

**MINISTERUL EDUCAȚIEI ȘI CERCETĂRII**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare Informatică și Microelectronică**

**Departamentul Ingineria, Software și Automatică**

**Admis la susținere**

**Șef departament: Fiodorov I. dr., conf.univ..**

\_\_\_\_\_

„\_” \_\_\_\_\_ 2022

**ANALIZA ȘI COMPARAREA PERFORMANȚEI  
MANIPULĂRII DOM CU DIFERITE FRAMEWORK-URI  
JAVASCRIPT**

**Proiect de master**

**Student:** \_\_\_\_\_ Timofeev Maxim  
**Conducător:** \_\_\_\_\_ conf. univ. Romanenko A.  
**Coordonator:** \_\_\_\_\_ asistent univ. Cojocaru S.

**Chișinău 2022**

## АННОТАЦИЯ

Веб-сайты 2020 года часто являются многофункциональными и интерактивными приложениями. JavaScript - популярный язык программирования для веб, с множеством фреймворков. Общим знаменателем для интерактивных веб-приложений является необходимость в эффективных методах манипулирования объектной моделью документа для обеспечения надежного пользовательского опыта

В этом исследовании сравниваются чистый JavaScript и JavaScript-фреймворки Angular, React и Vue.js в производительности DOM, методологии манипулирования DOM и размера приложения.

Что касается методологии манипулирования DOM, существует явное различие между чистым JavaScript и выбранными фреймворками. В чистом JavaScript DOM манипулирование осуществляется путем прямого взаимодействия с DOM API. При использовании выбранных фреймворков фактическое взаимодействие с интерфейсом DOM абстрагируется от разработчика и обрабатывается фреймворком. В то время как React и Vue.js реализовали Virtual DOM для оптимизации взаимодействия с DOM, Angular использует инкрементный DOM.

Чистый JavaScript имеет лучшую производительность во всех тестах и наименьший размер приложения. Среди фреймворков React имеет лучшую производительность, Angular показал близкие результаты с React почти во всех тестах, а Vue.js был немного медленнее. Vue.js был немного медленнее в большинстве тестов. Почти во всех тестах приложения работали лучше в Google Chrome.

## REZUMAT

Site-urile web 2020 sunt adesea aplicații interactive și bogate în funcții. JavaScript este un limbaj de programare popular pentru web, cu multe cadre de lucru. Numitorul comun al aplicațiilor web interactive este nevoia de metode eficiente de manipulare a modelului de obiect document pentru a oferi o experiență robustă utilizatorului.

Acest studiu compară JavaScript pur și cadrele JavaScript Angular, React și Vue.js în ceea ce privește performanța DOM, metodologia de manipulare a DOM și dimensiunea aplicației.

În ceea ce privește metodologia de manipulare a DOM, există o diferență clară între JavaScript pur și cadrele selectate. În JavaScript pur, manipularea DOM se face prin interacțiunea directă cu API-ul DOM. În cazul cadrelor selectate, interacțiunea efectivă cu interfața DOM este abstractizată de la dezvoltator și este gestionată de cadru. În timp ce React și Vue.js au implementat Virtual DOM pentru a optimiza interacțiunea DOM, Angular utilizează DOM incremental.

Pure JavaScript are cea mai bună performanță în toate testele și cea mai mică dimensiune a aplicației. Dintre framework-uri, React are cea mai bună performanță, Angular s-a comportat aproape de React în aproape toate testele, iar Vue.js a fost puțin mai lent. Vue.js a fost puțin mai lent în majoritatea testelor. Aplicațiile au funcționat mai bine în Google Chrome în aproape toate testele.

## **ABSTRACT**

Websites of 2020 are often feature rich and highly interactive applications. JavaScript is a popular programming language for the web, with many frameworks available. A common denominator for highly interactive web applications is the need for efficient methods of manipulating the Document Object Model to enable a solid user experience.

This study compares Vanilla JavaScript and the JavaScript frameworks Angular, React and Vue.js in regards to DOM performance, DOM manipulation methodology and application size.

