COS30043 Interface Design and Development

Lecture 10 – Vue CLI and Single Page Application

2022 – Semester 1

Contents

Problems of Global Components

- In many small to medium-sized Vue projects, global components will be defined. But in more complex projects, there are many disadvantages:
  - Global definitions force unique names for every component
  - String templates lack syntax highlighting
  - No CSS support
- Vue.js's single-file components solve all these problems

Single File Components

- Single file components have extension .vue
- Single-file components allow us to build an entire component (structure, style, and function) in one file, consist of 3 parts:
  - template
  - script
  - Style
- Most code editors provide syntax highlighting
- An application may include many .vue files. These .vue files can be compiled to .js files using build tools such as webpack.

Single File Components Example

```
<template>
        <div class="hellostyle">
                <h1>{{ msg }}</h1>
        </div>
</template>

<script>
export default {
        name: 'HelloWorld’,
        data: function (){   return {msg: 'Hello World’}   }
}
</script>

<style>
        .hellostyle {background-color:pink;}
</style>
```

Single File Components Example

```
<template>
        <HelloWorld  />
</template>

<script>
        import HelloWorld from './components/HelloWorld.vue’

        export default {
                name: 'App',
                components: {   HelloWorld  }
        }
</script>

<style>
        ...
</style>
```

Contents

Vue CLI

- Vue CLI is a command-line utility that allows you to choose from a range of build tools, which it will then install and configure for you.
- It will also scaffold out your project, providing you with a pre-configured starting point that you can build on, rather than starting everything from scratch.

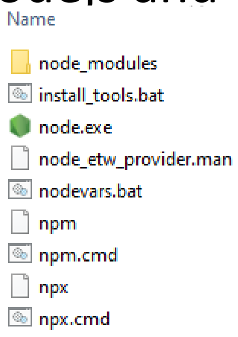
Vue CLI Installation

- Vue CLI installation needs node.js
- Node.js is a cross-platform JavaScript runtime environment for servers and applications.

  Go to https://nodejs.org/en/download/

  Download and install node.js

  Latest LTS Version: 16.15.0 (includes npm 8.5.5)
- Let's suppose we install node.js on C:/nodejs. From the file explore, you will see a new folder nodejs and the corresponding files.



Vue CLI Installation

- Now let's install Vue CLI. We will use the npm command from node.js to install Vue CLI.
- npm is a software Package Manager and Installer.

Open command line and type the following commands to install Vue CLI

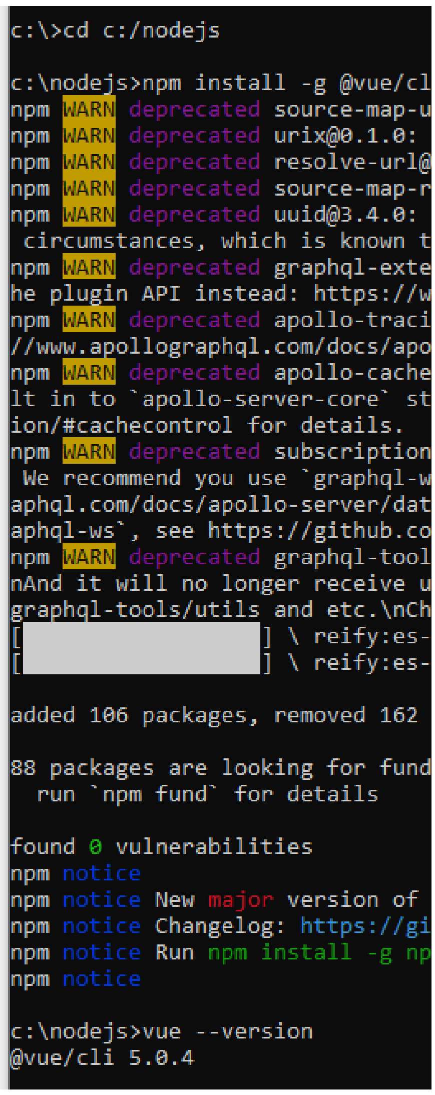
cd c:/nodejs

npm install -g @vue/cli

When the installation is finished, you can check the version

vue –version

If you can see the version number, you installation is successful.

