

Nuxt3 mit Vuetify 3 – Teil 3: Entwickeln in Vue / Typescript

Typescript muss in VS Code installiert werden. -g steht für globally (Alle Projekte)

```
[crayon-6776a9515b0a9850699670/]
```

Typescript Version prüfen

```
[crayon-6776a9515b0b1484376369/]
```

Enums

```
[crayon-6776a9515b0b3172184421/]
```

Objekte werden mit `reactive<type>({...})` reaktiviert und primitive Datentypen mit `ref<string>(…)`

```
[crayon-6776a9515b0b4260090890/]
```

Mixins werden in eine Vue Instanz hinzugefügt. So als würde man 2 Vue Instanzen verschmelzen. Composables sind dem ähnlich. In Mixins wird alles importiert, bei Composables muss jede Variable, Methode, etc. einzeln importiert werden.

Guter Artikel dazu:

[What is a Vue.js Composable? – Vue.js Tutorials \(vueschool.io\)](https://vueschool.io/)

Nuxt3 mit Vuetify 3 – Teil 2:

Layout

Wenn man durch eine App navigiert, dann ist der Seitenaufbau überall identisch. Das Menü, die Kopfzeile und Fusszeile sind auf jeder Seite identisch. Lediglich der Inhalt ändert sich.

In Nuxt 3 werden Layouts im „layouts“-Ordner erstellt und anschließend in die app.vue hinzugefügt. Einzelne Seiten werden im „pages“-Ordner erstellt.

Nuxt 3 routet die pages anhand des Namens. Nur die page „index.vue“ wird zum root Knoten geroutet.

Also erstellen wir ein Verzeichnis „layouts“ und ein Verzeichnis „pages“.

In das „layouts“ Verzeichnis, eine Datei erstellen „default.vue“.

und folgenden Inhalt kopieren:

```
[crayon-6776a9515b654484003483/]
```

Dort, wo `<slot />` steht, kommt der Inhalt der einzelnen Pages hinein.

Jetzt in das Verzeichnis „pages“

die Datei index.vue mit folgenden Inhalt

```
[crayon-6776a9515b658926806647/]
```

und die Datei „pageOne.vue“ mit folgenden Inhalt erstellen

```
[crayon-6776a9515b65a136076354/]
```

In der app.vue müssen wir nun sagen, dass wir nicht mehr die Nuxt Welcome page sehen wollen, sondern unser Layout:

```
[crayon-6776a9515b65b242128060/]
```

Nuxt3 mit Vuetify 3 – Teil 1: Installation

Wenn man aus der Microsoft Welt kommt und Webseiten mit JQuery bearbeitet hat, dann sind UI Frameworks wie Vue eine wahre Veränderung. In meiner ASP WebApp habe ich Vue2 über CDN eingebunden und Vue anstatt JQuery genutzt. Vue funktioniert aber am besten mit NodeJs. Der Versuch NodeJs zu erlernen verlief aber immer in Frust. Weil es keinen roten Faden gibt, wie etwas umgesetzt werden kann. Es gibt keinen besten Weg. Es gibt sehr viele neue Begriffe und Definitionen, die Stück um Stück erlernt werden müssen. Es ist definitiv ein anderes lernen wie eine erprobte Sprache wie C# oder Java.

Deshalb ist hier **mein** roter Faden. Ziel soll am eine App sein, die folgende Punkte mit sich bringt:

1. Nuxt 3 als Wrapper über Vue3. Nuxt 3 schafft eine Ordnerstruktur und macht die imports z.B. überflüssig.
2. Vuetify 3 Beta. Zwar ist es noch in Beta Phase, aber es wird das beste UI Framework (Meiner Meinung nach)
3. Typescript: Die App soll in Typescript geschrieben sein
4. Pinia wird unser Statemanagement anstatt Vuex
5. Übersetzungen werden über i18n realisiert
6. Daten werden über externe .NET 6 Minimal Api mit Swagger abgerufen
7. Authorisierung erfolgt über eine ASP .NET 6 Proxy Anwendung

Installation

Voraussetzung

NodeJS, Visual Studio Code, Volar Extension. Aktuelle Links

sind hier zu finden:

Nuxt 3 – Quick Start (nuxtjs.org)

In Visual Studio über „Strg + Shift + ö“ das Terminal öffnen

Neue App erstellen:

```
[crayon-6776a9515b898155868852/]
```

Die App wird nun im Unterverzeichnis „nuxt-app“ erstellt.

