ASP.NET Core Vue template with custom configuration using CLI 3.0

Libor Pansky
Aug 25, 2018 · 5 min read

> Do you know everything about .NET and Vue CLI 3.0? 😎

Just jump to template creation process or clone git repository to get started.

Preface

Lately I become a big fan of Vue.js and coming from a ASP.NET background, I have to complain about poor quality of templates available for .NET Core. I'm sorry that Microsoft discontinued support for official Vue template and that the only ones available for a new dotnet projects are React and Angular. There are some good open source templates out there, but either outdated version of Webpack and plugins or combination of libraries I don't use make a new project creation time consuming. Fortunately, Vue team was working hard over the past few months and now a stable version of its CLI 3.0 has been released!

> Vue CLI 3.0?? What does it mean for us, developers?

New CLI aims to minimize the amount of configuration we have to go through to set-up a new project. Even if I find myself very important to understand Webpack configuration and all the magic going behind the scenes, I understand for beginners can be quite complicated and hard to set up.

Vue CLI 3.0 is based on Webpack 4 and offers out-of-the-box support for:

- Hot module replacement
- Code-splitting
- Tree-shaking

- ES2017 transpilation
- Long term caching and so on

You can choose optional integrations to fit your needs such as:

- TypeScript
- Progressive Web App
- Vue Router & Vuex
- Linting, unit testing, E2E testing
- 3rd party component frameworks (Vuetify, Vue Bootstrap etc.)
- publish your personal/enterprise plugin and so on

For a full feature list, I suggest you to read the official CLI release statement by Evan You.

Prerequisites

- .NET Core SDK `dotnet --version 2.1.402`
- Node.js >= 8.9 `node --version v8.11.4`
- Vue CLI >= 3.0 `npm install -g @vue/cli vue --version 3.0.3`
- Your favourite editor (I prefer VS Code)

. . .

Template

I will describe two ways of template creation: one for developers who want to start with no configuration and the other for people who want to setup their own template and choose custom integrations. I will describe the process using CLI as well as GUI, so everyone can choose their preferred method.

I. CLI Configuration

- A - **CLI Template creation**: choose custom features and setup a new project under 3 minutes [**RECOMMENDED**]

- B - **Clone Template**: clone git repository and run project with simple commands

II. GUI Configuration

- A - **GUI Template creation**: choose custom features and setup a new project under 3 minutes
- B - **Clone Template**: clone git repository and run project with simple commands

. . .

CLI Template creation

Create .NET Project

There's no ASP.NET Core Vue template, so we will create a project from React template and modify it to play well with Vue. To create a new .NET Core app just run the following command:

```
dotnet new react -o aspnet-core-vue-app
```

Open folder in your favourite editor, for VS Code owners:

```
code aspnet-core-vue-app/.
```

Modify Template

Change Startup.cs:

Change Vue production build output directory on line 28:

```
1  - configuration.RootPath = "ClientApp/build";
2  + configuration.RootPath = "ClientApp/dist";
```

ASP.NET Core Vue starter - Startup.cs root path hosted with ❤️ by GitHub view raw

Remove HTTPS redirect middleware for development on line 45 (otherwise it's necessary to configure ASP.NET Core SSL certificate for a Vue app, it's better to disable it).

```
1  if (!env.IsDevelopment())
2  {
```

```
3       app.UseHttpsRedirection();
4   }
```

ASP.NET Core Vue starter - Startup.cs HTTPS redirect middleware hosted with ❤ by GitHub view raw

> *Steve Sanderson from Microsoft wrote:*
>
> *As far as I'm aware, we don't have plans to introduce Vue-specific features. This isn't because we have anything against Vue, but rather just to limit the growth in the number of frameworks that we're maintaining support for. The dev team only has a finite capacity for handling third-party concepts, and last year we made the strategic choice to focus on only Angular and React.*

For this reason Vue CLI middleware is available as a separated NuGet package. Install the package:

```
dotnet add package VueCliMiddleware
```

Replace React middleware with Vue CLI on line 62:

```
1   + using VueCliMiddleware;
2
3   - spa.UseReactDevelopmentServer(npmScript: "start");
4   + // run npm process with client app
5   + spa.UseVueCli(npmScript: "serve", port: 8080);
6   + // if you just prefer to proxy requests from client app, use proxy to SPA dev server i
7   + // app should be already running before starting a .NET client:
8   + // spa.UseProxyToSpaDevelopmentServer("http://localhost:8080"); // your Vue app port
```

ASP.NET Core Vue starter - Startup.cs proxy to SPA middleware hosted with ❤ by GitHub view raw

Scaffold Vue application

Remove all the contents of the folder /ClientApp and create a new Vue project by using Vue CLI:

```
vue create client-app
```

> Unfortunately Vue CLI does not allow us to set a project name by C# standards using Upper Camel Case (Pascal Case) naming convention, so let's initiate app inside of

> client-app folder and then move the content to ClientApp.

Move all the contents from the new folder /client-app to /ClientApp.

Now the application is ready to run. 👌

Start the application

Start the application with `dotnet run` command.

Let's start our ASP.NET Core and Vue project

Clone template using CLI

- Clone starter repository `git clone https://github.com/SoftwareAteliers/asp-net-core-vue-starter`
- Start the application using `dotnet run`

> *It will take some time during the first run to download all client side dependencies.*

GUI Template creation

- Use Visual Studio 2017 and create a new ASP.NET Core Web Application using React.js template (or run `dotnet new react` if you don't have VS installed).

Create a new ASP.NET Core Web Application in Visual Studio 2017

- To modify template **follow the same procedure as before**.
- **Remove all the contents** of the folder /ClientApp and create a new Vue project by using Vue CLI 3.0 GUI running `vue ui` command.
- Follow the wizard that guides you through the project creation process. **Do not forget to create the app inside of the ClientApp folder.**

Create a new Vue application with Vue CLI 3.0 using a GUI

- Run the application in Visual Studio Code or Visual Studio 2017 by hitting `F5` and browse http://localhost:5000.

Publish to Production

To publish application in production configuration you can just run `dotnet publish` command and it will do following:

- Run `npm install` in /ClientApp folder
- Run `npm run build` in /ClientApp folder
- Publish ASP.NET Core app and copy /ClientApp/dist folder to publish output folder.

To make publish process working we have to modify .csproj file to change React publish output folder /build to Vue /dist. Just change line 41 from:

```
<DistFiles Include="$(SpaRoot)build\**" />
```

to:

```
<DistFiles Include="$(SpaRoot)dist\**" />
```

Clone template using GUI

Use ready to start template available at Software Ateliers GitHub repository.

- Download starter as a ZIP file or `git clone https://github.com/SoftwareAteliers/asp-net-core-vue-starter`
- Run the application in Visual Studio Code or Visual Studio 2017 by hitting `F5` and browse http://localhost:5000.

Further reading

- Docs for Vue CLI 3.0

What do you think about the new CLI? Am I missing something in this article? Let me know in the comments and I'll add it in!

Now let's create something awesome with ASP.NET Core and Vue!

Happy coding 👋

Aspnetcore Vue Template Get Started Development

About Help Legal