# Vue+vuetify开发搭建流程

## 软件需求

#### 编辑工具

* + 1. Vscode

#### 仓库管理工具

* + 1. NodeJs

提供服务端JS运行环境

* + 1. NPM

包管理工具

#### vue

* + 1. vue2.X

现在最新的Vue发布版本

* + 1. vueX

Vuex状态管理用来管理，组件之间通信的一个插件

* + 1. vue-cli

Vue脚手架，快速搭建项目

* + 1. axios

基于promise对ajax封装进行网络请求，实现前后端数据交互

* + 1. vue-router

Vue Router 是 Vue.js 官方的路由管理器。它和 Vue.js 的核心深度集成，让构建单页面应用变得易如反掌。包含的功能有：嵌套的路由/视图表；模块化的、基于组件的路由配置；路由参数、查询、通配符；基于 Vue.js 过渡系统的视图过渡效果；细粒度的导航控制；带有自动激活的 CSS class 的链接；HTML5 历史模式或 hash 模式，在 IE9 中自动降级；自定义的滚动条行为。

#### vuetify

Vue前端图形框架

#### Promsj

暂未使用

#### mobile

暂未使用

#### eslint

代码格式校验，提高代码规范性

#### sass

Sass 是一款强化 CSS 的辅助工具，它在 CSS 语法的基础上增加了变量 (variables)、嵌套 (nested rules)、混合 (mixins)、导入 (inline imports) 等高级功能，这些拓展令 CSS 更加强大与优雅。使用 Sass 以及 Sass 的样式库（如 [Compass](http://compass-style.org/)）有助于更好地组织管理样式文件，以及更高效地开发项目。

* 1. **vue-next**

Vue下一代版本V3.0

## 步骤

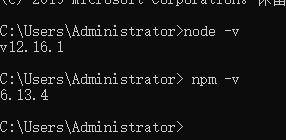
#### 安装NodeJS

* + 1. 下载地址<https://nodejs.org/en/download/>
    2. 根据自己的机器配置及其开发要求下载对应版本，这里使用node版本 12.16.1，npm版本6.13.4
    3. 安装