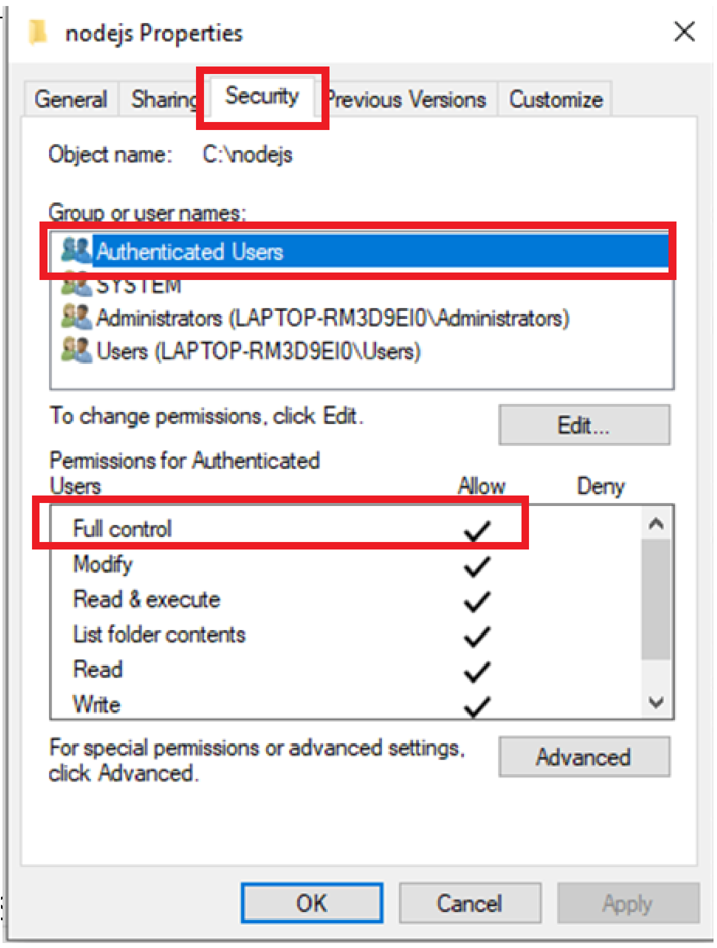


```
c:\>cd c:/nodejs

c:\nodejs>npm install -g @vue/cl
npm WARN deprecated source-map-u
npm WARN deprecated urix@0.1.0:
npm WARN deprecated resolve-url@
npm WARN deprecated source-map-r
npm WARN deprecated uuid@3.4.0:
 circumstances, which is known t
npm WARN deprecated graphql-exte
he plugin API instead: https://w
npm WARN deprecated apollo-traci
//www.apollographql.com/docs/apo
npm WARN deprecated apollo-cache
lt in to `apollo-server-core` st
ion/#cachecontrol for details.
npm WARN deprecated subscription
 We recommend you use `graphql-w
aphql.com/docs/apollo-server/dat
aphql-ws`, see https://github.co
npm WARN deprecated graphql-tool
nAnd it will no longer receive u
graphql-tools/utils and etc.\nCh
[                  ] \ reify:es-
[                  ] \ reify:es-

added 106 packages, removed 162

88 packages are looking for fund
  run `npm fund` for details

found 0 vulnerabilities
npm notice
npm notice New major version of
npm notice Changelog: https://gi
npm notice Run npm install -g np
npm notice

c:\nodejs>vue --version
@vue/cli 5.0.4
```

Vue CLI Installation

Grant Vue permission to create project

- In file explorer, right click the nodejs folder, choose Properties
- Select the Security tab, give Authenticated Users Full control.



nodejs Properties

General | Sharing | Security | Previous Versions | Customize

Object name: C:\nodejs

Group or user names:

- Authenticated Users
- SYSTEM
- Administrators (LAPTOP-RM3D9EI0\Administrators)
- Users (LAPTOP-RM3D9EI0\Users)

To change permissions, click Edit. Edit...

