 2. **Інтерфейс web-додатку TodoMVC**

Іншою причиною, яка зумовила вибір TodoMVC для проведення порівняння є те, що задачі зберігаються у локальному сховищі браузера. Більш розповсюдженою практикою є завантаження даних для додатку з сервера. Але в цьому випадку час оновлення інтерфейсу додатку включатиме час передачі даних між браузером та сервером, що ускладнює порівняння JavaScript фреймворків. Завантаження даних з локального сховища браузера здійснюється з постійною швидкістю.

Кожна з задач являє собою хеш з наступними даними:

```
{
    id: 123,
    title: "задача 123",
    completed: false
}
```

де `id` – унікальний ідентифікатор задачі;
`title` – текст задачі;
`completed` – статус виконання.

У роботі досліджувалось три версії додатку TodoMVC, створені на базі фреймворків React, Angular та Vue.

Перед початком дослідження було створено та розміщено в локальному сховищі браузера 10000 задач. Така кількість елементів практично гарантовано уповільнює роботу будь-якого web-додатку, що дозволяє підвищити точність визначення часу, який витрачають фреймворки на формування сторінки, за рахунок зменшення відносної похибки вимірювань, що виникає внаслідок дії зовнішніх факторів. Зокрема, інші програми можуть в певні моменти часу використовувати системні ресурси, що уповільнює роботу браузера та впливає на результати вимірювань.

Дослідження проводилось із використанням браузера Google Chrome версії 105.0.5195.102. Браузер запускався в режимі «інкогніто», що гарантувало відключення всіх додатків.

Визначення тривалості виконання операцій додатком здійснювалось з використанням вбудованих у Google Chrome інструментів розробника (Chrome Development Tools). Вони дозволяють вимірювати час та використання ресурсів на формування сторінки додатку. Також інструменти розробника дозволяють визначати час формування сторінки та час на протязі якого інтерфейс не реагує на дії користувача (Total blocking time). Приклад звіту з цими даними показано на наступному рис. 3.

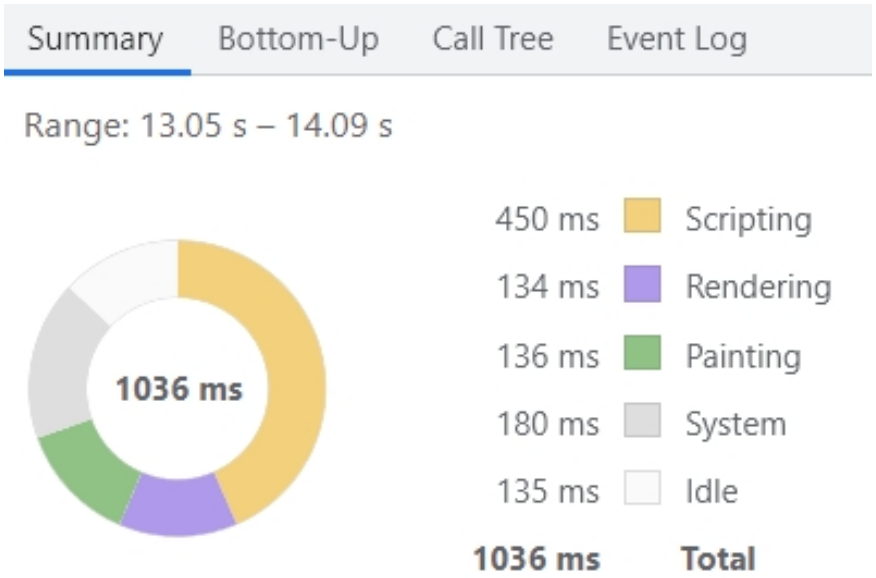

Рис. 3. **Звіт з інформацією про тривалість формування сторінки, сформований у Chrome Development Tools**

У роботі проведено дослідження трьох операції.

