МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ

КАФЕДРА ПРОГРАМНОЇ ІНЖЕНЕРІЇ  
  
  
  
  
  
  
  
  
  
Дисципліна «Сучасні каркаси web-додатків»

Лабораторна робота № 4

Виконала: студентка гр. 8.1213  
 Абдуллаєва Севда   
 Самандар Кизи

Перевірив: Лебідь Максим   
 Володимирович

Запоріжжя  
2024

**Тема: Створення плагінів**

**Завдання**

1. Проаналізувати статтю.

2. Створити провайдер для сервісів створених у попередній лабораторній роботі.

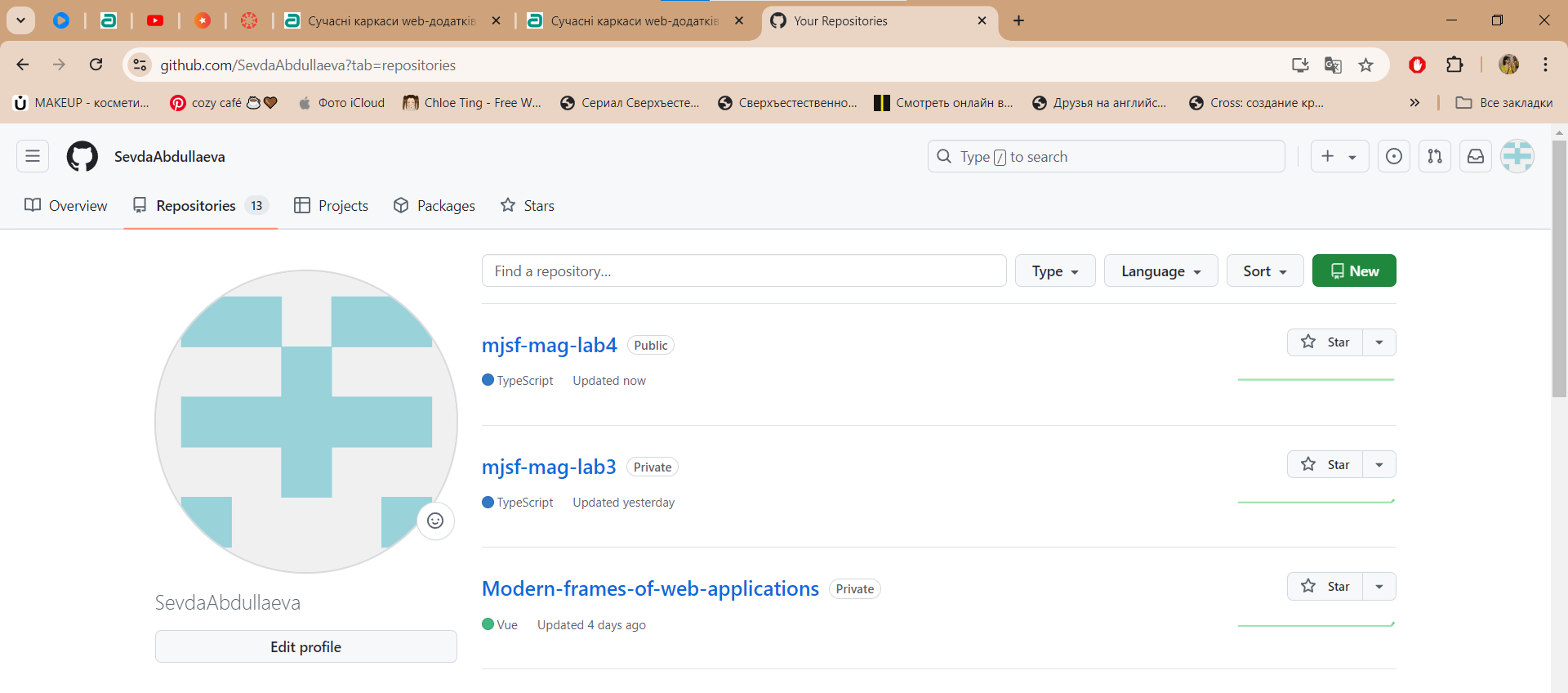
3. Створити плагін для реестрації провайдеру.

4. Інтегрувати плагін та сервіс в компоненти створені в попередніх роботах.

Всі завдання вище були виконані та код лабораторної роботи № 3 був модифікований, а саме:

* створено провайдер для сервісі створених у попередній лабораторній роботі;
* створено плагіни для реєстрації провайдеру;
* інтегровано плагіни та сервіси в компоненти, які були створені в попередніх роботах.

5. Зберегти зміни в новому репозиторію з назвою mjsf-mag-lab4.



6. Надати посилання на репозиторій у якості звіту.