Permissions for Authenticated Users	Allow	Deny
Full control	✓	
Modify	✓	
Read & execute	✓	
List folder contents	✓	
Read	✓	
Write	✓	

For special permissions or advanced settings, click Advanced. Advanced

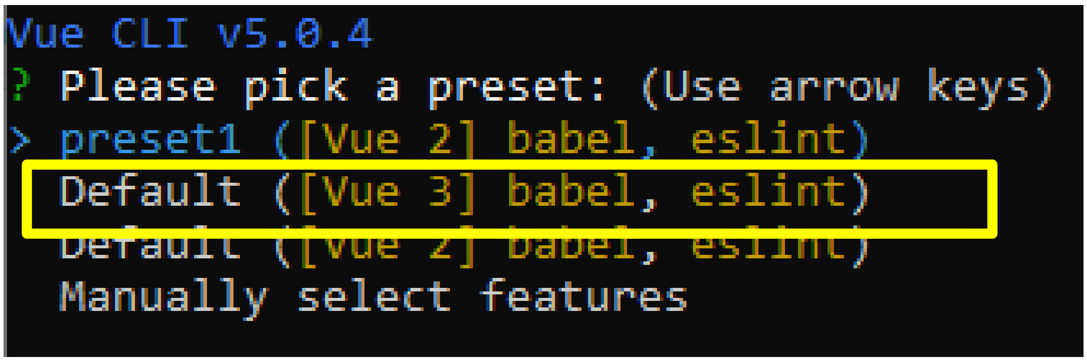
OK Cancel Apply

Create a Project

- To create a new project, use the “vue create” command. The command below will create a project named hello-world.

  vue create hello-world

  You will be asked to choose options, use the up or down arrow to choose an option. We will choose Vue 3 now.



```
Vue CLI v5.0.4
? Please pick a preset: (Use arrow keys)
> preset1 ([Vue 2] babel, eslint)
  Default ([Vue 3] babel, eslint)
  Default ([Vue 2] babel, eslint)
  Manually select features
```

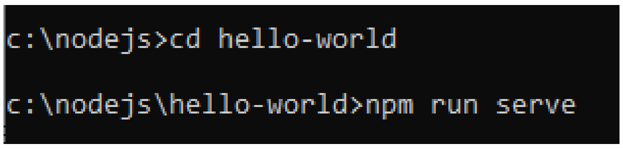
Create a Project

This will take a few minutes. When the project is created, you can run it by the following command.

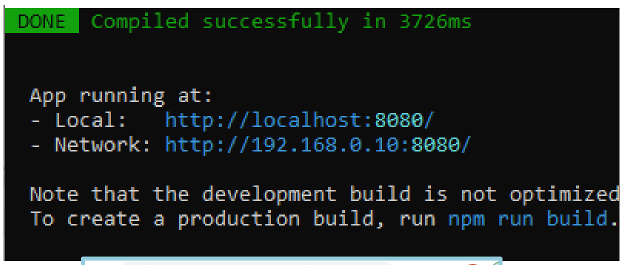
cd hello-world

npm run serve

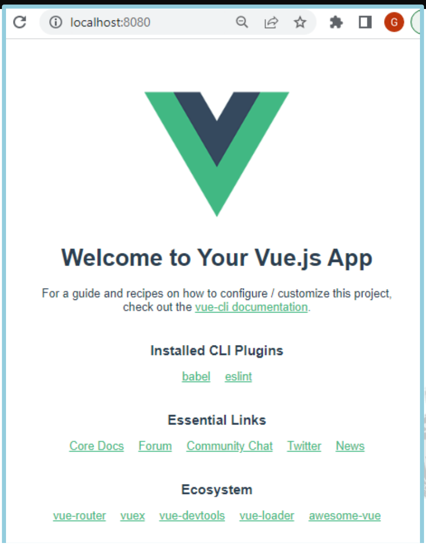
If successful, you will see this screen →



```
c:\nodejs>cd hello-world

c:\nodejs\hello-world>npm run serve
```



```
 DONE  Compiled successfully in 3726ms


  App running at:
  - Local:   http://localhost:8080/
  - Network: http://192.168.0.10:8080/

  Note that the development build is not optimized
  To create a production build, run npm run build.
```



localhost:8080

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

babel eslint

Essential Links

Core Docs Forum Community Chat Twitter News

Ecosystem

vue-router vuex vue-devtools vue-loader awesome-vue

Follow the instruction to open your browser and type the url http://localhost:8080/ you will see your app running.