1) Формування сторінки та первинне завантаження списку. Це найбільш ресурсоємна операція під час якої сторінка формується «з нуля». На цьому етапі у фреймворка немає можливості використати попередньо сформовані елементи. Окрім часу додатково визначалось максимальне використання оперативної пам’яті додатком та кількість переданих даних (сумарний розмір фреймворку та додатку). Результати представлені у табл. 1.

Об’єм пам’яті, що використовується системою, є практично однаковим для всіх фреймворків. За кількістю переданих даних, найкращий показник має Vue. Водночас, мінімальний час формування сторінки забезпечив фреймворк Angular.

Таблиця 1

Результати дослідження процесу формування сторінки

Параметр	Фреймворк		
	React	**Angular**	**Vue**
Загальний час, мс	4203	3158	3692
Total blocking time, мс	3609	2532	3111
Завантаження сторінки, мс	241	5	4
Scripting, мс	2636	1554	1311
Rendering, мс	710	723	819
Painting, мс	13	13	12
Системні операції, мс	369	653	1307
Очікування, мс	235	211	239
Максимальне використання пам’яті, МБ	132	150	126
Кількість переданих даних, кБ	1207	1723	267

2) Зміна статусу задачі. Ця операція здійснюється за допомогою перемикача, що розташований ліворуч від тексту задачі (рис. 2). При цьому змінюється відображення лише одного елементу списку. Результати представлені у табл. 2.

Таблиця 2

Результати дослідження зміни статусу задачі

Параметр	Фреймворк		
	React	**Angular**	**Vue**
Загальний час, мс	947	966	1036
Scripting, мс	342	334	450
Rendering, мс	172	150	134
Painting, мс	101	129	136
Системні операції, мс	247	215	180
Очікування, мс	84	137	135

Всі фреймворки виконали цю операцію практично за однаковий час. Трохи кращий результат у React. Також слід відзначити, що тривалість виконання цієї операції значно менша за час формування сторінки. Це свідчить про те, що в усіх фреймворках процес рендерингу є оптимізованим і зміни в одному з компонентів не призводять до повторного формування всієї сторінки.

3) Видалення задачі зі списку. Ця операція призводить до зміни кількості елементів у списку. При цьому задача, що видаляється, зникає зі сторінки, а всі задачі, що розташовані нижче, переміщуються на одну позицію вгору. Результати вимірювань представлені у табл.3.

Таблиця 3

Результати дослідження видалення задачі зі списку

Параметр	Фреймворк		
	React	**Angular**	**Vue**
Загальний час, мс	3647	1996	2039
Scripting, мс	2996	672	1576
Rendering, мс	254	1035	296
Painting, мс	14	13	11
Системні операції, мс	317	186	100
Очікування, мс	66	91	56

Час виконання цієї операції більший ніж у випадку зміни статусу задачі. Це пояснюється тим, що всі елементи списку які розташовані нижче задачі, що видаляється, мають бути переміщені. Найкращий час в даному випадку продемонстрували фреймворки Angular та Vue. Також в даному випадку чітко помітна різниця у розподіленні часу між етапами Scripting та Rendering, що свідчить про суттєві відмінності у архітектурі фреймворків.

Висновки.

1) В результаті проведеного дослідження визначено найбільш популярні JavaScript фреймоврки, що використовуються для створення web-інтерфейсів. До них відносяться фреймворки React, Angular та Vue. Визначено час, що витрачають зазначені фреймоврки на виконання операцій зі списком інтерактивних елементів.

2) Найкращі результати показав фреймворк Angular: загальний час на виконання найбільш складної операції (повного формування сторінки) у нього на 45% краще ніж у React та на 44% краще ніж у Vue.

3) Встановлено, що фреймворки мають різну архітектуру, що може змінювати час роботи в інших додатках. Зокрема, тестування швидкості видалення задачі показало, що фреймоврк Angular значно менше часу витрачає на етапі Scripting ніж React та Vue, але значно більше – на етапі рендерінгу. Це свідчить про інший принцип підготовки даних для елементів web додатку. Тому перевірка швидкості роботи має здійснюватись з урахуванням особливостей конкретного web додатку, що розробляється.