In regards to DOM manipulation methodology, there is a distinct difference between Vanilla JavaScript and the selected frameworks. In Vanilla JavaScript DOM manipulation is handled by direct interaction with the DOM interface. When using the selected frameworks the actual interaction with the DOM interface is abstracted away from the developer and handled by the framework. While React and Vue.js both have implemented a Virtual DOM to optimize DOM interactions, Angular has implemented Incremental DOM. Vanilla JavaScript had the best DOM performance in all tests and the smallest application size. Amongst the frameworks React had the best DOM performance, Angular performed close to React in nearly all test, and Vue.js was slightly slower in.

## Содержание

ВВЕДЕНИЕ.....	10
1 АНАЛИЗ ОБЛАСТИ ИССЛЕДОВАНИЯ.....	11
1.1 Объектная модель документа - DOM.....	11
1.2 Объектная модель CSS.....	12
1.3 Рендеринг HTML документа в браузере.....	14
1.3.1 Описание первичного рендера.....	14
1.3.2 Отрисовка и перерасчет дерева и описание браузерных движков.....	16
1.4 Анализ JavaScript и фреймворков.....	16
1.4.1 Анализ языка JavaScript.....	17
1.4.2 Анализ фреймворка Angular.....	17
1.4.3 Анализ фреймворка React.....	17
1.4.4 Анализ фреймворка Vue.js.....	17
1.5 Постановка исследовательских вопросов.....	17
2 АНАЛИЗ МЕТОДИКИ ИССЛЕДОВАНИЯ.....	20
2.1 Схема проведения эксперимента.....	20
2.2 Тестовые примеры и их характеристика.....	21
2.3 Автоматизация тестов для приложения.....	23
2.4 Описание програм для исследования.....	23
3 ОПИСАНИЕ ПРОГРАМ ДЛЯ ИССЛЕДОВАНИЯ.....	24
3.1 Сравнение методологии манипулирования DOM.....	24
3.2 Сравнение производительности приложений.....	25
3.2.1 Начальный рендеринг страницы.....	25
3.2.2 Создание 10000 элементов в приложении.....	28
3.2.3 Обновление всех элементов приложения.....	30
3.2.3 Обновление каждого второго элемента приложения.....	33
3.2.3 Удаление всех элементов приложения.....	35

3.3 Сравнение размера приложения .....	38
4 Анализ проделанной работы.....	40
ЗАКЛЮЧЕНИЕ .....	43
БИБЛИОГРАФИЯ.....	44

## БИБЛИОГРАФИЯ

- [1] I. Allen. The brutal lifecycle of javascript frameworks. Disponibil: <https://stackoverflow.blog/2018/01/11/brutal-lifecycle-javascript-frameworks/>
- [2] angular.io. Angular cli. Disponibil: <https://cli.angular.io/>
- [3] chromium.org. Chromedriver - webdriver for chrome. Disponibil: <https://chromedriver.chromium.org/downloads>
- [4] S. Fluin. Version 9 of angular now available - project ivy has arrived. Disponibil: <https://blog.angular.io/version-9-of-angular-now-available-project-ivy-has-arrived-23c97b63cfa3>
- [5] Mozilla Foundation. Mozilla geckodriver. Disponibil: <https://github.com/mozilla/geckodriver/releases>
- [6] Mozilla Foundation. Mdn web docs. Disponibil: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference>
- [7] A. Goel. 10 best javascript frameworks to use in 2020. Disponibil: <https://hackr.io/blog/best-javascript-frameworks>
- [8] S. Greif. The 12 things you need to consider when evaluating any new javascript library. Disponibil: <https://www.freecodecamp.org/newsthe-12-things-you-need-to-consider-when-evaluating-any-new-javascript-library-3>
- [9] IEEE. Ieee standard glossary of software engineering terminology. IEEE Std 610.12-1990, pages 1–84, 1990.
- [10] ECMA International. Standard ecma-262. Disponibil: <https://www.ecma-international.org/publications/standards/Ecma-262.html>
- [11] S. Peyrott. React virtual dom vs incrementaldom vs ember's glimmer. Disponibil: <https://auth0.com/blog/face-off-virtual-dom-vs-incremental-dom-vs-glimmer>
- [12] React. React github page. Disponibil: <https://github.com/facebook/react>
- [13] selenium.dev. Selenium official website. Disponibil: <https://www.selenium.dev/>