To stop the serve, use ctrl+c

ne

Build the Project

By default, Vue CLI assumes your app will be deployed at the root of a domain, e.g. https://www.my-app.com/. If your app is deployed at a sub-path, you will need to specify that sub-path using publicPath in vue.config.js. The publicPath value can be set to an empty string ('') so that all assets are linked using relative paths.

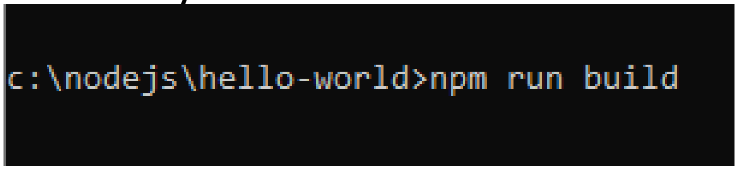
To build the project for distribution, add the following code

```
module.exports = {
        publicPath: ''
};
```

to “vue.config.js” under the hello-world folder.

Run the following command. Make sure you are in the hello-world directory

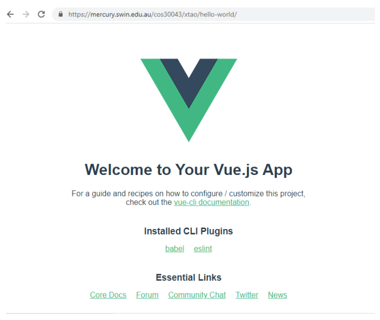
npm run build



```
c:\nodejs\hello-world>npm run build
```

Build the Project

- When the build is complete, from file explore go to the project directory, you will see a new folder named “dist”.
- Now you can deploy this folder to a web server. For example, you can upload this folder to Mercury under the www/htdocs, and rename “dist” to “hello-world”, go to a browser and type the corresponding url, you will see your app running



https://mercury.swin.edu.au/cos30043/xtao/hello-world/

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

babel eslint

Essential Links

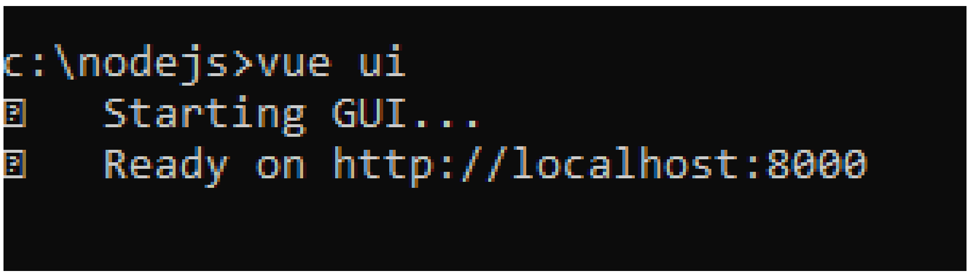
Core Docs Forum Community Chat Twitter News

- For example, your mercury
  URL could be
  https://mercury.swin.edu.au/cos30043/s1234567/hello-world

Create a Project Using GUI

You can also create and manage projects using a graphical interface with the vue ui command:

vue ui



```
c:\nodejs>vue ui
    Starting GUI...
    Ready on http://localhost:8000
```

Open a browser, type http://localhost:8000, then create your project using the GUI

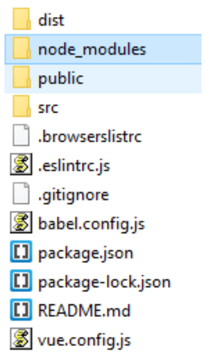
Contents

- Single File Components
- Vue CLI and Installation
- SPA Created Using Vue CLI
- Continue with Our Development
- Using Bootstrap