References

1. Meloni, J., Kyrnin, J. (2018). HTML, CSS, and JavaScript All in One. 3rd edition. Sams Publishing. 800 p.
2. Addendum TodoMVC: website. URL: https://todomvc.com/
3. Ohliad CSS-freimvorkiv: veb-sait [Overview of CSS Frameworks: website]. URL: http://iantonov.me/page/obzor-css-frejmvorkov.
4. Adaptyvna verstka: veb-sait [Responsive layout: website]. URL: https://webtune.com.ua/statti/web-rozrobka/adaptyvna-verstka-sajtu/#id2.
5. Klasyfikatsiia bibliotek: veb-sait [Classification of libraries: website]. URL: https://codeguida.com/post/2322.
6. Veb-interfeisy: veb-sait [Web Interfaces: Website]. URL: https://www.alltechbuzz.net/uk/the-most-essential-frontend-web-development-tools/.
7. SEO-perevah adaptyvnoho veb-dyzainu: veb-sait [SEO Benefits of Responsive Web Design: Website]. URL: https://lemarbet.com/ua/prodvizhenie-sajtov-internet-magazinov/7-osnovnih-seo-perevag-adaptivy/.
8. JavaScript (JS): veb-sait [JavaScript (JS): Website]. URL: https://www.techopedia.com/definition/3929/javascript-js.
9. Angular: veb-sait [Angular: Website]. URL: https://www.typescriptlang.org/docs/handbook/angular.html.
10. Adaptyvni CSS-freimvorky, sitky, klasy vydymosti: veb-sait [Responsive CSS frameworks, grids, visibility classes: website]. URL:http://klondike-studio.ru/blog/responsive- cssframework/.

Література

1. Meloni J., Kyrnin J. HTML, CSS, and JavaScript All in One. 3rd edition. Sams Publishing, 2018. 800 p.
2. Додаток TodoMVC: веб-сайт. URL: https://todomvc.com/
3. Огляд CSS-фреймворків: веб-сайт. URL: http://iantonov.me/page/obzor-css-frejmvorkov.
4. Адаптивна верстка: веб-сайт. URL: https://webtune.com.ua/statti/web-rozrobka/adaptyvna-verstka-sajtu/#id2.
5. Класифікація бібліотек: веб-сайт. URL: https://codeguida.com/post/2322.
6. Веб-інтерфейси: веб-сайт. URL: https://www.alltechbuzz.net/uk/the-most-essential-frontend-web-development-tools/.
7. SEO-переваг адаптивного веб-дизайну: веб-сайт. URL: https://lemarbet.com/ua/prodvizhenie-sajtov-internet-magazinov/7-osnovnih-seo-perevag-adaptivy/.
8. JavaScript (JS): веб-сайт. URL: https://www.techopedia.com/definition/3929/javascript-js.
9. Angular: веб-сайт. URL: https://www.typescriptlang.org/docs/handbook/angular.html.
10. Адаптивні CSS-фреймворки, сітки, класи видимості: веб-сайт. URL:http://klondike-studio.ru/blog/responsive- cssframework/.

11. Ohliad CSS-freimvorkiv: veb-sait [Overview of CSS Frameworks: website]. URL: http://iantonov.me/page/obzor-css-frejmvorkov.
12. Google Optimize: veb-sait [Google Optimize: website]. URL: https://pr-cy.ru/news/p/6336.
13. Pokaznyky populiarnosti freimvorkiv: veb-sait [Indicators of the popularity of frameworks: website]. URL: https://dou.ua/forums/topic/34739/.

11. Огляд CSS-фреймворків: веб-сайт. URL: http://iantonov.me/page/obzor-css-frejmvorkov.
12. Google Optimize: веб-сайт. URL: https://pr-cy.ru/news/p/6336.
13. Показники популярності фреймворків: веб-сайт. URL: https://dou.ua/forums/topic/34739/.