Daher müssen wir in das neue Verzeichnis navigieren

```
[crayon-6776a9515b89c108830867/]
```

packages nachinstallieren:

```
[crayon-6776a9515b89e766058822/]
```

Vuetify installieren

Referenzen:

The Easy Way to add Vuetify Vue.js 3 Material Components to Your Nuxt 3 / Vue.js 3 Application – YouTube

[Solved] vuetify icon not showing – Local Coder (Solution 5)

vuetify – npm (npmjs.com)

Unter „Version“ die neuste Version raussuchen

```
[crayon-6776a9515b89f572478331/]
```

Sass muss ebenfalls für Vuetify installiert werden

```
[crayon-6776a9515b8a1058807204/]
```

Pinia als StateManager

```
[crayon-6776a9515b8a2654717812/]
```

```
[crayon-6776a9515b8a3299209130/]
```

und die mdi Icons für Vuetify

```
[crayon-6776a9515b8a4218539649/]
```

Plugins werden in Nuxt im plugins Verzeichnis erstellt. Also muss im Root ein neues Verzeichnis „plugins“ erstellt

In das plugins Verzeichnis wird eine neue Datei „vuetify.ts“ erstellt und folgendes kopiert:

[crayon-6776a9515b8a5586157814/]

Jetzt muss noch die nuxt.config.ts angepasst werden und das css und der transpiler müssen mit vuetify verheiratet werden. Daher kann der folgende Teil einfach copy/pasted werden. Der Vite Teil wird bei mir als Fehler angezeigt. kompilieren geht trotzdem. Wem das stört, kann den vite Knoten auskommentieren [crayon-6776a9515b8a7158658609/]

Jetzt sollte die App starten können:

In der Beta 1 läuft dauert der Start über 1 Minute. Ich hoffe das wird noch gefixt

[crayon-6776a9515b8ba000233218/]

Weiter geht's mit Teil 2 Layout:
<https://devandy.de/?p=890&preview=true>

Cmd Pfad autocomplete funktioniert nicht

Cmd öffnen

[crayon-6776a9515baad333586585/]

Befehle eingeben

Cmd neustarten und mit cd Tab den Pfad autocompleten lassen ☐

Applikationsdesign:

Datensätze abarbeiten nach Punktesystem

Mir kam mal folgende Problemstellung in die Gedanken:

Mal angenommen ich muss unterschiedliche Aufgaben abarbeiten. Diese befinden sich in einer Warteschleife. So dass sie nacheinander oder auch asynchron parallel abgearbeitet werden sollen.

Jede Aufgabe besitzt eine Priorität. Sagen wie 1 (unwichtig) – 10 (sehr wichtig). In die Warteschleife kommen immer wieder Aufgaben nach. Das Problem ist, es werden die sehr wichtigen immer zuerst abgearbeitet. Die weniger wichtigen kommen daher nie dran und werden nie verarbeitet. Die Lösung ist ein Punktesystem mit Zeitangabe. Die Priorität ist dient dann als Faktor und wird mit der Anzahl Minuten multipliziert. Das ergibt die Summe der Punkte pro Datensatz

Beispiel: Es ist Momentan 12:00 Uhr Mittags.