Folder Structure

This is the default folder structure

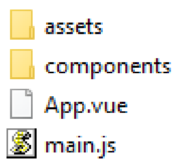
- node_modules: This is the folder where all the dependencies and libraries which are required to build Vue are stored.
- public: In the public directory, you'll usually put files that you don't want to process through webpack. For example images.
- src: This is the folder where the application source code of your Vue Application resides.



dist
node_modules
public
src
.browserslistrc
.eslintrc.js
.gitignore
babel.config.js
package.json
package-lock.json
README.md
vue.config.js

Under the src folder:

- assets: In this folder, you'll store all the assets of your application, which includes fonts, images, etc.
- components: This folder should contain all the components or the building blocks of your application.
- App.vue: App.vue file is the root component, all other components are nested within this component.
- main.js: This is the file that renders our app and mounts it to the DOM.



assets
components
App.vue
main.js

src/main.js

```
import { createApp } from 'vue’
import App from './App.vue’
createApp(App).mount('#app’)
```

The main.js is the entry point of our application.

At the top of this file, it imports all the necessary dependencies. First is Vue, the Vue core library. Second is App.vue, which is the root component of this project.

We create the Vue instance and mount it on the element with an id app.

Public/index.html

In index.html file, you will find that there is a div with an id of app. This is where our App component is mounted.

```
<div id="app"></div>
<!-- built files will be auto injected -->
```

After we build our app, if you look into index.html file inside the dist directory, you’ll find the bundled js file appended inside <head>.

```
<script defer="defer" src="/js/chunk-vendors.9fdfbc01.js"></script>
<script defer="defer" src="/js/app.f1c6d140.js"></script>
```

The chunk-vendors file contains all the dependencies and plugins code, and the app file contains the Application js code.

src/App.vue

```
<template>
        <img alt="Vue logo" src="./assets/logo.png" />
        <HelloWorld msg="Welcome to Your Vue.js App" />
</template>
<script>
    import HelloWorld from "./components/HelloWorld.vue";
    export default {
        name: "App",
        components: {   HelloWorld  },
    };
</script>
   …
```

App is the root component of this project.

src/components/HelloWorld.vue

```
<template>
    <div class="hello">
        <h1>{{ msg }}</h1>
        <!--   other information  -->
    </div>
</template>
<script>
    export default {
        name: "HelloWorld",
        props: {   msg: String,  },
    };
</script>
…
```

Contents

Create a New Component

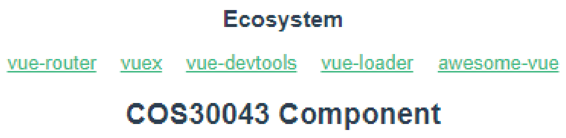
- Use HelloWorld.vue as an example, create a new component, cos30043.vue for example

```
<template>
  <div >
      <h2>COS30043 Component</h2>
     <p>This is the cos30043 unit, we are learning VueJS, Bootstrap.</p>
  </div>
</template>
<script>
  export default {
      name: "cos30043“
  }
</script>
… …
```

Create a New Component

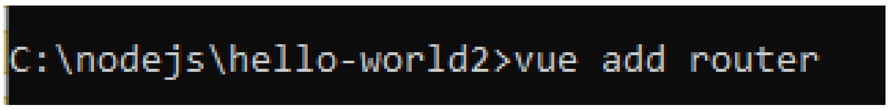
Import and use this new component in App.vue

```
<template>
    <div id="app">
        <img alt="Vue logo" src="./assets/logo.png" />
        <HelloWorld msg="Welcome to Your Vue.js App" />
        <cos30043   />
    </div>
</template>
<script>
  import HelloWorld from "./components/HelloWorld.vue";
  import cos30043 from "./components/cos30043.vue";
  export default {
      name: "App",
      components: {   HelloWorld,   cos30043  },
  };
</script>
```



Use Router

- We can add router to our project by using “vue add router”.
- But be careful when you run this command, because it will overwrite your App.vue file, so either plan to add vue-router before you start to work on the project, or backup your App.vue file before you add it.
- In order to show what are the changes made by “vue add router”, here we created a new project, hello-world2, and add router to it. (If you need router in your project, add router first.)



```
C:\nodejs\hello-world2>vue add router
```

Use Router – main.js

In main.js, it imports router from the /router directory, and adds it to the main Vue instance for your application, so you can use it at the top level of your app.

