МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет «Одеська юридична академія»

ЄВРОПЕЙСЬКІ ОРІЄНТИРИ РОЗВИТКУ УКРАЇНИ В УМОВАХ ВІЙНИ ТА ГЛОБАЛЬНИХ ВИКЛИКІВ XXI СТОЛІТТЯ: СИНЕРГІЯ НАУКОВИХ, ОСВІТНІХ ТА ТЕХНОЛОГІЧНИХ РІШЕНЬ

МАТЕРІАЛИ
Міжнародної науково-практичної конференції

19 травня 2023 року

У двох томах

Том 1

Видавництво
«Юридика»
2023

УДК 005.332.2(4):316.42(477)"364""20"(062.552)
Є24

Рекомендовано до друку Вченою радою
Національного університету «Одеська юридична академія»
(протокол № 3 від 16.06.2023 р.)

За загальною редакцією **С. В. Ківалова.**

Відповідальний за випуск **М. Р. Аракелян.**
Матеріали видано в авторській редакції.

Є24 **Європейські** орієнтири розвитку України в умовах війни та глобальних викликів XXI століття: синергія наукових, освітніх та технологічних рішень : у 2 т. : матеріали Міжнар. наук.-практ. конф. (м. Одеса, 19 травня 2023 р.) / за загальною редакцією С. В. Ківалова. – Одеса : Видавництво «Юридика», 2023. – Т. 1. – 790 с.

ISBN 978-617-8263-39-3

Матеріали Міжнародної науково-практичної конференції «Європейські орієнтири розвитку України в умовах війни та глобальних викликів XXI століття: синергія наукових, освітніх та технологічних рішень». У першому томі збірника містяться наукові напрацювання вчених, практиків, військовослужбовців у теоретичній та емпіричній площині в умовах повномасштабного військового вторгнення у сферах філософських основ, загальної теорії та історичних досліджень держави і права, актуальних проблем світових соціально-політичних процесів, соціології та психології. Висвітлено питання загроз національній безпеці в їхньому конституційному вимірі, у рамках міжнародного та європейського права, трудового права та права соціального забезпечення, земельного, аграрного та екологічного права, пов'язані з функціонуванням економіки та підприємництва в умовах європейського вибору України. Розглянуто проблеми інформатизації та цифровізації суспільства, захисту інформації та кібербезпеки в умовах військового вторгнення, питання методики викладання іноземних мов, теорії та практики перекладу, проблеми лінгвістики та журналістики.

Збірник розраховано на наукових та науково-педагогічних працівників, здобувачів вищої освіти, практичних працівників у сферах юридичної, економічної, соціологічної, політологічної, психологічної, філологічної наук, журналістики та кібербезпеки тощо.

УДК 005.332.2(4):316.42(477)"364""20"(062.552)

ISBN 978-617-8263-37-9 (у 2 т.)
ISBN 978-617-8263-39-3 (т. 1)

СЕКЦІЯ 12
ІНФОРМАТИЗАЦІЯ ТА ЦИФРОВІЗАЦІЯ СУСПІЛЬСТВА В ЕПОХУ СТРІМКОГО РОЗВИТКУ ТЕХНОЛОГІЙ ШТУЧНОГО ІНТЕЛЕКТУ

ДИКА АНАСТАСІЯ ІВАНІВНА

Національний університет «Одеська юридична академія», асистент кафедри інформаційних технологій

ЛОБОДА ЮЛІЯ ГЕННАДІЇВНА

Національний університет «Одеська юридична академія», доцент кафедри інформаційних технологій, кандидат педагогічних наук, доцент

РОЗВИТОК ЗАСТОСУВАННЯ WEBASSEMBLY У ВЕБРОЗРОБЦІ

За довгу історію Інтернету JavaScript був основною мовою програмування у веброзробці. Завдяки своєму простому синтаксису та підтримці у всіх популярних браузерах, мова набула популярності серед розробників. Однак у міру зростання потужності комп'ютерів і потреб користувачів прості сайти перетворилися на вебзастосунки, які є повноцінними аналогами десктопних програм. Зі зростанням можливостей зростали і вимоги до продуктивності таких програм. Рушії браузерів розробили способи оптимізації коду, який вони запускають, і інтенсивна конкуренція між браузерами сприяла якісному підвищенню продуктивності.

Незважаючи на всі способи збільшення швидкості виконання коду JavaScript, головним зупиняючим фактором була динамічна типізація мови. Завдяки динамічному набору тексту, рушій браузера повинен перевіряти кожен раз, коли програма виконується, чи змінна є цілим числом, числом з рухомою крапкою або будь-яким іншим допустимим типом. Отже, кожна інструкція JavaScript має пройти кілька перевірок типу та перетворення, що сповільнює виконання. Це призвело до ідеї використання мов із суворою типізацією, яка могла б компенсувати цей недолік. Але рушії браузера не можуть виконувати код інших мов, тому з'явилася технологія WebAssembly.