BASHOVYI VADYM
Graduate Student, Kyiv National University of Technologies and Design, Ukraine
https://orcid.org/0000-0001-7557-8841
E-mail: Andragorn1996@gmail.com

STATSENKO VOLODYMYR
Doctor of Technical Sciences, Associate Professor, Department of Computer Engineering and Electromechanics, Kyiv National University of Technologies and Design, Ukraine
https://orcid.org/0000-0002-3932-792X
Scopus Author ID: 57210344190
Researcher ID: C-3646-2017
E-mail: statsenko.v@knutd.edu.ua

STATSENKO DMITRY
Candidate of Technical Sciences, Associate professor, Department of Computer Engineering and Electromechanics Kyiv National University of Technologies and Design, Ukraine
https://orcid.org/0000-0002-3064-3109
Scopus Author ID: 57210341005
ResearcherID: C-3644-2017
E-mail: statsenko.dv@knutd.edu.ua

БАШОВЫЙ В. М., СТАЦЕНКО В. В., СТАЦЕНКО Д. В.
Киевский национальный университет технологий и дизайна, Украина

ОПРЕДЕЛЕНИЕ СКОРОСТИ РАБОТЫ СОВРЕМЕННЫХ ФРЕЙМВОРКОВ ДЛЯ СОЗДАНИЯ WEB-ИНТЕРФЕЙСОВ

Цель. *Исследование скорости формирования списков интерактивных компонентов современными фреймворками, которые используются для создания web-интерфейсов.*

Методика. *Используются способы описания и анализа эффективности программных средств, способы и инструменты исследования web-приложений.*

Результаты. *Определены наиболее популярные JavaScript фреймовркы, которые используются для создания web-интерфейсов. К ним относятся фреймворки React, Angular и Vue. Определено время, затрачиваемое указанными фреймоврками на выполнение операций со списком интерактивных компонентов. Установлено, что лучшие результаты показал фреймворк Angular: общее время для выполнения наиболее сложной операции (полного формирования страницы) у него на 45% лучше, чем у React и на 44% лучше, чем у Vue. Установлено, что фреймворки имеют разную архитектуру, что может изменять время работы в других приложениях. В частности, тестирование скорости удаления компонента показало, что фреймовркр Angular значительно меньше времени тратит на этапе Scripting, чем React и Vue, но гораздо больше – на этапе рендеринга. Это свидетельствует о разнице в алгоритмах подготовки данных для элементов web приложения.*

Научная новизна. *Представлены результаты исследования скорости работы JavaScript фреймворков для создания web-приложений. Проанализировано время выполнения операций формирования списков интерактивных компонентов, их изменения и удаления.*

Практическая значимость*. Представленная информация позволяет сделать обоснованный выбор фреймворка для создания web-приложений.*

Ключевые слова*: web-приложение; web-интерфейс; JavaScript; фреймворк; Angular; React; Vue.*

BASHOVYI V. M., STATSENKO V. V., STATSENKO D. V.
Kyiv National University of Technologies and Design, Ukraine

DETERMINING OF MODERN FRAMEWORKS SPEED FOR CREATING WEB-INTERFACES

Purpose*. Study of forming lists of interactive components speed by modern frameworks that are used to create web-interfaces.*

Methodology*. Methods for describing and analyzing the effectiveness of software tools, methods and tools for researching web applications are used.*

Results*. The most popular JavaScript frameworks that are used to create web interfaces have been identified. These include React, Angular and Vue frameworks. The time taken by the specified frameworks to perform operations on the list of interactive components is determined. It was found that the Angular framework showed the best results: the total time to perform the most complex operation (complete page generation) is 45% better than React and 44% better than Vue. It has been established that frameworks have different architectures, which can change the operating time in other applications. In particular, component removal rate testing showed that the Angular framework spends significantly less time in the Scripting phase than React and Vue, but much more in the rendering phase. This indicates a difference in the algorithms for preparing data for web-application elements.*

Scientific novelty*. The results of the JavaScript frameworks speed study for creating web applications are presented. The operations execution time of forming interactive components lists, their modification and deletion is analyzed.*

Practical significance*. The presented information allows making an informed choice of a framework for creating web-applications.*

Keywords*: web-application; web-interface; JavaScript; framework; Angular; React; Vue.*