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

createApp(App).use(router).mount('#app')
```

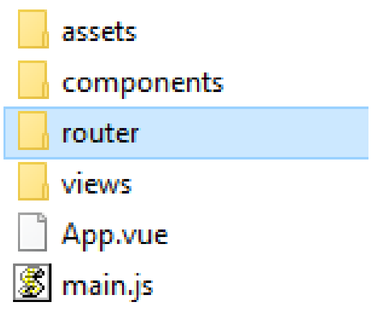
Use Router – App.vue

In the changes to the App.vue file that “vue add router” makes, they add router-view and router-link:

```
<template>
        <nav>
            <router-link to="/">Home</router-link> |
            <router-link to="/about">About</router-link>
        </nav>
         <router-view/>
</template>
```

Use Router – router/index.js

- It also adds folders called router and views.
- In the views folder, it generated some basic example .vue files, About.vue and Home.vue.
- The router folder has only one file, index.js. The file creates a new VueRouter and exporting it to be used in the rest of the app.
- We will define all the routes we want in our app in the routes list in this file.



Use Router – router/index.js

```
import { createRouter, createWebHistory } from 'vue-router'
import HomeView from '../views/HomeView.vue'
import AboutView from '../views/AboutView.vue’

const routes = [
 {    path: '/’,    name: 'home’,    component: HomeView   },
 {    path: '/about’,     name: 'about’,    component: AboutView   }
]
const router = createRouter({
 history: createWebHistory(process.env.BASE_URL),
 routes
})
export default router
```

Use Router – add a new menu item

If you need to add a new menu "product" for example, you need to:

- Create the product.vue file
- Add a route in router/index.js

```
import ProductView from '../views/ProductView.vue'
… …
const routes = [      …
   {   path: '/product',
       name: 'product',
       component: ProductView },
   …
   ]
```

- In App.vue

```
<div id="nav">
   …
   <router-link to="/product">Product</router-link>
</div>
```

Contents

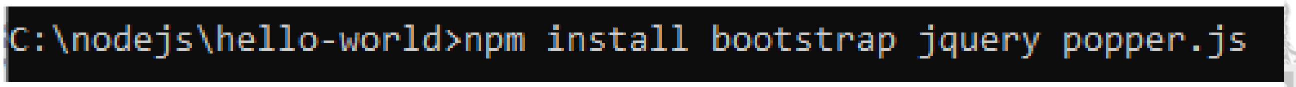
Install Bootstrap

To install Bootstrap on your project, you can run

npm install bootstrap jquery popper.js

This will install the latest stable version of Bootstrap library and will also install the required dependencies of Bootstrap i.e. jquery and popper.js into the project.

if you only need the bootstrap styles and don't need the javascript provided by bootstrap then you can leave out jquery and popper.js dependencies.



```
C:\nodejs\hello-world>npm install bootstrap jquery popper.js
```

Import Bootstrap

Go to src/main.js and add the following lines.

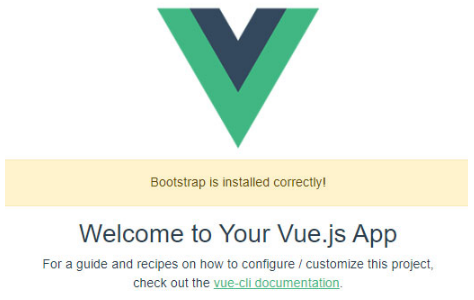
```
import 'bootstrap'
import 'bootstrap/dist/css/bootstrap.min.css'
```

If you only use the styles you can leave out the import bootstrap line and only import the minified CSS.

You can add a simple alert to HelloWorld.vue component under src/components/ to test if bootstrap is installed correctly.

```
<p class="alert alert-warning">Bootstrap is installed correctly!</p>
```

If it works, you will see the alert in "warning" background color.



Bootstrap is installed correctly!

Welcome to Your Vue.js App

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

WHAT'S NEXT?
- PORTFOLIO PREPARATION AND UNIT OVERVIEW