安装的过程中一路next，无须修改默认勾选项

* + 1. 检查安装是否成功

使用node -v 和npm -v，出现版本号则正常



#### 安装VS Code

* + 1. 下载地址<https://code.visualstudio.com/>

#### 配置环境变量

非必须，但是如果node安装在C，建议把仓库和缓存配置到其他盘，避免占用C盘空间

* + 1. 配置仓库

①到nodejs安装目录，创建两个文件夹 node\_gobal

②输入命令npm config set prefix "D:\Program Files\nodejs\node\_gobal"，实际地址根据自己电脑配置

* + 1. 配置缓存

①到nodejs安装目录，创建两个文件夹 node\_cache

②输入命令npm config set cache "D:\Program Files\nodejs\node\_cache"，实际地址根据自己电脑配置

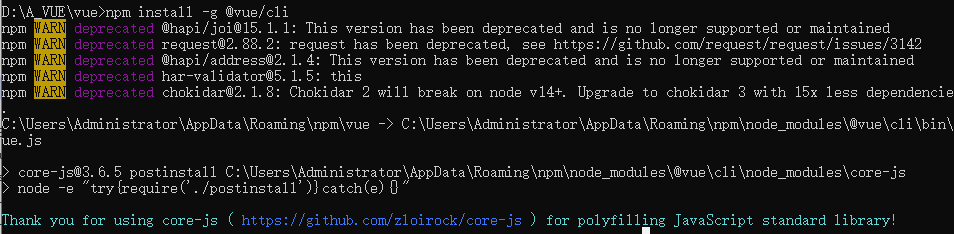
#### 安装Vue-cli

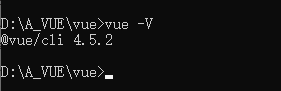
2.4.0 设置淘宝镜像，解决下载速度慢的问题（非必须）