<https://github.com/SevdaAbdullaeva/mjsf-mag-lab4.git>

**Контрольні питання**

**1. Що таке плагіни в контексті Vue.js 3 і як вони розширюють функціональність Vue?**

Плагіни в контексті Vue.js 3 — це спеціальні модулі, які дозволяють розширювати функціональність додатку на глобальному рівні. Вони надають можливість додати певні функції, сервіси або компоненти, які будуть доступні у всьому додатку. Це зручно для реалізації функціоналу, що має бути доступним у різних частинах програми, наприклад, глобальних методів, директив або сервісів для роботи з API, авторизації, управління сесіями тощо.

Плагіни можуть бути визначені як об'єкти з методом install або як звичайні функції. Метод install приймає екземпляр додатку Vue і параметри, передані під час підключення плагіна через app.use(). Цей метод дозволяє додавати глобальні методи або властивості через app.config.globalProperties, реєструвати компоненти або директиви, а також надавати ресурси для ін’єкції через provide/inject.

Плагіни розширюють можливості Vue-додатка за рахунок глобального доступу до функціональності, яка зазвичай використовується в багатьох місцях додатку. Наприклад, за допомогою плагінів можна:

Додавати глобальні методи або властивості, які доступні в усіх компонентах (наприклад, методи перекладу або обробки даних).

Реєструвати глобальні компоненти або директиви, що робить їх доступними у будь-якому компоненті без необхідності імпорту.

Ін’єктувати сервіси, такі як API-запити, авторизація або робота з локальним сховищем, які потім можна використовувати через глобальні властивості додатку.

Плагіни дозволяють зручно інтегрувати сторонні бібліотеки або налаштування у Vue-додаток. Вони допомагають централізувати функціональність і спрощують повторне використання коду в різних компонентах або навіть у різних проєктах.

Таким чином, плагіни в Vue.js 3 є потужним інструментом для розширення можливостей додатку, дозволяючи робити код більш організованим, структурованим і легким для підтримки та масштабування.

**2. Як встановити плагін?**

Щоб встановити плагін у Vue.js 3, необхідно скористатися методом app.use(), який підключає плагін до вашого додатку. Плагін можна визначити як об'єкт із методом install або як функцію. Після створення або імпорту плагіна, його підключають у головному файлі додатку (зазвичай це main.js).

* Спочатку створюється або імпортується плагін. Наприклад, якщо плагін визначений як об'єкт із методом install, він може виглядати так:

const myPlugin = {

install(app, options) {

// Додаємо глобальні методи або властивості

app.config.globalProperties.$myMethod = function() {

console.log('Метод з плагіна!');

};

}

};

Якщо плагін уже існує в окремому файлі, ви можете його імпортувати:

import myPlugin from './plugins/myPlugin';

* Після цього плагін підключається до додатку через метод app.use() у файлі main.js:

import { createApp } from 'vue';

import App from './App.vue';

import myPlugin from './plugins/myPlugin'; // Імпорт плагіна

const app = createApp(App);

app.use(myPlugin); // Підключення плагіна

app.mount('#app');

* Опціонально можна передати параметри до плагіна під час його встановлення:

app.use(myPlugin, { option1: true, option2: 'значення' });

Після підключення плагіна його функціональність стає доступною у всіх компонентах програми, наприклад, глобальні методи можна викликати через this.$myMethod(). Таким чином, встановлення плагіна у Vue.js 3 включає його імпорт або створення та реєстрацію через метод app.use(), що робить його доступним на рівні всього додатку.

**3. Які два способи визначення плагіна підтримуються в Vue.js 3?**

У Vue.js 3 підтримуються два способи визначення плагіна: як об'єкт із методом install і як функція. Обидва ці підходи дозволяють створити плагін, який можна підключити до Vue-додатку за допомогою методу app.use().

Плагін як об'єкт із методом install: У цьому випадку плагін визначається як об'єкт, який повинен містити метод install. Цей метод викликається Vue при підключенні плагіна і приймає два аргументи: екземпляр додатку Vue (app) та опціональні параметри плагіна. У межах цього методу можна додавати глобальні методи, компоненти, директиви або інші елементи для додатку.

Приклад плагіна як об'єкта:

const myPlugin = {

install(app, options) {

// Додаємо глобальні методи або властивості

app.config.globalProperties.$myMethod = function() {

console.log('Це метод із мого плагіна!');

};

}

};

Такий підхід дозволяє добре організувати логіку плагіна та додавати декілька різних функціональних елементів у додаток.

* Плагін як функція: Другий спосіб — це визначення плагіна як простої функції. Ця функція виконує роль методу install і отримує ті ж аргументи: екземпляр додатку Vue і параметри, передані при підключенні плагіна. У цьому випадку функція безпосередньо додає необхідну функціональність.

