УДК 004.27

Н.А. Жиляк, Г.А. Шершнёв
Белорусский государственный технологический университет
Минск, Беларусь

ВАЛИДАЦИЯ ФОРМЫ С ПОМОЩЬЮ AJV, VUE.JS И TYPESCRIPT

Аннотация. *Валидация форм является важной частью фронтенд-разработки, которая помогает улучшить пользовательский опыт и предотвратить ошибки при отправке данных на сервер. В этом докладе мы рассмотрим, как использовать библиотеку AJV совместно с Vue.js и TypeScript для создания мощной системы валидации формы.*

N.A. Ghilyak, G.A. Shershniov
Belarusian State Technological University
Minsk, Belarus

FORM VALIDATION USING AJV, VUE.JS AND TYPESCRIPT

Abstract. *Form validation is an important part of front-end development that helps improve the user experience and prevent errors when sending data to the server. In this report, we'll look at how to use the AJV library with Vue.js and TypeScript to create a powerful form validation system.*

Перед отправкой данных на сервер важно убедиться, что все обязательные поля формы заполнены данными в корректном формате. Это называется валидацией на стороне клиента и помогает убедиться, что данные, введённые в каждый элемент формы, соответствуют требованиям.

Валидация на стороне клиента — это первичная проверка введённых данных, которая существенно улучшает удобство взаимодействия с интерфейсом; обнаружение некорректных данных на стороне клиента позволяет пользователю немедленно их исправить. Если же проверка происходит только на сервере, процесс заполнения может быть более трудоёмким, так как требует повторения одних и тех же действий отправки данных на сервер для получения обратного ответа с сообщением о том, что нужно исправить.

JavaScript-валидация кодируется с помощью JavaScript. Она полностью настраиваема, но требует программирования всей логики (или использования библиотеки).

В связи с этим дальнейшим направлением работы будет исследование и разработка методики валидации формы с

использованием библиотеки AJV, фреймворка VueJS и языка программирования TypeScript. Также будет создана система валидации форм, способной обеспечивать точную проверку вводимых данных пользователем.

Задачами работы определены следующие:

– изучение библиотеки AJV, интеграция с VueJS и TypeScript;

– разработка компонентов веб-приложения, включая формы и соответствующие схемы валидации JSON;

– разработка системы обработки ошибок и предоставление пользователю информации о неверно введенных данных.

Библиотека AJV (Another JSON Schema Validator) – это быстрая библиотека валидации данных в формате JSON с поддержкой JSON Schema. JSON Schema – это язык описания структуры и валидации данных в формате JSON. AJV позволяет проверять данные по подготовленным схемам валидации.

Прежде чем начать работать с данной библиотекой, пользователь должен убедиться в наличие на рабочем компьютере:

– Node.js v18.16.1;

@vue/cli 5.0.8.

Создадим новый проект с помощью Vue CLI с такими параметрами:

```
vue create ajv-validation
```

Далее необходимо установить необходимые зависимости в проекте:

```
npm install ajv ajv-formats ajv-errors
```

И следующим шагом будет создание файлов схемы валидации login.json:

```
{
 "$id": "/login.json",
 "type": "object",
 "additionalProperties": false,
 "required": ["login", "password"],
 "properties": {
  "login": {
   "type": "string",
   "format": "email",
   "errorMessage": "enter a valid email address"
  },
  "password": {
   "type": "string",
   "minLength": 6,
   "maxLength": 1024
  }
 }
}
```

В этой схеме определяется тип каждого поля (строка), а также устанавливаются некоторые правила валидации, такие как формат e-mail и минимальная длина пароля. Поля "login" и "password" обязательны для заполнения.