npm config set registry https://registry.npm.taobao.org

* + 1. 安装vue-cli（安装目前最新脚手架版本4.5.2）

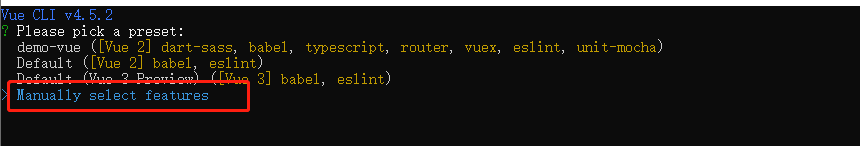
①在目标文件夹下打开终端，输入npm install -g @vue/cli





②创建项目vue create hello-world

选择最后一项，手动配置

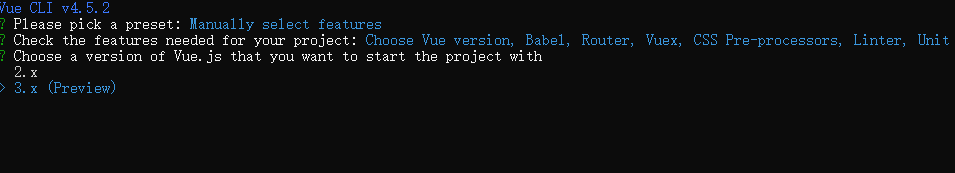


上下移动，空格选择

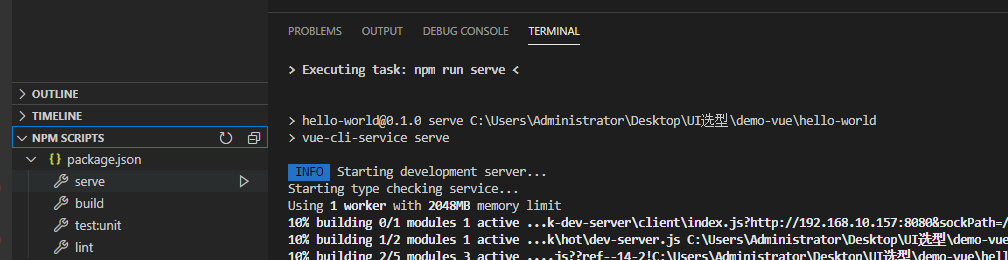


选择vue版本这里选择3.X

2.x与3.x区别 <https://www.e-learn.cn/content/qita/2362925>



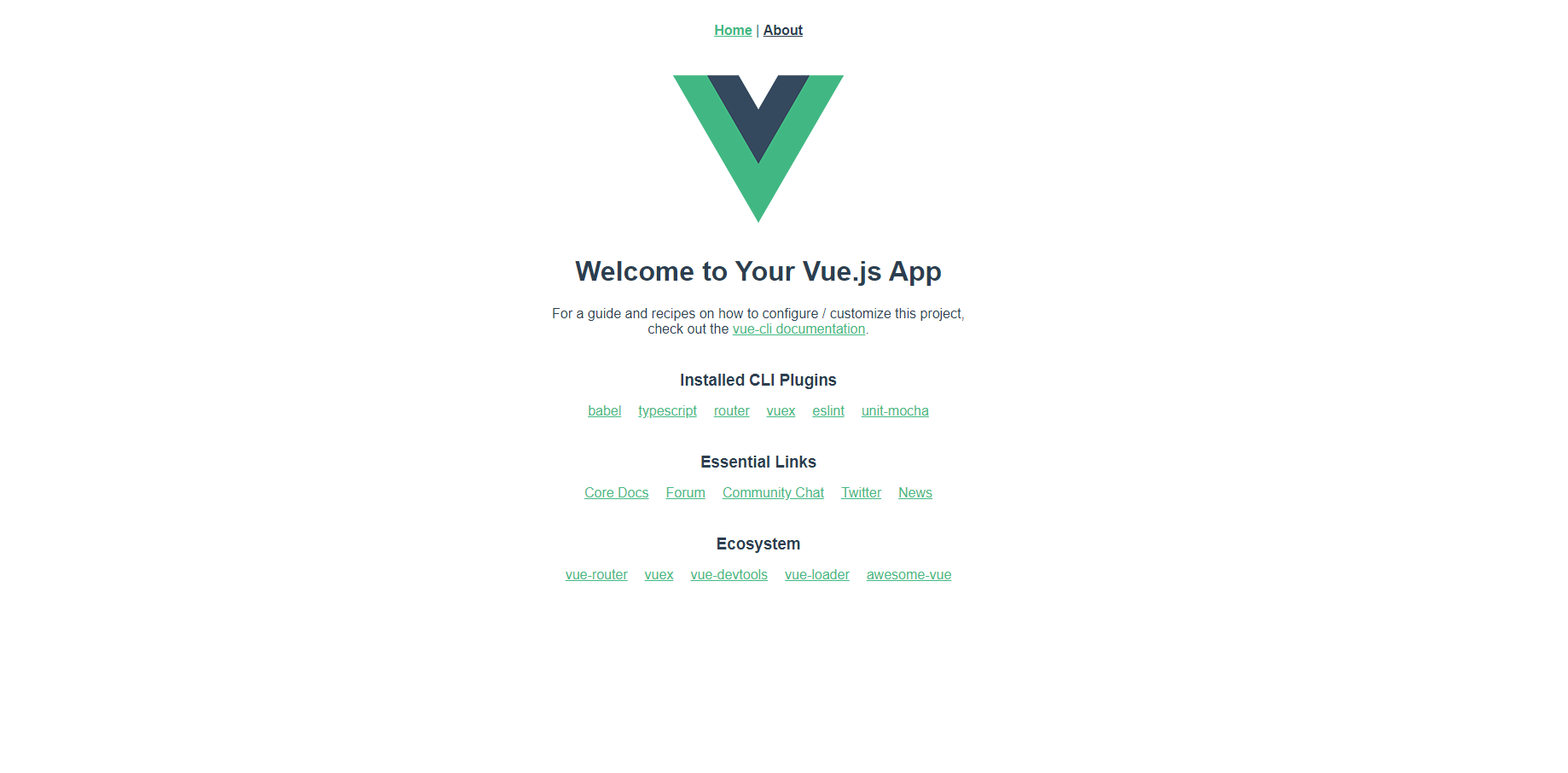
③将项目导入vscode并运行



或者在vscode中断中键入rpm run server

④查看是否成功

启动无报错，默认端口8080，在浏览器输入[http://localhost:8080](http://localhost:8080/" \l "/)



