Single Page Architectures with Vue.JS and ASP.NET Core

Kevin Griffin

Hello, I’m Kevin!

11-time Microsoft MVP
Focused on ASP.NET / ASP.NET Core / Azure

twitter.com/1kevgriff
kevin@consultwithgriff.com
consultwithgriff.com

Kevin Griffin
Owner, Swift Kick
Get the Resources

https://consultwithgriff.com/spas-with-vuejs-aspnetcore

Client-Side Applications of the Past (for .NET)
New Ways To Develop Client-Side Applications

- CLI-based

- Development and Deployment happens outside of the backend application

- Front end doesn't care where it is served from.

- Front end doesn't care where it gets data from.
In an ASP.NET Core world, how can be better support these types of applications?

Goals of this talk

- Discuss my approach for building SPAs with VueJS and ASP.NET Core

- Ask and answer the logical questions I’ve asked myself while learning this process.

- Explain WHY certain libraries are used, instead of just saying “use this and that”
Things you will NOT learn

- Everything about VueJS
- Everything about ASP.NET (Core)
- But – the examples should give you an idea of what’s going on.

Architecture of a Single Page Application

ASP.NET Core Application (Back-end)
- Data Access
- Authentication
- Authorization

Vue.JS Application (Front-end)
- Routing
- State Management (Vuex)
- Data fetching (Axios)
Question: What do I need to know about Vue?

- VueJS is a client-side framework for building web applications.
- It is component-based.
- Each component can maintain its own:
  - Template
  - Styles
  - State
  - Properties
  - Events
- Limited opinion
Question: How do I set up a VueJS application?

Step 1: Installing Vue CLI

```bash
npm install -g @vue/cli
# OR
yarn global add @vue/cli
```
Why use the Vue CLI?

- Vue applications don’t require the Vue CLI to be built.
- The CLI helps tremendously in scaffolding the base application.
- Hot reload ensures you are always looking at the current code.
- Version upgrades are easier because dependencies are hidden from you.
Step 2: Create a Vue.JS Application

$ vue create frontend

Vue CLI v4.0.5
Please pick a preset: (Use arrow keys)
- default (babel, eslint)
- Manually select features

- Babel
  - TypeScript
  - Progressive Web App (PWA) Support
  - Router
  - Vuex
  - CSS Pre-processors
  - Linter / Formatter
  - Unit Testing
  - E2E Testing
Step 2: Create a Vue.JS Application

Vue CLI v4.0.5
1. Please pick a preset: Manually select features
2. Check the features needed for your project: (Press <space> to select, <a> to toggle all, <i> to invert selection) Babel, Linter
3. Pick a linter / formatter config: (Use arrow keys)
   - ESLint with error prevention only
     - ESLint + Airbnb config
     - ESLint + Standard config
     - ESLint + Prettier

(*) Lint on save
( ) Lint and fix on commit
Step 2: Create a Vue.JS Application

Vue CLI v4.0.5

- Please pick a preset: Manually select features
- Check the features needed for your project: (Press <space> to select, <a> to toggle all, <i> to invert selection)Babel, Linter
- Pick a linter / formatter config: Basic
- Pick additional lint features: (Press <space> to select, <a> to toggle all, <i> to invert selection)Lint on save
- Where do you prefer placing config for Babel, PostCSS, ESLint, etc.? (Use arrow keys)
- In dedicated config files
  In package.json

Step 2: Create a Vue.JS Application

Vue CLI v4.0.5

- Creating project in C:\spa\frontend.
- Initializing git repository...
- Installing CLI plugins. This might take a while...

[---------------------] - fetchMetadata: $ pacote range manifest for wrap-ansi@^5.1.0 fetched i
Step 2: Create a Vue.JS Application

- Installing additional dependencies...
- Added 56 packages from 44 contributors and audited 24408 packages in 16.333s
- Found 0 vulnerabilities
- Running completion hooks...
- Generating README.md...
- Successfully created project frontend.
- Get started with the following commands:
  
  ```
  $ cd frontend
  $ npm run serve
  ```

Step 3: npm run serve

- Generates a “development” build of the application
  - All JavaScript, HTML, CSS, and Images are built and combined – but for development
  - Source maps allow for easy debugging
  - Enables development server – which handles hot reload responsibilities and real-time rebuilding of the application.
Tada!

Welcome to Your Vue.js App

For a guide and recipes on how to configure / customize this project, check out the vue cli documentation.