1. Datensatz 1: Prio 4, Hinzugefügt um: 11 Uhr, Punkte: $4 * 60 = 240$
2. Datensatz 2: Prio 9, Hinzugefügt um: 8 Uhr, Punkte: $9 * 240 = 2160$
3. Datensatz 3: Prio 2, Hinzugefügt um: 4 Uhr, Punkte: $2 * 480 = 960$

Sortiert man diese Liste nun nach Punkten ergibt sich eine neue Reihenfolge

1. Datensatz 2: Prio 9, Hinzugefügt um: 8 Uhr, Punkte: $9 * 240 = 2160$
2. Datensatz 3: Prio 2, Hinzugefügt um: 4 Uhr, Punkte: $2 * 480 = 960$
3. Datensatz 1: Prio 4, Hinzugefügt um: 11 Uhr, Punkte: $4 * 60 = 240$

So hat jeder Datensatz eine Chance drangenommen zu werden. Je länger er in der Warteschleife verweilt, desto höher wird sein Punktesatz.

Windows Dienst installieren / Löschen

```
sc.exe create MyServiceName binpath=
„C:\inetpub\wwwroot\...exe“
sc.exe delete MyServiceName
```

Bibliotheken für Authorisierung

Das Thema Authorisierungen ist sehr komplex und es gibt dazu viele Ansätze. Hier ist eine Liste von Bibliotheken:

Casbin

Vergleich Casbin mit OPA: OPA vs Casbin (github.com)

<https://www.openpolicyagent.org/>

oso Documentation – oso Documentation (osohq.com) (Nur Python und Java)

ory/ladon: A SDK for access control policies: authorization for the microservice and IoT age. Inspired by AWS IAM policies. Written for Go. (github.com) (Veraltet)

teramoby/speedle-plus: Speedle+ is an open source project for access management. It is based on Speedle open source project and maintained by previous Speedle maintainers. (github.com)

VueCliMiddleware unter IIS installieren

Dies ist eine Fortsetzung zu

<https://hosting149637.a2ebb.netcup.net/devandy.de/vuecli-installieren/>

1. Über Windows Cmd Console zum Ordner navigieren wo das neue Zertifikat erstellt werden soll (cd...)
2. in die Console folgendes eingeben: `mkcert -pkcs12 localhost 127.0.0.1`
3. Im Ordner wird nun eine `localhost+1.p12` Datei erstellt. Diese Datei umbenennen zu `localhost.pfx`
4. In Die Console `mkcert - install` eingeben und mit Ja bestätigen
5. Die `localhost.pfx` Datei doppelt anklicken und „Lokaler Computer auswählen“. Ganz wichtig. Sonst funktioniert es nicht
6. Bei Kennwort „changeit“ eingeben und „Schlüssel als exportierbar machen“ anhaken
7. Zu IIS wechseln, ASP Core App mit VueCliMiddleware auswählen, und auf der rechten Seite auf „Bindungen...“ klicken.
8. Doppelt auf die https Sitebindung klicken
9. Auf den Button „Auswählen“ klicken und gerade installiertes Zertifikat auswählen und mit OK bestätigen
10. Bei Hostname noch localhost eintragen

11. Zur VueApp wechsel und in die VueConfig folgendes eintragen:
12. [crayon-6776a9515bc40518508877/]
Der Port 8080 sollte gleich sein, wie in VueCliMiddleware angegeben. Die localhost.pfx Datei muss im angegebenen Pfad vorhanden sein

Hot reloading funktioniert leider noch nicht

VueCli installieren

Begriffe Definition

Babel

Ein JavaScript Compiler. Übersetzt Code in EcmaScript, welcher auch mit alten JavaScript Browsern kompatibel ist und vieles mehr

PostCss

Beim kompilieren werden Prefixe an Css Klassen / Ids hinzugefügt, damit in einer SinglePage Applikation nicht plötzlich ein und dieselbe Id für unterschiedliche Komponente genutzt werden

Lint

Ein Prozess wo der Source Code nach potentiellen Fehler durchsucht wird

package.json

Beschreibt die Abhängigkeiten (dependencies) inkl. Version zu anderen Bibliotheken

Jede App in NodeJs hat im Root Verzeichnis der App so eine Datei.