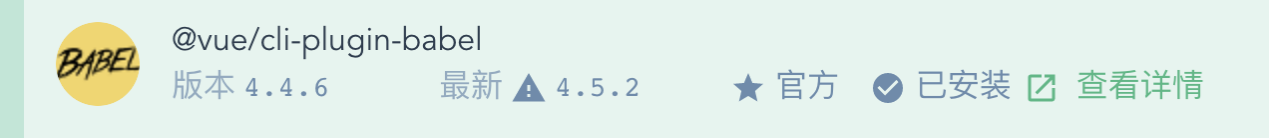
#### 安装插件

* + 1. 安装BABEL

Vue项目中普遍使用es6语法，但有时我们的项目需要兼容低版本浏览器，这时就需要引入babel插件，将es6转成es5。

方式一：脚手架创建项目时已经选择了安装（V4.5.0）；

方式二：也可通过Vue UI界面独立安装：

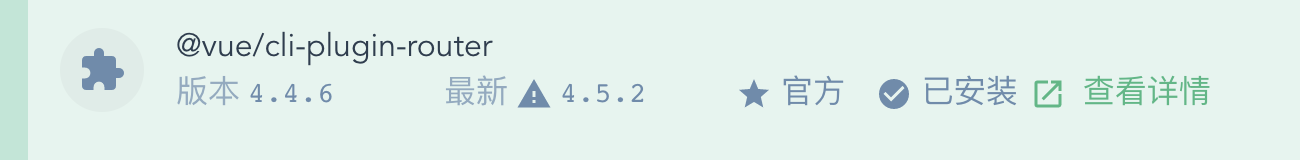


* + 1. 安装Vue-Router

Vue官方提供的路由插件

方式一：脚手架创建项目时已经选择了安装（V4.5.0）；

方式二：也可通过Vue UI界面独立安装：

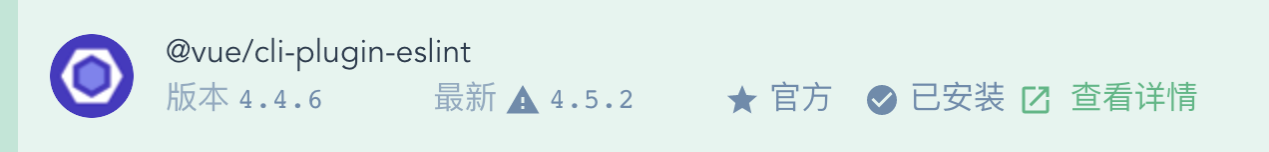


* + 1. 安装eslint

代码风格检查插件

方式一：脚手架创建项目时已经选择了安装（V4.5.0）；

方式二：也可通过Vue UI界面独立安装：



* + 1. 安装vuex

管理组件之间通信插件

方式一：脚手架创建项目时已经选择了安装（V4.5.0）；

方式二：也可通过Vue UI界面独立安装：



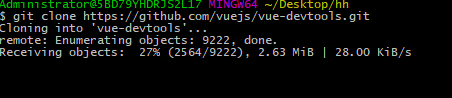
* + 1. 安装vue-devtools

Vue可视化调试工具

方式一：可以用翻墙工具到谷歌浏览器插件里面安装。

方式二（V4.1.4）：

①从git获取项目，地址https://github.com/vuejs/vue-devtools.git



②安装项目所需的npm包 npm install

③编译项目 npm run build

④添加至Chrome浏览器



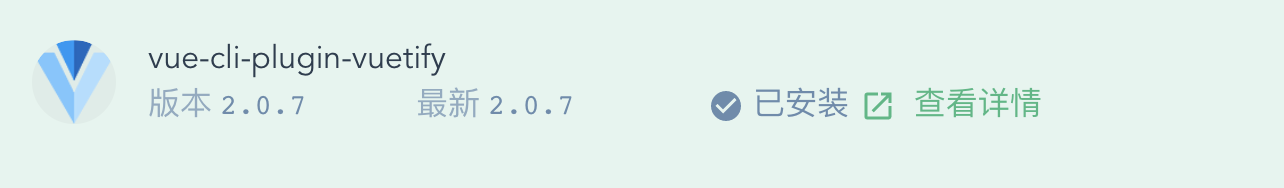
* + 1. 安装vuetify

Vue前端图形框架

方式一（V2.3.8）：

npm install vuetify --save/npm install vuetify -S 需用于生产环境

方式二：也可通过Vue UI界面安装：

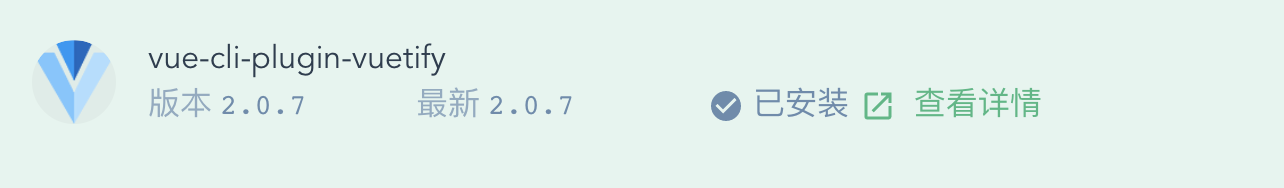


* + 1. 安装vue-cli-plugin-vuetify

方式一（V2.0.7）：

npm i vue-cli-plugin-vuetify

也可通过Vue UI界面安装：



* + 1. 安装vuex（vuetify）

Vuex（vuetify）状态管理，用来管理组件之间通信的一个插件

方式一（V0.5.4）：npm install @vuetify/vuex-cognito-module

* + 1. vuetify-loader

用于按需加载，无需单独安装，安装Vuetify后便可使用

* + 1. 安装eslint-plugin-vuetify

Vuetify的eslint插件，校验代码格式

方式一（V1.0.0-beta.7）：npm i eslint-plugin-vuetify

* + 1. 安装eslint-config-vuetify

Vuetify的自定义eslint配置，校验代码格式

方式一（V0.6.1）：npm install eslint-config-vuetify

* + 1. 安装Chats

图形化界面

①High charts（V1.3.5）

<https://github.com/highcharts/highcharts-vue>

Highcharts Vue 基于 Vue 框架封装的 Highcharts，可以很方便的在 Vue 开发环境中使用 Highcharts 创建交互性图表，

npm install highcharts-vue

②E-Charts（V4.8.0）

