Курс "Технології створення web-застосунків"

Лабораторна робота №3 "Технології JS. Основи Vue"

Завдання. За допомогою бібліотеки Vue розробити сайт "Згенеруй свою команду". Команда генерується з трьох випадково вибраних персонажів семи серій "Зоряних війн" (swapi.co), які підключаються через відповідний API. Кожен персонаж є окремим компонентом. При кліку на компоненті повинен генеруватися новий випадковий персонаж.

Потрібні знання:

- HTML, CSS
- JavaScript
- Синтаксис ES6 (ECMAScript 6)
- Технології Node.js (пакетний менеджер npm)
- Основи бібліотеки Vue

Хід роботи

1. Інсталюйте Node.js (https://nodejs.org/)
2. Інсталюйте редактор Visual Studio Code (https://code.visualstudio.com/) або Sublime Text.
4. Створіть каталог проектів Vue (наприклад, D:/VueProjects).
5. Запустіть термінал (Node.js command prompt) і перейдіть в каталог проектів Vue.
1. Інсталюйте інструмент командного рядка Vue CLI:

   npm install -g vue-cli
4. Створіть проект StarBase:

   vue-init webpack-simple starbase

   (Буде задано декілька запитань: Project name, Project description, Author, Use sass?(y/N) – натискати Enter)

7. Перейдіть в каталог starbase (cd starbase) і подайте команду npm install (до вашого проекту будуть додані всі залежності)
8. Запустіть сервер розробки: npm run dev В браузері з'явиться стандартний екран проектів Vue
9. Відкрийте в редакторі каталог starbase та вивчіть його структуру. Відкрийте файл компонента src/App.vue. Тут ви побачите шаблон, що відображається в браузері.
10. Впевніться, що файл src/main.js має наступний вміст:

```
import Vue from 'vue'
import App from './App.vue'

new Vue({
  el: '#app',
  render: h => h(App)
})
```

11. Відкоригуйте файл src/App.vue:

```
<template>
    <div id="app">
        <h3>{{title}}</h3>
    </div>
</template>
<script>
export default {
    name: 'app',
        data() {
            return {
                title: 'Створи Свою Команду'
            }
        }
}
</script>
```

Перейдіть в браузер щоб побачити результат

12. Додайте до вашого застосунку трохи CSS стилів. Спочатку підключіть Bootstrap. Відкрийте файл index.html і після тегу </title> вставте

```
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" crossorigin="anonymous">
```

Перевірте в браузері результат. Далі додайте до каталогу starbase файл index.css наступного вмісту:

```
body {
  background: #263238;
}

#app {
  font-family: 'Avenir', Helvetica, Arial, sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
  text-align: center;
  color: white;
  margin-top: 60px;
}

.character-card {
  border: 2px solid #4FC08D;
  border-radius: 4px;
  cursor: pointer;
}
```

Відкрийте файл index.html і підключіть стилі:

```
<link rel="stylesheet" href="./index.css">
```

Перевірте в браузері результат.

13. Вийдіть на сайт https://swapi.co/ і ознайомтесь з його вмістом. Щоб завантажити ресурси з цього сайту ми скористаємось функцією fetch, що входить до Fetch API. Тож відкрийте файл App.vue, після функції date(){…} поставте кому і додайте:

```
methods: {
   fetchCharacter () {
      fetch('https://swapi.co/api/people', {
       method:'GET'
   }).then(response => console.log(response))
}
```

Далі після тегу </h3> додайте

```
<button class="btn btn-primary" @click="fetchCharacter">Test Fetch Character</button>
```

Перейдіть в браузер Chrome, натисніть Ctrl-Shift-I (або права кнопка і Inspect Code), відкрийте вкладку Console. Далі натисніть кнопку Test Fetch Character і впевніться, що запит спрацював.

Далі застосуємо JSON формат, змінивши трохи ф-ю fetchCharacter :

```
……..
})
  .then(response => response.json())
  .then(json => console.log(json))
}
```

Збережіть зміни і перевірте в браузері результат.

