

ОПТИМІЗАЦІЯ UI-БІБЛІОТЕК У ВЕБЗАСТОСУНКАХ: МЕТОДИ МІНІМІЗАЦІЇ НАДМІРНОГО КОДУ ДЛЯ ПІДВИЩЕННЯ ПРОДУКТИВНОСТІ ВЕБЗАСТОСУНКІВ

Недьошев М.В., Кириченко В. В.

У сучасній веброзробці оптимізація UI-бібліотек стала критичним аспектом створення швидких та ефективних вебзастосунків. Попри всі переваги, які надають бібліотеки інтерфейсних компонентів, їх використання часто призводить до збільшення розміру кінцевого продукту через надмірний код. Це дослідження аналізує різні методи оптимізації UI-бібліотек, включаючи мінімізацію коду, компонентний підхід, асинхронне завантаження та пропонує нову технологію збірки з використанням статичного аналізатора, що допомагає суттєво підвищити продуктивність вебзастосунків при збереженні функціональності та якості користувацького інтерфейсу.

Сучасні бібліотеки інтерфейсних компонентів надають розробникам багатий набір інструментів для створення інтерактивних і привабливих користувацьких інтерфейсів. Однак використання цих бібліотек призводить до значного збільшення розміру вебзастосунків, що безпосередньо впливає на їх продуктивність.

Основна проблема полягає у тому, що більшість UI-бібліотек за замовчуванням підключаються повністю, навіть якщо розробник використовує лише невелику частину компонентів. Це призводить до завантаження та виконання надмірного коду, який ніколи не використовується у конкретному застосунку. Такий підхід негативно впливає на час завантаження сторінки, споживання пам'яті та загальну продуктивність застосунку.

Для розробки вебзастосунків використовується мова JavaScript, динамічна природа якого ускладнює автоматичне виявлення та видалення невикористаного коду. На відміну від статично типізованих мов, де компілятор може легко визначити невикористані функції, у JavaScript набагато складніше визначити, який код реально використовується, особливо у випадках з динамічними імпортами.

Об'єднання (bundling) і мінімізація (minification) є основними методами оптимізації, які значно покращують час завантаження вебзастосунків. Об'єднання дозволяє поєднати кілька файлів в один, що зменшує кількість HTTP-запитів до сервера. Використання цих методів пришвидшує завантаження сторінки на 50% [1].

Для оптимізації UI-бібліотек найпотужнішим сучасним методом оптимізації є Tree Shaking [2], який дозволяє видаляти невикористаний код під час збірки застосунку. Цей метод працює шляхом аналізу статичних імпортів та залежностей, визначаючи, який код реально використовується, а який можна безпечно видалити. Tree Shaking найефективніший при використанні модулів у стандарті JavaScript ES6 з статичними імпортами. Такий підхід дозволяє інструментам збірки (як Webpack, Rollup, Vite) ідентифікувати та включати до фінальної збірки лише ті компоненти, які використовуються в коді. Однак Tree Shaking має обмеження, пов'язані з динамічною природою JavaScript – Tree Shaking не ефективний при роботі з динамічними імпортами або коли код використовує глобальні об'єкти або рефлексію та потребує розділення коду на частини. Розділення коду на частини (Code Splitting) дозволяє розділити вихідний код на менші частини (chunks), які можуть завантажуватися за потребою. Це зменшує розмір вихідних файлів і прискорює час до інтерактивності застосунку. UI-бібліотеки розділяють компоненти на окремі частини, які потім можна проігнорувати під час збірки вебзастосунка.

Більшість UI-бібліотек покладаються у своїй природі на рефлексію [3], використовуючи зовнішні реквізити (props), які керують поведінкою та зовнішнім виглядом компонента. Хоча більшість зовнішніх реквізитів задаються у коді до збірки додатків і є статичними, методи оптимізації як Tree Shaking не можуть видаляти зайві

частини з компонентів. Самі компоненти в більшості випадків містять безліч налаштувань, що створює багато зайвого коду для завантаження і виконання у браузері, особливо що таких компонентів на сторінці використовуються десятки або сотні.

Пропонується додатковий метод оптимізації компонентів шляхом аналізу вхідних реквізитів і побудови коду компонента залежно від вхідних даних під час збірки вебзастосунка. Такі компоненти можна називати віртуальними. Принцип генерування компонента показано на Рис. 1. Плагін для інструментів збирання як Webpack, Rollup, Vite містить у собі такі ключові елементи: код віртуальних компонентів, аналізатор віртуальних компонентів, генератор (renderer) віртуальних компонентів, виконавець та оптимізатор коду, аналізатор вхідного коду вебзастосунка та транслятор вхідного коду.

Для програміста процес розробки залишається знайомим, схожим на процес розробки з використанням UI-бібліотеки. Вихідні файли мають маленький і простий вигляд, наче компоненти створили з нуля спеціально під потреби вебзастосунка.

	Без віртуальних компонентів	З віртуальними компонентами
Кількість елементів	1000	1000
Кількість унікальних компонентів	8	8
Кількість символів вхідного коду	5357	1682
Розмір JS файлу	30.0 kB	28.5 kb
Час завантаження (preset fast 4G)	665 ms	543 ms
FCP	271 ms	275 ms
Memory usage	38.9 mb	28 mb

Таблиця 1 – порівняння розробки форми авторизації з нуля та з використанням віртуальних компонентів.

Отже, використовуючи додаткову оптимізацію з використанням віртуальних компонентів, які створюються під час збірки зменшується час завантаження, кількість коду, який потрібно писати та споживання пам'яті.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Anderson R. Bundling and Minification [Електронний ресурс] / Anderson Rick. – 2022. – Режим доступу до ресурсу: <https://learn.microsoft.com/uk-ua/aspnet/mvc/overview/performance/bundling-and-minification>
2. Hendriks F. How To Make Tree Shakeable Libraries [Електронний ресурс] / François Hendriks. – 2021. – Режим доступу до ресурсу: <https://blog.theodo.com/2021/04/library-tree-shaking/>
3. Mráz M. Component-based UI Web Development / Marcel Mráz – 2019.

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І
ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**



**ТЕОРЕТИЧНІ ТА ПРИКЛАДНІ
АСПЕКТИ РОЗРОБКИ
КОМП'ЮТЕРНИХ СИСТЕМ**

**ЗБІРНИК НАУКОВИХ ПРАЦЬ ЗА МАТЕРІАЛАМИ
VII ВСЕУКРАЇНСЬКОЇ НАУКОВО-ПРАКТИЧНОЇ КОНФЕРЕНЦІЇ
СТУДЕНТІВ І АСПІРАНТІВ
*24 квітня 2025 року***

Київ 2025

УДК 004

Відповідальна за випуск: М.І. Лендел

Збірник наукових праць за матеріалами VII Всеукраїнської науково-практичної конференції студентів і аспірантів «ТЕОРЕТИЧНІ ТА ПРИКЛАДНІ АСПЕКТИ РОЗРОБКИ КОМП'ЮТЕРНИХ СИСТЕМ '2025», 24 квітня 2025 року, НУБіП України, Київ. – 452 с. (електронне видання)

Відповідальність за зміст публікацій несуть автори.

Передрук матеріалів, а також використання їх будь-якій формі допускається лише з дозволу авторів