<https://github.com/ecomfe/vue-echarts/blob/master/README.zh_CN.md>

vue-echarts是封装后的vue插件， 基于 [ECharts](http://echarts.baidu.com/index.html) v4.0.1+ 开发，依赖 [Vue.js](https://vuejs.org/) v2.2.6+

npm install echarts vue-echarts

③amchart（V4.9.34）

<https://www.amcharts.com/docs/v4/getting-started/integrations/using-vue-js/>

amCharts可以轻松实现所有图表类型，支持React，Angular，Vue，Ember等

npm install @amcharts/amcharts4

④v-chart（V5.0.0-beta.0）

[https://v-charts.js.org/#/](https://v-charts.js.org/" \l "/)

在使用 echarts 生成图表时，经常需要做繁琐的数据类型转化、修改复杂的配置项，v-charts 的出现正是为了解决这个痛点。基于 Vue2.0 和 echarts 封装的 v-charts 图表组件，只需要统一提供一种对前后端都友好的数据格式设置简单的配置项，便可轻松生成常见的图表

npm install v-echarts echarts

#### 插件使用

* + 1. 使用BABEL

根目录下的babel.config.json或者babel.config.js，具体是什么格式在创建的时候可以自主选择

官网配置说明[https://babeljs.io/docs/en/plugins/#plugin-preset-options](https://babeljs.io/docs/en/plugins/" \l "plugin-preset-options)

Json

{  
 "presets": [...],  
 "plugins": [...]  
}

Js

module.exports = function (api) {  
 api.cache(true);  
 const presets = [ ... ];  
 const plugins = [ ... ];  
 return {  
 presets, plugins  
 };  
}

* + 1. 使用Vue-Router

在src/router/index.js里进行配置，例如

import Vue from 'vue' //引入Vue  
import Router from 'vue-router' //引入vue-router  
import Hello from '@/components/Hello' //引入根目录下的Hello.vue组件  
  
Vue.use(Router) //Vue全局使用Router  
  
export default new Router({  
 routes: [ //配置路由，这里是个数组  
 { //每一个链接都是一个对象  
 path: '/', //链接路径  
 name: 'Hello', //路由名称，  
 component: Hello //对应的组件模板  
 }，{  
 path:'/hi',  
 component:Hi,  
 children:[ //子路由,嵌套路由 （此处偷个懒，免得单独再列一点）  
 {path:'/',component:Hi},  
 {path:'hi1',component:Hi1},  
 {path:'hi2',component:Hi2},  
 ]  
}  
]  
})