Далее создается компонент формы:

```
<template>
  <div class="login-form">
    <h2>Login</h2>
    <form @submit.prevent="onLogin">
      <div class="form-group">
        <label for="login">Email</label>
        <input
          v-model="formData.login"
          :class="{ 'is-invalid': isInvalid(errors.has('login')) }"
          @blur="onBlur"
          type="text"
          id="login"
          placeholder="Enter your email"
        />
        <ul class="error-wrapper">
          <li v-for="errorMsg of errors.get('login')" :key="errorMsg">
            {{ errorMsg }}
          </li>
        </ul>
      </div>

      <div class="form-group">
        <label for="password">Password</label>
        <input
          v-model="formData.password"
          :class="{ 'is-invalid': isInvalid(errors.has('password')) }"
          @blur="onBlur"
          type="password"
          id="password"
          placeholder="Enter your password"
        />

        <ul class="error-wrapper">
          <li v-for="errorMsg of errors.get('password')" :key="errorMsg">
            {{ errorMsg }}
          </li>
        </ul>
      </div>

      <button type="submit" :disabled="errors.size > 0">
        Login
      </button>
    </form>
  </div>
</template>
```

Тут ведется работа с двумя методами:

– метод onLogin — это метод, который вызывается при попытке входа пользователя в систему (логине). Он выполняет проверку

и валидацию введенных пользователем данных и предпринимает соответствующие действия в зависимости от результата проверки. А именно, выполняет проверку введенных данных на корректность с помощью функции валидации validate. Эта функция использует схему валидации данных и проверяет соответствие данных этой схеме. Возвращается флаг isValid, который указывает, прошла ли валидация успешно. Если данные некорректны (isValid равен false), выполняется обработка ошибок;

```
function onLogin() {
    errors.value.clear();
    if (validator.validate("/login.json", formData.value)) {
      // valid, do nothing
    } else if (validator.errors?.length) {
      for (const [, e] of validator.errors.entries()) {
        if (!e.message) {
          continue;
        }
        const fieldName = e.instancePath.substring(1);
        const fieldErrors: string[] = errors.value.get(fieldName) || [];
        fieldErrors.push(e.message);
        errors.value.set(fieldName, fieldErrors);
      }
    }
  }
```

– метод onBlur — это метод, который вызывается при работе события "blur" (потеря фокуса) на текстовом поле ввода формы. Он используется для валидации данных, введенных пользователем, когда пользователь переходит с поля на другой элемент формы или выполняет манипуляции активизации вне текстового поля.

```
    function onBlur(e: any) {const fieldName = e.target.id;
    const fieldValue = e.target.value;
    if (
      validator.validate(`/login.json#/properties/${fieldName}`, fieldValue)
    ) {
      errors.value.delete(fieldName);
    } else if (validator.errors?.length) {
      errors.value.set(
        fieldName,
        validator.errors.map((e) => e.message) as string[]
      );
    }
  }
```

А именно:

а) получение имени (идентификатор) и значения поля ввода, в котором «произошло событие» "blur";

б) получение схемы валидации для данного поля;

в) выполнение проверки значения поля на корректность с помощью функции валидации validate;

г) если значение поля некорректно (isValid равен false), выполняется обработка ошибки.

Таким образом, разработка и внедрение системы валидации формы с помощью AJV, VueJS и TypeScript позволят значительно улучшить процесс валидации данных на клиентской стороне. Это позволит снизить вероятность ошибок и повысить производительность веб-приложения, а также обеспечить корректную обработку данных на сервере.

Список использованных источников

1. Node.js [Электронный ресурс] – Режим доступа: https://nodejs.org/en – Дата доступа: 03.11.2023.
2. TypeScript [Электронный ресурс] – Режим доступа: https://www.typescriptlang.org/ – Дата доступа: 03.11.2023
3. Vue.js [Электронный ресурс] – Режим доступа: https://vuejs.org/. – Дата доступа: 03.11.2023.
4. JSON Schema [Электронный ресурс] – Режим доступа: https://json-schema.org/ – Дата доступа: 03.11.2023.
5. AJV [Электронный ресурс] – Режим доступа: https://ajv.js.org/ – Дата доступа: 03.11.2023.

УДК 004.27

Н.А. Жиляк
Белорусский государственный технологический университет
Минск, Беларусь

ПЕРСПЕКТИВЫ РАЗВИТИЯ ТЕХНОЛОГИИ BIG DATA

Аннотация. *В рамках статьи изучаются и анализируются методы и алгоритмы реализации больших объемов данных. Будут рассмотрены теоретические аспекты, связанные с появлением феномена больших данных, выявлена эпистемология и эвристические возможности больших данных.*

N. Ghilyak
Belarusian State Technological University
Minsk, Belarus

PROSPECTS FOR THE DEVELOPMENT OF BIG DATA TECHNOLOGY