WebAssembly (або скорочено Wasm) – це нова технологія, що дозволяє виконувати вебзастосунки з високою швидкістю, наближеною до швидкості нативних застосунків. Ця технологія дозволяє писати код мовами зі статичною типізацією, а потім аналізувати його в більш нативний і машиночитаний формат, що прискорює виконання програм порівняно з JavaScript. Технологія є кросплатформною та підтримує основні мови програмування: C++, C, Java, C# і може компілювати їх до Wasm-байт-коду, що може бути виконаний в браузері.

WebAssembly підтримується W3C (World Wide Web Consortium), громадською міжнародною організацією, що займається відкритими стандартами. Прихильність W3C до WebAssembly дарує довіру та впевненість у технології. Але необхідно зазначити, що Wasm це не мова програмування, подібно до того як байт-код Java це не мова програмування, а результат компіляції і блок коду, що запускається [1].

WebAssembly офіційно став стандартом W3C у 2019 році. Проте більшість вебкористувачів і велика кількість веброзробників ніколи

про це не чули. Це й не дивно. Використання WebAssembly є прозорим для користувача, а у вебзастосунку доступ до коду Wasm здійснюється через стандартні API. Якби розробник не шукав його, він би не знав, що він там є. Ця прозорість є однією з чудових переваг WebAssembly. Код, написаний на WebAssembly, виконує свою роботу і не заважає іншим частинам стеку розробки програмного забезпечення. Основні проблеми, які можна вирішити за допомогою WebAssembly у браузері, – це швидкість, гнучкість та портативність.

Для використання WebAssembly в браузері потрібно додатково встановити Wasm-модуль та скрипти JavaScript, які будуть викликати функції з Wasm-модуля. WebAssembly зберігається у текстовому форматі, який легко читається (.wat), а потім передається браузеру у вигляді двійкового подання (.wasm). Wasm завжди завантажується і викликається тільки з JavaScript. Ба-більше, JavaScript і Wasm працюють в одній «пісочниці», і виконуються одним і тим самим рушієм (V8). Одночасно з Wasm можна викликати JavaScript. Це може бути виклик функції з передачею аргументів і поверненням значення, або це може бути просто виконання довільного рядка як JavaScript-коду. Для цього можна використовувати спеціальні бібліотеки та інструменти, такі як Emscripten та WebAssembly Studio [2].

Веббраузер завантажує JavaScript як звичайний текст. Нестиснутий файл JavaScript може бути в 100 разів більшим, ніж еквівалентний файл WebAssembly. Навіть зі стисненим JavaScript файл WebAssembly становить одну десяту розміру еквівалентного фрагмента коду JavaScript. Перевага полягає в тому, що менші програми WebAssembly завантажуються швидше, що скорочує час, необхідний для того, щоб вебсторінка стала інтерактивною. Скомпільований код WebAssembly працює набагато швидше, ніж динамічно завантажувані файли JavaScript. Такі прийоми підвищення продуктивності, як своєчасна (JIT) або випереджаюча (AOT) компіляція, можна використовувати для досягнення майже рідної продуктивності коду WebAssembly. Майже нативна продуктивність просто неможлива з JavaScript на баці браузера.

Іншою перевагою WebAssembly є те, що він дозволяє розробникам переносити складні операції з боку клієнта на бік сервера, що зменшує навантаження на клієнтський браузер та забезпечує швидшу роботу вебзастосунків. Крім того, WebAssembly може використовуватись для оптимізації роботи вебзастосунків, наприклад, для оброблення медіаданих, виконання наукових розрахунків та ігрових рушіїв.

З WebAssembly написання вебзастосунків стає більш гнучким. JavaScript досі був найкращим вибором для більшості потреб у веброзробці, але натомість з'являється можливість звертатися до спеціалізованої мови, коли дійсно потрібне прискорення. Такі частини як UI та логіка застосунку можуть бути на JavaScript з основною функціональністю на WebAssembly [3]. При оптимізації продуктивності в наявних JavaScript-застосунках, «вузькі місця» можна переписати мовою, що краще підходить для можливої проблеми. У веб стало можливим переносити не лише власні застосунки, а й неймовірну кількість бібліотек C++ та застосунків з відкритим сирцевим кодом, що існують зараз.

C++ підтримується практично на кожній платформі, разом з iOS та Android. WebAssembly дає змогу використовувати цю мову як спільну мову для веб та мобільних застосунків.

WebAssembly має численні переваги, які роблять його важливим інструментом у фронтенд розробці, але він також має деякі недоліки:

Обмежена інтеграція зі стандартними бібліотеками JavaScript. Хоча WebAssembly може взаємодіяти з JavaScript, з'єднання двох технологій може бути складним і вимагати додаткової роботи зі сторони розробника. WebAssembly та JavaScript використовуються для різних цілей, тому порівняння їх продуктивності не зовсім коректне. JavaScript використовується для розробки вебзастосунків та взаємодії з користувачем, тоді як WebAssembly використовується для виконання складних обчислень на стороні клієнта та сервера. Оскільки JavaScript та WebAssembly можуть використовуватися разом, розробники можуть використовувати кожну технологію там, де вона найбільш ефективна.