Vue官方路由配置文档说明<https://router.vuejs.org/>

* + 1. 使用Eslint

在package.json内的"eslintConfig"或者在.eslintrc.js文件配置（具体是哪个文件取决于安装eslint时选择将配置与package.json合并还是独立的配置文件）

例如：

package.json

"eslintConfig": {  
 "root": true,////此项是用来告诉eslint找当前配置文件不能往父级查找  
 "env": {  
 "node": true//此项指定环境的全局变量，下面的配置指定为node环境  
 },  
 "extends": [// 此项是用来配置vue.js风格，就是说写代码的时候要规范的写，如果你使用vs-code我觉得应该可以避免出错  
 "plugin:vue/essential",  
 "@vue/standard"  
 ],  
 "rules": {//规则配置写在这里  
 "indent": [1, 4]  
 },  
 "parserOptions": {  
 "parser": "babel-eslint"//此项是用来指定eslint解析器的，解析器必须符合规则，babel-eslint解析器是对babel解析器的包装使其与ESLint解析  
 }  
},

eslintrc.js

module.exports = {  
 root: true,  
 //此项是用来指定javaScript语言类型和风格，sourceType用来指定js导入的方式，默认是script，此处设置为module，指某块导入方式  
 "parserOptions": {  
 "parser": "babel-eslint",  
 "ecmaVersion": 2017,  
 "sourceType": "module"  
 },  
 globals: {  
 "$": true  
 },  
 //此项指定环境的全局变量，下面的配置指定为浏览器环境  
 env: {  
 browser: true,  
 node: true,  
 es6: true,  
 },  
 // https://github.com/feross/standard/blob/master/RULES.md#javascript-standard-style  
 // 此项是用来配置标准的js风格，就是说写代码的时候要规范的写，如果你使用vs-code我觉得应该可以避免出错  
 extends: [  
 'standard',  
 'plugin:vue/essential'  
 ],  
 // required to lint \*.vue files  
 // 此项是用来提供插件的，插件名称省略了eslint-plugin-，下面这个配置是用来规范html的  
 plugins: [  
 "vue"  
 ],  
 // add your custom rules here  
 // 下面这些rules是用来设置从插件来的规范代码的规则，使用必须去掉前缀eslint-plugin-  
 // 主要有如下的设置规则，可以设置字符串也可以设置数字，两者效果一致  
 // "off" -> 0 关闭规则  
 // "warn" -> 1 开启警告规则  
 //"error" -> 2 开启错误规则  
 // 了解了上面这些，下面这些代码相信也看的明白了  
 'rules': {  
  
 // 缩进  
 "indent": 0,  
 // 要求或禁止使用分号而不是 ASI（这个才是控制行尾部分号的，）  
 "no-mixed-spaces-and-tabs": [1, "smart-tabs"],  
 "semi": 0,  
 "no-tabs": 0,  
 "space-before-function-paren": [0, "always"],  
 // 强制在注释中 // 或 /\* 使用一致的空格  
 "spaced-comment": 0,  
 // always-multiline：多行模式必须带逗号，单行模式不能带逗号  
 "comma-dangle": [1, "never"],  
 // 强制使用一致的反勾号、双引号或单引号  
 "quotes": 0,  
 // allow paren-less arrow functions  
 'arrow-parens': 0,  
 // allow async-await  
 // 'generator-star-spacing': 0,  
 // allow debugger during development  
 'no-debugger': process.env.NODE\_ENV === 'production' ? 2 : 0  
 }  
}

Eslint在github上的文档

[https://github.com/vuejs/vue-cli/tree/dev/packages/%40vue/cli-plugin-eslint](https://github.com/vuejs/vue-cli/tree/dev/packages/@vue/cli-plugin-eslint)

* + 1. 使用Vuex

内容概念较多，参考官方文档

官方使用介绍<https://vuex.vuejs.org/zh/>

* + 1. 使用Vue-devtools

git clone https://github.com/vuejs/vue-devtools.git  
npm install  
npm run build