Приклад плагіна як функції:

function myPlugin(app, options) {

// Додаємо глобальний метод

app.config.globalProperties.$myMethod = function() {

console.log('Це метод із плагіна, визначеного як функція!');

};

}

Використання функції як плагіна підходить для простіших плагінів, де не потрібно створювати складні об'єкти.

Таким чином, Vue.js 3 дозволяє визначати плагіни двома способами: через об'єкт із методом install або через функцію. Обидва підходи мають однакову функціональність і можуть використовуватися в залежності від складності плагіна та його призначення.

**4. Які можливі сценарії використання плагінів Vue?**

Плагіни Vue.js мають широкий спектр використання і дозволяють розширювати функціональність додатка на глобальному рівні. Вони корисні в багатьох сценаріях, де необхідна загальна для всього додатка функціональність або інтеграція сторонніх бібліотек. Ось основні сценарії використання плагінів у Vue.js:

* Глобальна реєстрація компонентів. Плагіни можуть реєструвати глобальні компоненти, які можна використовувати в будь-якій частині додатка без необхідності імпорту. Це корисно для загальних компонентів, таких як кнопки, модальні вікна або форми.

app.component('MyGlobalComponent', MyComponent);

* Реєстрація глобальних директив. Плагіни можуть додавати користувацькі директиви, доступні для використання в усьому додатку. Це корисно для створення власних поведінкових патернів, таких як керування фокусом або валідація введених даних.

app.directive('focus', {

mounted(el) {

el.focus();

}

});

* Ін'єкція сервісів або залежностей. Плагіни можуть надавати глобальні сервіси, такі як робота з API, управління сесією користувача або авторизація. Ці сервіси можна ін'єктувати у всі компоненти додатка через provide/inject або додавати до глобальних властивостей Vue.

app.config.globalProperties.$auth = authService;

* Додавання глобальних методів або властивостей. Плагіни дозволяють додати функції або об'єкти до всіх компонентів через глобальні властивості. Наприклад, можна додати метод для перекладу тексту, логування або взаємодії з браузерним сховищем.

app.config.globalProperties.$translate = function(key) {

// Логіка для перекладу

};

* Підключення сторонніх бібліотек. Плагіни часто використовуються для інтеграції сторонніх бібліотек з Vue-додатком. Це може бути інтеграція з бібліотеками для роботи з графікою, анімаціями, графіками або іншими інструментами, які не є частиною Vue.

app.use(SomeExternalLibrary, options);

* Модульність та повторне використання. Плагіни дозволяють структурувати код таким чином, що певні частини логіки стають модульними та легко повторно використовуваними. Це допомагає створювати кастомні функціональні модулі, які можна використовувати в кількох проєктах.
* Реалізація глобальної логіки або поведінки. За допомогою плагінів можна реалізовувати глобальні патерни поведінки, наприклад, обробку помилок, відстеження подій або логування. Це допомагає централізувати важливі аспекти додатка і застосовувати їх у всіх компонентах.
* Забезпечення інтеграції з роутером та Vuex. Vue Router і Vuex — це плагіни, які додають маршрутизацію та централізоване управління станом додатка. Їх можна підключати та налаштовувати через плагіни, що дозволяє керувати складними аспектами додатка.
* Таким чином, плагіни в Vue.js мають багато різних сценаріїв використання. Вони дозволяють робити додаток більш модульним, зручним для масштабування та організації коду, а також спрощують повторне використання загальних функцій і логіки в різних частинах додатка.

**5. Як передати параметри конфігурації в плагін під час встановлення?**

Щоб передати параметри конфігурації в плагін під час його встановлення, зазвичай потрібно використовувати опції, передані до функції ініціалізації плагіна. Наприклад, у Vue.js це можна зробити через метод Vue.use(), де ви передаєте плагін разом з параметрами в якості другого аргументу. Це може виглядати приблизно так: Vue.use(MyPlugin, { option1: value1, option2: value2 }). У плагіні ви можете отримати ці параметри через options, що передається в функцію плагіна. Таким чином, ви зможете налаштувати поведінку плагіна відповідно до ваших потреб під час його встановлення.

**6. Як можна використовувати inject для надання функції або атрибута користувачам плагіна?**

Щоб використовувати inject для надання функції або атрибута користувачам плагіна у Vue.js, спочатку потрібно визначити, що саме ви хочете надати. Ви можете створити плагін, в якому за допомогою provide передаєте значення або функції в контексті вашого компоненту. Потім, у дочірніх компонентах або інших частинах програми, які використовують ваш плагін, ви можете скористатися inject, щоб отримати ці значення. Це дозволяє легко ділитися функціональністю або даними між компонентами без необхідності передавати їх через props. Наприклад, у вашому плагіні ви могли б написати Vue.prototype.$myFunction = function() { /\* ваш код \*/ }, а потім у дочірньому компоненті використовувати inject для доступу до цієї функції, що дозволить користувачам вашого плагіна зручно її використовувати.