

РОЗРОБКА SPA ЗАСТОСУНКІВ ЗА ДОПОМОГОЮ JAVASCRIPT ФРЕЙМВОРКУ VUE.JS

По перше, розглянемо різницю між поняттям «фреймворк» та «бібліотека» для того щоб зрозуміти чому Vue.js вважається фреймворком. Отже, фреймворк – це програмне середовище спеціального призначення, своєрідний каркас, який використовується для того, щоб істотно полегшити процес об'єднання певних компонентів при створенні програм. Це основа, яка дозволяє додавати компоненти в залежності від потреб. База, на якій можна сформувати програму будь-якого призначення досить швидко і без особливих труднощів. Бібліотека – це збірник класів або функцій на мові JavaScript. Тож 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. З іншого боку, Vue повністю підходить і для створення складних односторінкових застосунків (SPA, Single-Page Applications), якщо використовувати його спільно з сучасними інструментами та додатковими бібліотеками.

Творцем Vue.js є Еван Ю, колишній співробітник Google. Почав він розробляти фреймворк в 2013 році, а в лютому 2014 року відбувся перший публічний реліз. Vue широко використовується серед китайських компаній, наприклад: Alibaba, Xiaomi. Він входить в ядро Laravel і PageKit. Нещодавно вільна система управління репозиторіями GitLab теж перейшла на Vue.js. В кінці вересня 2016 вийшов в реліз Vue.js 2.0, з більш кращими оновленнями і з упором на продуктивність – тепер використовується віртуальний DOM, підтримується серверний рендеринг, можливість використовувати JSX та ін. Хоча зараз він підтримується тільки співтовариством, він тримається гідно навіть на рівні продуктів таких гігантів, як Google і Facebook (Angular2 і React 15), і поступово наздоганяє їх по популярності.

SPA (single page application) – це тип веб застосунків, в яких завантаження необхідного коду відбувається на одну сторінку, що дозволяє заощадити час на повторне завантаження одних і тих же елементів.

Зрозуміти, наскільки зручними і корисними для користувачів бувають односторінкові додатки або SPA, можна на прикладі кількох популярних сервісів гіганта Google: Gmail та Google Translate. Користувачі постійно

використовують дані сервіси, і навряд чи у кого-то виникає бажання перейти на десктопні аналоги.

Робота SPA додатку полягає у з'єднанні клієнтської частини з REST API. Клієнт посилає запит на сервер, сервер повертає дані у форматі JSON, клієнт за допомогою певних скриптів виводить дані у потрібному для додатка вигляді.

Для побудови frontend частини SPA застосунку на Vue.js необхідні такі інструменти: пакетний менеджер NPM, за допомогою якого можна встановлювати всі необхідні для розробки пакети, Vue-cli – рекомендований офіційний шаблон проєктів на Vue.js з налаштуваннями Webpack. За допомогою команди `npm install -g vue-cli` встановлюємо Vue-cli, після цього виконуємо команду `vue init vue-spa`, таким чином в нас з'явиться готовий шаблон проєкту який нам презентував vue-cli. Для розмежування бізнес-логіки програми та користувацького інтерфейсу нам знадобиться інструмент Vuex (`npm i --save-dev vuex`). Це дуже зручно, коли логіка (код) програми відділена від html-структури. Для зв'язку з REST API знадобиться бібліотка AXIOS (`npm i --save-dev axios`). У ролі REST API щоб не витратити час на його розробку можна використати такий сторонній сервіс (псевдо REST API) Firebase від Google, або jsonplaceholder.typicode.com.

Структура проєкту. Що ж містить шаблон проєкту vue-cli? По перше це всі необхідні залежності проєкту, (файл `package.json`), файл з налаштуваннями Webpack, файл `index.html` містить просту розмітку HTML з єдиним елементом "app" в body. Він буде замінений на DOM, згенерований vue. З цієї причини тег body не рекомендується використовувати в якості кореневого елемента, також файл `gitignore` та `readme` – для роботи з віддаленим git-репозиторієм. Також в структурі можна побачити папку `src`, в якій зберігається точка входу проєкту (файл `main.js`), папку `components` (збереження окремих компонентів vue), папку `router` (тут зберігаються маршрути окремих компонентів), файл `App.vue` в якій зберігається основна частина додатку, та папка `assets` (для зображень або файлів стилю), особисто для роботи з бізнес-логікою проєкту створено папку `store`, де зберігаються файли які містять всі необхідні об'єкти Vuex (`store`, `actions`, `mutations`, `getters`).

Vue-component. Кожен файл `*.vue` складається з блоків трьох типів: `<template>`, `<script>` і опціонально `<style>`.

В результаті, можна розділити проєкт на пов'язані компоненти. Усередині компонента його шаблон, логіка і стилі невід'ємно пов'язані, і їх поєднання фактично робить компонент більш цілісним і легко підтримуваним.