完成在谷歌浏览器开发者模式下插件下添加该文件夹开启后，F12即可使用

* + 1. 使用Vuetify

内容较多请参考官方文档

<https://vuetifyjs.com/zh-Hans/getting-started/quick-start/>

* + 1. 使用vue-cli-plugin-vuetify

var vueCliPluginVuetify = require("vue-cli-plugin-vuetify")

插件官网地址<https://www.npmjs.com/package/vue-cli-plugin-vuetify>

* + 1. 使用vuex（vuetify）

修改src / main.js

import attachCognitoModule from '@vuetify/vuex-cognito-module'

src下新建src/store.js，并添加修改

import store from './store'  
  
attachCognitoModule(store, {  
 userPoolId: 'your-data-here',  
 identityPoolId: 'your-data-here', // This field is now optional  
 userPoolWebClientId: 'your-data-here',  
 region: 'your-data-here'  
}, 'cognito')

GitHub参考样例<https://github.com/vuetifyjs/vuex-cognito-example>

官方参考文档[https://cognito.vuetifyjs.com/guide/install.html#vue](https://cognito.vuetifyjs.com/guide/install.html" \l "vue)

* + 1. 使用vuetify-loader

在src/plugins/vuetify.js中

import Vue from 'vue'  
import Vuetify from 'vuetify/lib'  
  
Vue.use(Vuetify)  
  
const opts = {}  
  
export default new Vuetify(opts)

在vue.config.js

const VuetifyLoaderPlugin = require('vuetify-loader/lib/plugin')  
  
module.exports = {  
 configureWebpack: {  
 plugins: [  
 new VuetifyLoaderPlugin()  
 ],  
 },  
}

官方参考文档<https://vuetifyjs.com/zh-Hans/customization/a-la-carte/>

* + 1. 使用eslint-plugin-vuetify

在.eslintrc.js中添加/修改

module.exports = {  
 extends: [  
 'plugin:vue/base'  
 ],  
 plugins: [  
 'vuetify'  
 ],  
 rules: {  
 'vuetify/no-deprecated-classes': 'error'  
 }  
}

参考文档<https://www.npmjs.com/package/eslint-plugin-vuetify>

* + 1. 使用eslint-config-vuetify

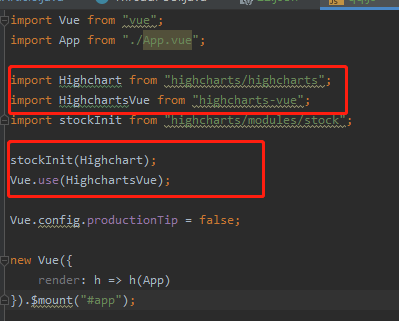
在.eslintrc.js中添加/修改

{  
 "extends": "vuetify"  
}

参考文档<https://github.com/vuetifyjs/eslint-config-vuetify>

* + 1. 使用High charts（V1.3.5）

在main.js中引入并注册

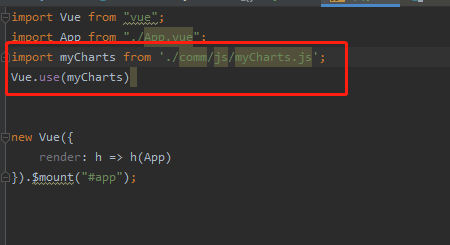


具体使用细节考官方文档

<https://www.highcharts.com.cn/docs/highcharts-vue>

* + 1. 使用E-Charts（V4.8.0）

在main.js中引入并注册



具体使用细节考官方文档或者GItHub

<https://github.com/ecomfe/vue-echarts>

<https://echarts.apache.org/zh/tutorial.html>

* + 1. 使用amchart（V4.9.34）

用例

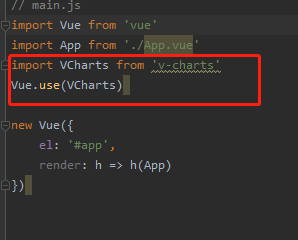
// Import stuff  
import \* as am4core from "@amcharts/amcharts4/core";  
import \* as am4charts from "@amcharts/amcharts4/charts";  
  