Хоча WebAssembly може бути виконаний в різних вебраузерах, WebAssembly не має прямого доступу до об'єктної моделі документа та подій, що потрібні для маніпулювання вебсторінками. Тому, для роботи з ним, потрібно використовувати мінімальну кількість коду на JavaScript. Проте, розробники вже створили рішення, що дозволяє використовувати WebAssembly в поєднанні з іншими середовищами, наприклад, фреймворк Microsoft Blazor використовує середовище.NET, що завантажується як скомпільований Wasm-файл. Blazor працює з JavaScript та надає розширені можливості для створення вебзастосунків [4]. Це лише один з прикладів використання WebAssembly, існують інші експерименти, такі як Pyodide [5], що дозволяє використовувати мову Python в браузері з доповненим списком математичних інструментів для роботи з даними. Але навіть якщо працювати тільки з JavaScript, все ще можна отримати перевагу від WebAssembly і пришвидшення, що їм забезпечується, через поліпшені бібліотеки та фреймворки. Досить скоро з'явиться можливість завантажити та імпортувати модулі як будь-які інші ECMAScript модулі, використовуючи <*script type='module'*> та просто викликаючи їх функції з JavaScript. Щодо фреймворків, Ember вже вивчає реалізацію WebAssembly для їх Glimmer VM, також існує потенціал для впровадження деяких функцій React у WebAssembly [3].

Отже, не можна точно сказати, що WebAssembly завжди швидший за JavaScript, оскільки вони використовуються для різних цілей. Однак, WebAssembly може бути корисним інструментом для виконання складних обчислень та операцій з великими обсягами даних, які можуть сповільнювати виконання JavaScript коду. WebAssembly має багато переваг, його використання потребує обережності та розуміння можливих ризиків. Одночасно, WebAssembly відкриває новий світ у сфері вебзастосунків. Стають доступними раніше неможливі типи програм і види взаємодії з користувачем. Спільнота з провідних компаній, такі як Google, Mozilla, Apple, Microsoft разом створюють та розвивають цю технологію. Все більшої популярності набувають нейронні мережі, доповнена реальність та інші цікаві технології,

які невдовзі ми зможемо використовувати прямо на клієнтській стороні – у нашому браузері.

Список використаних джерел:

1. Burkhart N., Liao W., & Guzide O. (2020). An Overview of WebAssembly. *Proceedings of the West Virginia Academy of Science, 92(1)*. https://doi.org/10.55632/pwvas.v92i1.682
2. Ben L. Titzer. 2022. A fast in-place interpreter for WebAssembly. *Proc. ACM Program. Lang. 6, OOPSLA2*, Article 148 (October 2022), 27 pages. https://doi.org/10.1145/3563311
3. WebAssembly: як та чому. https://codeguida.com/post/1517
4. ASP.NET Core Blazor: https://learn.microsoft.com/uk-ua/aspnet/core/blazor/?WT.mc_id=dotnet-35129-website&view=aspnetcore-7.0
5. What is Pyodide? https://pyodide.org/en/stable/index.html

Ключові слова: веброзробка, вебпрограмування, веббраузер, JavaScript, WebAssembly.

Key words: web development, web programming, web browser, JavaScript, WebAssembly.

ТРОФИМЕНКО ОЛЕНА ГРИГОРІВНА

Національний університет «Одеська юридична академія»,
доцент кафедри інформаційних технологій,
кандидат технічних наук, доцент

МАНАКОВ СЕРГІЙ ЮРІЙОВИЧ

Національний університет «Одеська юридична академія»,
доцент кафедри інформаційних технологій, кандидат технічних наук

ПРОБЛЕМИ БЕЗПЕКИ ВЕБЗАСТОСУНКІВ

Нині за умов розповсюдження кіберзлочинів та проведення цілеспрямованих інформаційних війн вебресурси повсюдно піддаються атакам різного масштабу і складності з метою впливу на конфіденційність, цілісність і доступність даних інформаційних систем. Поширеність порушень безпеки сайтів та вебзастосунків, а також важливість їх запобігання зробило тестування безпеки невіддільною складовою життєвого циклу розробки відповідного програмного забезпечення (ПЗ), що має виявляти вразливості, пов'язані із забезпеченням цілісного підходу до захисту програми від хакерських атак, вірусів, несанкціонованого доступу до конфіденційних даних.

Тестування веббезпеки перевіряє бізнес-логіку, проблеми автентифікації та авторизації, кодування вхідних і вихідних даних, а також інші можливі ризики, щоб виявити уразливі місця в безпеці і переконатися, що всі функції вебзастосунків безпечні. Для тестування безпеки важливо враховувати різницю між ризиком та ефективністю