Installed CLI Plugins

- vue-router
- vue-cli

Essential Links

- Core Docs
- Forum
- Community Chat
- Twitter
- News

Ecosystem

- vue-router
- vue-cli
- vue-cli-tools
- vue-hakku
- awesome-vue
Create an ASP.NET Core Application

C:\spa> dotnet new web --backend
The template "ASP.NET Core Empty" was created successfully.
Processing post-creation actions...
Running 'dotnet restore' on backend\backend.csproj...
  Restore completed in 80.01 ms for C:\spa\backend\backend.csproj.
Restore succeeded.
C:\spa> ls

Directory: C:\spa

<table>
<thead>
<tr>
<th>Mode</th>
<th>LastWriteTime</th>
<th>Length</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>d----</td>
<td>11/6/2019 2:58 PM</td>
<td></td>
<td>backend</td>
</tr>
<tr>
<td>d----</td>
<td>11/6/2019 3:30 PM</td>
<td></td>
<td>frontend</td>
</tr>
</tbody>
</table>

When you're done...
Question: How does a VueJS SPA work?

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.8">
  <link rel="icon" href="% BASE_URL %favicon.ico">
  <title>frontend</title>
</head>
<body>
  <div id="app"></div>
  <!-- built files will be auto injected -->
</body>
</html>
```
Question: How does a VueJS SPA work?

```html
<App>
  <Slot />
  https://myDomain/routeA
</App>
```

Question: How do I tell VueJS what components to use for routes?
Question: How do I tell VueJS what components to use for routes?

The official router for Vue.js. http://router.vuejs.org/

Question: How do I tell VueJS what components to use?

```javascript
import Vue from 'vue' 68.1K (gzipped: 24.3K)
import VueRouter from 'vue-router' 26.4K (gzipped: 9.2K)
import Home from 'views/Home.vue'
Vue.use(VueRouter)

const routes = [
  {
    path: '/',
    name: 'home',
    component: Home
  },
  {
    path: '/about',
    name: 'about',
    component: () => import('views/About.vue')
  }
]

const router = new VueRouter({
  mode: 'history',
  base: process.env.BASE_URL,
  routes
})
```
Question: Does Vue download all my components when the page renders?

- Yes!

- Lazy loading is available *if* you don’t want that to happen.

```javascript
{
  path: '/about',
  name: 'about',
  component: () => import(/* webpackChunkName: "about" */ './views/About.vue')
}
```
Question: What makes up a Vue component?

- There are several ways to build Vue components – and we cannot cover all them.

- In this architecture, we use Vue with “Single File Components”

- Three distinct sections:
  - HTML template
  - CSS styles
  - Script

```html
<template>
  <div class="home">
    <img alt="Vue logo" src="/assets/logo.png">
    <h2>HelloWorld msg="Welcome to Your Vue.js App"/>
  </div>
</template>

<style scoped>  
</style>

<script>
import HelloWorld from '@/components/HelloWorld.vue'
export default {
  name: 'home',
  components: {
    HelloWorld
  }
}
</script>
```
Question: What does development look like?

Architecture of a Single Page Application (Dev)

- **Browser**
- **Network**
  - ASP.NET Core Application (Back-end) | https://localhost:14500
  - Vue.js Application (Front-end) | https://localhost:8081
Question: What if I want *one* url during development?
Architecture of a Single Page Application (Dev)

Browser ➔ Network ➔ ASP.NET Core Application (Back-end)

Proxy ➔ Vue.JS Application (Front-end)

https://localhost:14500

Set up the proxy

Microsoft.AspNetCore.SpaServices.Extensions 3.0.0

Helpers for building single-page applications on ASP.NET MVC Core.

This package was built from the source code at https://github.com/aspnet/AspNetCore/tree/aadfe4e053f1a55aa62d7f7b3b88bed72f71babf

There is a newer prerelease version of this package available. See the version list below for details.

1. dotnet add package Microsoft.AspNetCore.SpaServices.Extensions --version 3.0.0
Set up the proxy

```csharp
public void ConfigureServices(IServiceCollection services)
{
    services.AddSpaStaticFiles(configuration: options => { options.RootPath = "wwwroot"; });
    services.AddControllers();
}
```

Set up the proxy

```csharp
public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }
    app.UseRouting();
    app.UseEndpoints(endpoints => { endpoints.MapControllers(); });
    app.UseSpa(configuration: builder =>
    {
        if (env.IsDevelopment())
        {
        }
    });
}
```
Run Vue Development Server

```
C:\spa\frontend [master]> npm run serve
> frontend@1.0 serve C:\spa\frontend
> vue-cli-service serve

INFO Starting development server...