14. Щоб отримувати данні в структурованому вигляді нам знадобиться компонент, який відповідає персоні, назвімо його Character. Тож створіть в каталозі src каталог components, всередині нього файл Character.vue. Компонент складається з шаблону і скрипта. Введіть:

```
<template>
  <div class="">
    {{character}}
  </div>
</template>

<script>
export default {
  data() {
    return {
      character: {}
    }
  },
  methods: {
    fetchCharacter() {
      fetch(`https://swapi.co/api/people/1`, {
        method: 'GET'
      })
        .then(response => response.json())
        .then(json => this.character = json)
      }
    },
  created() {
    this.fetchCharacter()
  }
}
</script>
```

Далі перейдіть в файл App.vue, вставте після тегу <script> рядок

```
import Character from './components/Character.vue‘
```

Видаліть секцію methods: {…} а замість неї вставте

```
components: {
  Character
}
```

Також видаліть рядок <button …>….</button>, а замість нього вставте

```
<Character />
```

Збережіть зміни. Перейдіть в браузер, де буде відображуватися інформація про Luke Skywalker.

15. Покращимо поведінку нашого компонента. Його недолік – відображення тільки першого з персонажів. Тож відкрийте файл Character.vue і перед data() вставте рядок

```
    props: ['id'],
```

Відкоригуйте також функцію fetchCharacter:

```
fetchCharacter(id) {
  fetch(`https://swapi.co/api/people/${id}`, {
```

Змініть також функцію created:

```
this.fetchCharacter(this.id)
```

Відкрийте файл App.vue і задайте властивість :

```
<Character :id="1"/>
```

Збережіть зміни. Оновіть браузер, і ви побачите все ж того Luke Skywalker.

За умовами завдання команда повинна складатися з трьох персонажів. Давайте завантажувати на початку роботи додатка трьох персонажів. id яких задані в масиві initial_ids. Отже в секції даних в кінці рядка title:. . . поставте кому, натисніть Enter і вставте рядок

```
initial_ids: [1, 13, 14]
```

Далі в секції шаблону видаліть рядок <Character…>, а після рядка <h3> вставте

```
<div class="col-md-12">
  <Character
     v-for="id in initial_ids"
    :id="id"
  />
</div>
```

Збережіть зміни і перевірте результат, оновивши браузер. Буде виведена інформація про Luke Skywalker, Chewbacca та Han Solo.

Відкрийте в браузері вкладку **Console**. Ви там побачите попередження "Component lists rendered with v-for should have explicit keys"

Тому введемо індекс:

```
<Character
    v-for="(id, index) in initial_ids"
    :id="id"
    key="index"
/>
```

Перевіримо результат в браузері.

16. Тепер подумаємо, як реалізувати зміну персонажу при кліку. Нам знадобиться метод **switchCharacter**, який додайте після методу **fetchCharacter** (методи відділяйте комою):

```
switchCharacter() {
    let random_id = Math.floor(Math.random() * 83) + 1
    this.fetchCharacter(random_id)
}
```

Відкоригуйте також другий рядок файлу:

```
<div class="" @click="switchCharacter">
```

Перевірте в браузері, як працює клік, змінюючи персонажа.

17. Тепер попрацюємо над структуризацією персонажів (членів нашої команди).

В другому рядку файла Character.vue задамо клас для тега :

```
<div class="col-md-4" @click="switchCharacter">
```

Далі замість {{character}} вставимо теги з вибірковою інформацією про персонажа:

```
<div class="character-card">
  <div class="card-block">
    <h4 class="card-title">{{character.name}}</h4>
    <p class="card-text">Зріст: {{character.height}}cm</p>
    <p class="card-text">Вага: {{character.mass}}kg</p>
    <p class="card-text">Колір волосся: {{character.hair_color}}</p>
    <p class="card-text">Колір очей: {{character.eye_color}}</p>
  </div>
</div>
```

Збережіть зміни і протестуйте результат в браузері.