Beim Abrufen von einer App, werden die Abhängigkeiten

automatisch mit herunter geladen

sass / less

Dateien werden von z.B. Webpack zu einer Css Datei interpretiert

TypeScript (.ts)

Dateien werden von z.B. Webpack zu einer Js Datei interpretiert
Dadurch sind z.B. Klassen und Vererbungen in JavaScript möglich

NodeJs

Ist eine JavaScript Serverumgebung

Npm

Node-package-manager verwaltet Node Applikationen (packages)

Vue.js

Vue.js ist ein clientseitiges JavaScript-Webframework zum Erstellen von Single-Page-Webanwendungen

Vuex

Nutzt man mehrere Komponenten, die untereinander noch verschachtelt sind, dann müssen Events immer nach oben gereicht werden.

Mit Vuex wird ein Datenstore geschaffen, der alle Daten abrufen und bereit stellt. Dadurch erhält man eine Zustandsmaschine

Webpack

Module Bundler. Packt mehrere JavaScript, Css usw. Dateien zu einer minified zusammen

VueCLI

Vue ist das JavaScript Framework und CLI steht für Command Line Interface

Installation VueCLI

1. NodeJs installieren x64 -> <https://nodejs.org/en/download/>
von nun an werden die Befehle in der NodeJs cmd eingegeben.

Node.js Command prompt:

C:\Program Files\nodejs\nodevars.bat

2. Vue Cli installieren -> npm i -g @vue/cli

Die Apps werden hier gespeichert: %UserProfile%/node_modules

3. vue ui -> startet den Service für Vue Cli Ui Interface
http://localhost:8000/dashboard

4. Vue global startbar machen
npm install -g @vue/cli-service-global

5. In PowerShell mkcert installieren
choco install mkcert

Damit erstellt man für die eigene Entwicklungsmaschine ein SSL Zertifikat. Zertifikate von CA können nicht mit localhost arbeiten und Self Signed werden von Applikationen nicht anerkannt

6. Über cd [... Pfad zu meiner Applikation] navigieren

7. mkcert -install

8. Ipconfig -> Eigene IP rausfinden (IPv4-Adresse
. □

9. (Durch eigene IPv4 ersetzen) mkcert localhost 127.0.0.1
192.168.178.47 ::1

Ordnerstruktur Vue App

node_modules Ordner

Hier liegen alle Bibliotheken (Modules), die für das Vue Projekt benötigt werden

public/Index.html

Die index.html ist die einzige Html Seite, die an den Client übertragen wird. Der Inhalt wird in sie rein gerendert. Und zwar hier zwischen:

Sobald man buildet, werden dort auch 2 Skript Tags generiert, die den JavaScript Teil enthalten

src/assets (Engl. Teile, Zubehör)

Hier liegen die Bilddateien, Fonts etc.

src/components

Hier liegen eigene Vue Komponente

src/App.vue

Beispiel Komponente

src/main.js

Diese JavaScript Datei ist die Haupt Datei, die sagt, nehme

die und die JavaScript Komponente und rendere sie in Vue hinein

.gitignore

Regeln für Git, welche Dateien in das Projekt kommen und bei Änderungen hochgeladen werden sollen

babel.config.js

Einstellung von Babel, z.B welche Version von EcmaScript genutzt werden soll

package.js

Konfigurationsdatei vom Vue Projekt.

- Name, Version
- „scripts“ welcher NodeJs Service beim builden, deployen und linten genutzt werden soll
- „dependencies“ welche Bibliotheken in welcher Version genutzt werden sollen
- „devDependencies“ welche Bibliotheken ausschließlich im Entwicklungsmodus genutzt werden sollen

Postcss.config.js

Konfigurationsdatei für PostCss

README.md

Dokumentationsdatei. Wird in GitHub z.B. als Doku angezeigt:

<https://guides.github.com/features/mastering-markdown/>

Tsconfig.json

Konfigurationsdatei für TypeScript

Tslint.json

Konfigurationsdatei für TypeScript Lint

dist Ordner

Mit dem Befehl „npm run build“ wird die App deployed und in das Verzeichnis dist gepackt. Auf einen produktiven Server kommt also nur der Inhalt vom dist Ordner

Openmediavault + Nextcloud auf Raspberry Pi installieren

Installation erfolgt von einem Windows Rechner.

