УДК 004.514

Утьонишев В.А., магістрант, Попівщий В.І,доцент, науковий керівник

ДОСЛІДЖЕННЯ ТА АНАЛІЗ ОСОБЛИВОСТЕЙ РЕАЛІЗАЦІЇ КОМПОНЕНТНОГО ПІДХОДУ В СУЧАСНИХ JAVASCRIPT-ФРЕЙМВОРКАХ ТА БІБЛІОТЕКАХ

Запорізька державна інженерна академія, кафедра ПЗАС

В наш час бізнесу потрібні великі та складні веб-додатки з багатою функціональністю. Такі рішення потребують специфічних веб-технологій. Найчастіше, щоб зменшити складність продукту, його поділяють на маленькі прості частини – компоненти, які надають ізоляцію одних підсистем від інших. Такий компонентний підхід зменшує складність, покращує структуру системи, збільшує ефективність командної роботи.

Компонентний підхід в веб-розробці розвивається в двох напрямках: у JS-фреймворках та на більш низькому рівні – в HTML5 WebComponents.

Об'єкт дослідження даної роботи – JS-фреймворки та JS-бібліотеки, в яких втілений компонентний підхід.

Предмет дослідження – способи реалізації компонентів в різних фреймворках.

Мета роботи – виявлення найбільш перспективних підходів в розробці компонентів, створення та дослідження компонентів повторного використання на прикладі тестового веб-додатку.

В роботі наведені вимоги до тестового веб-додатку. Це сайт навчального закладу, що має систему авторизації, базу даних, можливість перегляду, додавання, редагування інформації. Для введення даних та інших операцій треба запропонувати відповідні компоненти.

Далі в роботі наведено огляд розвитку компонентного підходу в веб-розробці. Спочатку складність веб-додатків в основному регулювалася з боку сервера за рахунок розділення програми на окремі сторінки. З впровадженням AJAX і пов'язаних технологій розробники змогли відмовитися від потреби робити «переходи» між різними сторінками веб-додатку. З'являється технологія створення односторінкових додатків (Single-Page Applications, SPA). Логіка клієнтських SPA-додатків істотно ускладнюється, іноді вона навіть стає складнішою, ніж на серверній стороні. Можливим вирішенням даної складності може бути подальший поділ на компоненти і ізоляція логіки всередині однієї сторінки або документа.

Весь JavaScript-код, який включений на сторінку, має доступ до одного і того ж глобального об'єкту. Як і інші мови програмування, JavaScript має області видимості, які надають певний рівень «приватності» для коду функції. Ці лексичні області видимості використовуються для ізоляції змінних і функцій від решти глобального оточення.

У багатьох фреймворках та бібліотеках JavaScript починають використовувати MVC модель (Model-View-Controller), яка допомагає реалізації компонентного підходу. Основна ціль застосування цієї концепції полягає в відділенні бізнес-логіки (моделі) від її візуалізації (уявлення, виду). За рахунок такого поділу підвищується можливість повторного використання.

Далі в роботі розглянуто існуючі JavaScript фреймворки та бібліотеки, які використовують компонентний підхід.

AngularJS – це JavaScript-фреймворк з відкритим програмним кодом, призначений для розробки односторінкових застосунків. Його мета — розширення браузерних застосунків на основі шаблону MVC. Фреймворк працює зі сторінкою HTML, що містить додаткові атрибути (директиви) і пов'язує області введення або виведення сторінки з моделлю, яка являє собою звичайні змінні JavaScript. Значення цих змінних задаються вручну або отримуються зі статичних або динамічних JSON-даних. У 1.5.x версії додали компоненти, які майже повністю замінюють директиви.

Angular 2.0 значно відрізняється від попередньої версії, тут переосмислено практично все. Багато з відмінностей випливають з того, що він використовує ES6 з анотаціями та типами та використовує мову TypeScript. Додаток Angular 2 тепер складається з компонентів і являє собою їх дерево. Ідея схожа на Web Components, навіть розмітка Angular 2 компонентів поміщається в Shadow DOM. Самі компоненти та сервіси являють собою ES6 класи з анотаціями.

React.js — відкрита JavaScript бібліотека для створення інтерфейсів користувача, яка покликана вирішувати проблеми часткового оновлення вмісту веб-сторінки, з якими стикаються при розробці односторінкових застосунків. Особливості React.js: одностороння передача даних (компонент не може напряму змінювати властивості, що йому передані, але може їх змінювати через callback функції), віртуальний DOM, мова JSX.

Vue – це прогресивний фреймворк для створення користувацьких інтерфейсів. На відміну від фреймворків-монолітів, Vue створений придатним для поступового впровадження. Його ядро в першу чергу вирішує завдання рівня уявлення (view), що спрощує інтеграцію з іншими бібліотеками та існуючими проектами. Іншою важливою концепцією Vue є компоненти, і це одна з найпотужніших можливостей Vue. Компоненти розширюють базові HTML-елементи, дозволяючи інкапсулювати повторно використовуваний код.

Далі в роботі наведено реалізацію компонентного підходу при розробці тестового веб-додатку. Порівнюється ефективність компонентів Angular 2, Vue та React.

Отже, в роботі розглянуті переваги компонентного підходу у веб-застосунках, досліджені сучасні JavaScript фреймворки та бібліотеки які дозволяють використовувати компонентний підхід при розробці, та їх переваги. Слід зауважити, що Angular 2.0 якнайкраще підходе для розробки, тому що вся реалізація пишеться у компонентах, які мають слабкий зв'язок з іншими, та можуть повторно використовуватись.

Висновки:

- проведено дослідження розвитку компонентного підходу в веб-розробці та його впровадження в сучасних JavaScript фреймворках та бібліотеках;
- створено та досліджено групу компонентів повторного використання для тестового веб-додатка.

Література

1. Компонентний підхід[Електронний ресурс]. – режим доступу https://en.wikipedia.org/wiki/Component-based_software_engineering
2. Стефанов Стоян React.js. Быстрый старт. — СПб.: Питер, 2017. — 304 с.
3. Matthew Scarpino Building Web Components with TypeScript and Angular 4. – Quiller Technologies LLC, 2017
4. Nir Kaufman, Thierry Templier Angular 2 Components. – Packt Publishing, 2016. – 168 p.
5. Adam Freeman ProAngular. SecondEdition. – Apress, 2017. – 788 p.
6. OlgaFilipovaLearningVue.js 2. – Packt Publishing, 2016. – 382 p.
7. Загальні відомості про Vue.js [Електронний ресурс]. – режим доступу https://ru.vuejs.org/v2/guide/index.html