// Create chart instance  
let chart = am4core.create("chartdiv", am4charts.PieChart);  
  
// Create pie series  
let series = chart.series.push(new am4charts.PieSeries());  
series.dataFields.value = "litres";  
series.dataFields.category = "country";  
  
// Add data  
chart.data = [{  
 "country": "Lithuania",  
 "litres": 501.9  
}, {  
 "country": "Czech Republic",  
 "litres": 301.9  
}, {  
 "country": "Ireland",  
 "litres": 201.1  
}, {  
 "country": "Germany",  
 "litres": 165.8  
}, {  
 "country": "Australia",  
 "litres": 139.9  
}, {  
 "country": "Austria",  
 "litres": 128.3  
}, {  
 "country": "UK",  
 "litres": 99  
}, {  
 "country": "Belgium",  
 "litres": 60  
}, {  
 "country": "The Netherlands",  
 "litres": 50  
}];  
  
// And, for a good measure, let's add a legend  
chart.legend = new am4charts.Legend();

官方文档

<https://www.amcharts.com/docs/v4/getting-started/basics/>

* + 1. 使用v-chart（V5.0.0-beta.0）

在main.js中引入



具体使用细节参考官方文档

[https://v-charts.js.org/#/start](https://v-charts.js.org/" \l "/start)

#### VS Code对vue支持相关插件

<https://liubing.me/vscode-vue-setting.html>

* + 1. Vetur

1. 语法高亮：vue文件的语法高亮显示，除了支持template模板以外，还支持大多数 主流的前端开发脚本和插件，比如Sass和TypeScript
2. Snippet：可以使用一些snippet来编写不同的脚本，比如在script中申明 lang=”ts” 来开发TypeScript
3. Emment：VSCode本身自带了Emmet，能够通过Tab键对HTML5的代码进行快速开 发，不过，VSCode中需要修改Emmet配置才能对Vue进行支持
4. 错误检测：Vetur默认使用 eslint-plugin-vue@beta 来检测
   * 1. Snippets

自动补全，代码提示

* + 1. language-stylus

写stylus用的代码提示

* + 1. Auto Close Tag

自动闭合标签所用

* + 1. Auto Rename Tag

自动修改重命名配对的标签

* + 1. Bookmarks

可以对成片的代码做一些书签标记，方便后续查看

* + 1. Bracket Pair Colorizer

对括号进行着色，方便区分

* + 1. Copy Relative Path

用于复制文件的完整路径和相对路径，有时候我们可能需要复制一些文件的路径，该插件就很方便了

* + 1. Path Intellisense

路径自动感知，在配置文件中配置@后我们就可以很方便快捷的引用各种文件了

* + 1. Document This

主要用于方法的注释，选中方法名，按两次Ctrl+Alt+D,即可快速生成标准的注释

psioniq File Header按两次Ctrl+Alt+H可快速在文件的头部生成注释信息，如果对默认的注释模板不满意的话，可以在配置文件中自定义注释模板

* + 1. Vue Peek

用于Vue快速查看组件定义以及组件跳转，具体使用见插件文档地址中的使用方法。

* + 1. JavaScript (ES6) code snippets

用于快速生成ES6代码片段

* + 1. Material Icon Theme

Material风格的icon文件图标，可以看下安装前后的区别

* + 1. StandardJS – JavaScript Standard Style

代码规范

* + 1. **Beautify**

主要拿它来格式化html的，也可以格式话vue template中的html

* + 1. **vscode-element-helper**

用element-ui的，应该都知道这个插件，功能看图就知道了。

* + 1. **Version Lens**

显示npm，jspm，bower，dub和dotnet核心的软件包版本信息

* + 1. **One Dark Pro**

一款热门的主题