Benötigt werden folgende Programme:

1. SD Card Formatter: <https://www.sdcard.org/downloads/formatter/>
2. Win32 Disk Imager: <https://sourceforge.net/projects/win32diskimager/>
3. Raspberry Pi OS (32-bit) Lite Image: <https://www.raspberrypi.org/downloads/raspberry-pi-os/>
4. KiTTY oder PuTTY (SSH Client) [Im folgenden Konsole genannt]: <https://www.fosshub.com/KiTTY.html>

Vorbereitung:

1. Mit SD Card Formatter SD Karte formatieren
2. Mit Win32 Disk Imager „Raspberry Pi OS (32-bit) Lite Image“ auf SD Karte schreiben
3. Auf der SD Karte im Root Verzeichnis eine Datei, mit Dateinamen „ssh“(ohne Dateiendung) anlegen
4. SD Karte in Raspberry schieben, RasPi neustarten und IP Adresse über Router herausfinden
5. In der Konsole mit „pi“ und „raspberrry“ einloggen

Installation Webmin + Nextcloud:

[crayon-6776a9515bd88423608017/]

Web einloggen <https://192.168.178.45:10000/>

mit pi und raspberrry

Servers -> MySQL Database Server -> „Create new database“ -> „nextcloud“

-> „User Permissions“ -> oben „Create new user“ ->

Radio auf leer und username eingeben „nextclouduser“

Password „Set to“ und password eingeben

Hosts Radio auf leer und „localhost“ eingeben
Permissions alles markieren
Navigations Menü: „Others“ -> „File Manager“ -> etc -> openal
-> php -> 7.3 -> apache2 -> php.ini (r.Maustaste) -> editieren
-> memory_limit 2048 -> oben rechte Save Diskette
-> home -> pi -> „File“ -> „Download from remote URL“ ->
nextcloud.com/de/install/#instructions-server (r.Maustaste)
auf download -> link kopieren
download -> r.Maustaste auf Datei und „Extract“
Ins Verzeichnis gehen -> Alle markieren -> Cut
var -> www -> html -> nextcloud Ordner erstellen und
reinkopieren
html Ordner r.Maustaste -> Properties -> Change ownership ->
Username, Group: „www-data“, recursive -> true

Navigations Menü: „System“ -> „Bootup and Shutdown“ -> apache2
-> „Restart“

####

Apache Port ändern auf z.B. 81. Damit OpenMediaVault auf Port
80 und 444 SSL installiert werden kann. Dies wird später
zurück geändert

/etc/apache2/ports.conf

Nextcloud installieren

<http://192.168.178.45/nextcloud/index.php>

Installation OpenMediaVault:

1. [crayon-6776a9515bd8b486522823/]
2. OpenMediaVault ist nun installiert. Wenn man die
IP Adresse im Browser aufruft, sollte die WebUi
erscheinen. Einzuloggen mit „admin“ und
„openmediavault“
3. Standard Passwort ändern: General Settings -> Web
Administrator Password
4. SSH Zertifikat: Certifikates -> Tab SSH -> Add ->

- Irgendetwas eingeben. z.B. SSHCert -> Apply
5. SSL Zertifikat: Certifikates -> Tab SSL -> Add -> ausfüllen -> Apply
 6. General Settings: Port -> 90, Auto logout -> disabled, Enable SSL auf true, Zertifikat auswählen, Force SSL auf true -> save & apply [WebUi ist ab jetzt nur mit https erreichbar]

In Webmin Port von Nextcloud zurück stellen:

/etc/apache2/ports.conf auf Port 80 bzw. 443

#Laufwerk dauerhaft einbinden:

```
sudo mount -t ext /dev/sda1 /media/Hdd1
```

<https://confluence.jaytaala.com/display/TKB/Mount+drive+in+linux+and+set+auto-mount+at+boot>