

*Коваленко Святослав,
здобувач другого(магістерського) рівня вищої освіти
фізико-математичного факультету
Науковий керівник: Жуковський Сергій,
кандидат педагогічних наук,
доцент кафедри комп'ютерних наук та інформаційних технологій,
Житомирський державний університет імені Івана Франка,
м. Житомир, Україна*

WEB – ДОДАТОК З ВИКОРИСТАННЯМ REACT ТА АРХІТЕКТУРНОГО ПІДХОДУ FLUX

На даний момент одним з найпоширеніших напрямків розробки є розробка веб додатків та сервісів. Це обумовлено тим, що Веб додатки не вимагають встановлення, або інших додаткових передумов для використання, вони є кросплатформенними, тобто можуть використовуватися будь-якими пристроями в яких є підключення до інтернету.

В зв'язку з високим попитом в даній сфері з'являється все більше технологій та різних підходів, зараз вже важко знайти веб сторінку яка просто використовує стандартний стек технологій(HTML, CSS, JavaScript), хоча це досі можливо та використовується залежно від мети проекту.

В цій статті я хочу розповісти вам про архітектурний підхід Flux розроблений Facebook для різних веб-додатків[1]. Ця архітектура об'єднує в собі такі технології як:

- React
- Redux
- Container components
- Single-page application

В основному цей підхід використовується з React[2]. React – це технологія фронтенд розробки, яка працює на JavaScript, але вона відповідає лише за вигляд(View). Основною перевагою його використання є те, що працюючи над проектом за допомогою цієї технології ви можете просто вказувати як би ви хотіли бачити той чи інший компонент, оскільки він заснований на компонентному підході. Завдяки цьому можна створювати інкапсульовані компоненти, які керують власним станом, а з них вже будувати складні інтерфейси.

Flux – архітектурний підхід або набір шаблонів програмування для побудови призначеного для користувача інтерфейсу веб-додатків, що поєднується з реактивним програмуванням і побудований на односпрямованих потоках даних.

Основною відмінною рисою Flux є одностороння спрямованість передачі даних між компонентами Flux-архітектури. Архітектура накладає обмеження на потік даних, зокрема, виключаючи можливість поновлення стану компонентів самими собою. Такий підхід робить потік даних передбачуваним і дозволяє легше простежити причини можливих помилок в програмному забезпеченні.

У мінімальному варіанті Flux-архітектура може містити три шари, які послідовно взаємодіють:

- Actions (дії)
- Stores (сховища)
- Views (відображення)

Хоча зазвичай між діями і сховищами додають Dispatcher (диспетчер). Для реалізації даного підходу зазвичай використовується React бібліотека Redux, яка дозволяє легко реалізувати взаємодію між шарами Flux підходу.

В першу чергу Flux працює з інформаційною архітектурою, яка потім відбивається в архітектурі програмного забезпечення, тому рівень відображень слабо взаємодіє з іншими рівнями системи.

В купі з Flux підходом дуже зручно використовувати такий патерн проектування як **Container components** [3]. Суть даного патерна полягає в тому, що фронтенд додаток складається з двох основних частин Container та component.

Container – може містити презентаційні та контейнерні компоненти, але за звичай не містить в собі власної DOM розмітки, за винятком обгорткових тегів. Відповідає за отримання та обробку даних, та надає ці дані до компонентів. Може викликати Flux події та надавати їх компонентам.

Component – може містити в собі презентаційні та контейнерні компоненти, та зазвичай вміщає DOM розмітку та власні стилі. Не містить залежностей з подіями та сховищами, при потребі може отримати їх виключно з контейнеру.

Також описані вище технології можуть використовувати принцип **single-page application** [4] (односторінковий застосунок). **Single-page application** це веб-застосунок чи веб-сайт, який вміщується на одній сторінці з метою забезпечити користувачу досвід близький до користування настільною програмою. В односторінковому застосунку весь необхідний код - HTML, JavaScript, та CSS - завантажується разом зі сторінкою, або динамічно довантажується за потребою, зазвичай у відповідь на дії користувача. Сторінка не оновлюється і не перенаправляє користувача до іншої сторінки у процесі роботи з нею. Взаємодія з односторінковим застосунком часто включає в себе динамічний зв'язок з веб-сервером.

В результаті використання описаних вище технологій, розробник отримає продукт в якому легко та зручно працювати з даними. Завдяки Flux підходу, в проекті будуть чітко визначені компоненти(Stores) які відповідають за збереження всієї інформації що використовується додатком, програма зможе отримати доступ до будь яких даних в будь-якій компоненті, тобто місце де зберігається інформація завжди чітко визначене. Також обробка даних та їх оновлення або збереження також мають чітко визначене місце(Actions), а кожен виклик подій контролюється та відстежується. При виникненні багів або якихось проблем розробник може чітко визначити коли, після якої події та в якому місці сховища ця проблема виникає завдяки спеціальному інструменту відладки Redux DevTools також завдяки цьому інструменту користувач може переглянути стан додатку в будь який момент часу та переглянути сховище в повному об'ємі.

Список використаних джерел та літератури

1. Використання React JS. URL: <https://uk.education-wiki.com/6785493-uses-of-react-js>.
2. React – JavaScript-бібліотека для створення користувацьких інтерфейсів. 2021. URL: <https://uk.reactjs.org/>.
3. Container Components in React. 2021 – URL: <https://www.section.io/engineering-education/container-components-in-react/>.
4. Односторінковий застосунок. 2020 URL: https://uk.wikipedia.org/wiki/%D0%9E%D0%B4%D0%BD%D0%BE%D1%81%D1%82%D0%BE%D1%80%D1%96%D0%BD%D0%BA%D0%BE%D0%B2%D0%B8%D0%B9_%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA.

Ковальчук Анастасія,

*здобувач першого(бакалаврського) рівня вищої освіти
фізико-математичного факультету*

Федорчук Анна,

*кандидат педагогічних наук,
доцент кафедри комп'ютерних наук та інформаційних технологій,
Житомирський державний університет імені Івана Франка,
м. Житомир, Україна*

ПЛАТФОРМИ ДЛЯ РОЗРОБКИ КОМП'ЮТЕРНИХ ІГОР

Гра відіграє велику роль у процесі розвитку мислення та формуванні особистості. В початковому процесі досить часто використовують ігровий процес як метод легшої форми подання та засвоєння інформації школярами.

Комп'ютерні ігри посідають важливу роль у житті людини. Дітям вони замінюють спілкування з однолітками, а більшості дорослим цікавіші ніж телебачення і книги. Проте в ігор є безліч позитивних властивостей. Вона дає гравцеві можливість активно діяти в сконструйованому світі, випробувати альтернативну, дещо спрощену модель чогось. Бути будь ким і будь де, не встаючи з крісла, незважаючи на свої фізичні властивості, можна побудувати власний дім, взаємодіяти з різними тваринами – це дає можливість розвивати образне мислення [4].

Зважаючи на досить високу актуальність комп'ютерних ігор у наш час постає необхідність проаналізувати найпопулярніші платформи для розробки ігор, а саме: CodeCombat, Construct 2, Game Maker, Unity і GDevelop, щоб визначити найзручніші з них.

Першою комп'ютерною грою була “Space War” створена в 1962 році С. Расселом. Проте через свою високу вартість вона не змогла стати масовим захопленням. Гру, яка була доступною для всіх розробив в США 1972 року Н. Бушель і називалася вона “Понг” – це була електронна версія пінг-понгу. Далі було безліч комп'ютерних ігор, які за своєю структурою не відрізнялися від “Понга” проте мали інше оформлення [4].