DONE Compiled successfully in 4461ms

3:18:11 PM

App running at:
- Local: http://localhost:8080/

Note that the development build is not optimized.
To create a production build, run `npm run build`.
```
Question: How do I deploy?

Deployment

C:\spa\frontend [master]> npm run build
> frontend@0.1.0 build c:\spa\frontend
> vue-cli-service build

\ Building for production...

DONE Compiled successfully in 7899ms

<table>
<thead>
<tr>
<th>File</th>
<th>Size</th>
<th>Gzipped</th>
</tr>
</thead>
<tbody>
<tr>
<td>dist\js\chunk-vendors.e4831489.js</td>
<td>88.92 KB</td>
<td>31.63 KB</td>
</tr>
<tr>
<td>dist\js\app.3c15b7f6.js</td>
<td>1.62 KB</td>
<td>0.64 KB</td>
</tr>
<tr>
<td>dist\css\app.e2713bb0.css</td>
<td>0.33 KB</td>
<td>0.23 KB</td>
</tr>
</tbody>
</table>

Images and other types of assets omitted.

DONE Build complete. The dist directory is ready to be deployed.

INFO Check out deployment instructions at https://cli.vuejs.org/guide/deployment.html
Deployment

```csharp
public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }
    app.UseRouting();
    app.UseEndpoints(endpoints => { endpoints.MapControllers(); });
    app.UseStaticFiles();
    app.UseSpa(configuration: builder =>
    {
        if (env.IsDevelopment())
        {
        }
    });
}
```
Griff’s Tip

- Use the “outputDir” option in the vue.config.js file to set where npm run build will deploy its built files too.

Question: How does VueJS make API calls to ASP.NET Core?
Question: How does VueJS make API calls to ASP.NET Core?

- VueJS does not have a built-in AJAX or HTTP client.

- I recommend Axios – but *any* JavaScript fetch-based client will work.

- In newer browsers, fetch will work just fine.
Question: How do I share state between components?

- Manage state in a higher level component
- Great for downstream data
- Horrible for upstream data
- Difficult to keep state in sync
Question: How do I share state between components?

- Use a global state management library, such as Vuex

Vuex in a Nutshell
Vuex in a Nutshell

- “State” is an object that is available to any Vue components that request it.

- Objects in state must be pre-defined for change tracking.

```javascript
state: {
  globalNumber: 42,
  error: {
    loadUsers: "",
  },
  users: [],
  products: [{ name: "Default Product", price: 3.50 }],
},
```

Vuex in a Nutshell

- “Mutations” are processes that change state.

- Change will bubble out to any components that need to know about them.

- How do you call a mutation?

```javascript
mutations: {
  loadUsersSuccess(state, data) {
    state.users = data;
  },
  loadUsersError(state, error) {
    state.error.loadUsers = error;
    state.globalNumber = -42;
  },
},
```
Vuex in a Nutshell

- "Actions" are methods that trigger one or multiple mutations.

```javascript
actions: {
  loadUsers({ commit }) {
    axios
      .get("https://localhost:14590/api/users")
      .then((response) => commit("loadUsersSuccess", response.data),
            (error) => commit("loadUsersError", error));
  }
}
```

Vuex in a Nutshell

- State can be unloaded into a component via the "mapState" helper.

```javascript
<template>
  <div class="hello">
    <p>Global Number is {{someNumber}}</p>
    <ul>
      <li v-for="u in users" :key="u.id">{{u}}</li>
    </ul>
  </div>
</template>

<script>
import {mapActions, mapState} from "vuex";
export default {
  name: 'HelloWorld',
  computed: {
    ...mapState({
      users: state => state.users,
      someNumber: state => state.globalNumber
    })
  },
  props: {
    msg: String
  }
</script>
```
Vuex in a Nutshell

- Actions can be called via the "mapActions" helper.

```javascript
<script>
import { mapState, mapActions } from "vuex";

export default {
  name: "HelloWorld",
  computed: {
    ...mapState({
      users: state => state.users,
      someNumber: state => state.globalNumber
    })
  },
  methods: {
    ...mapActions(['loadUsers']), // via Vuex actions
    localLoad() { // local method
      this.loadUsers();
    }
  },
  props: {
    msg: String
  }
}</script>
```

Question: Authentication?
Question: Authentication?

- **Cookie Authentication**
  - Works well in same-domain systems
  - Very well supported in ASP.NET Core
  - Automatically sent to the server on a request
  - Easy to revoke

- **JWTs (JSON Web Tokens)**
  - New “hotness”
  - Not necessarily more or less secure than Cookies.
  - JSON-based, so you can add metadata to the token
  - Not sent to the server by default.
  - In ASP.NET Core, you need to write your own refresh and revoking processes.

---

Question: Authentication

- Do not confuse “authentication” with “authorization”
- Never trust *anything* on the client
- Server will always be the single point of truth – and this is for your protection!
- Handle your errors!

```csharp
[Route("template:\"api/user\")]
[ApiController]
[Authorize]
public class UserController : ControllerBase
{
    public async Task<IActionResult> Get()
    {
        // TODO: write code
        return Ok();
    }
}
```
Sample Code

github.com/1kevgriff/sample-spa-vuejs-aspnetcore
Thanks for joining me!

Kevin Griffin
Owner, Swift Kick

twitter.com/1kevgriff
kevin@swiftkick.in
kevgriffin.com
swiftkick.in

© 2019 Swift Kick