Vue脚手架

`Vue.js`

前置知识

导出和导入

使用`export`命令可以将一个JS对象导出，其他JS文件可以使用`import 对象名 from 路径`的形式导入这个对象来使用，但在导入时必须知道导出对象的名字。如果使用`export default`命令导出对象的话，其他JS文件在导入时就无需知道导出对象的名字，可以自定义对象名。

注意：export和import为ES6规范中的关键字，目前仅被少量JS运行平台实现，在脚手架里会被自动用babel转换为ES5对应语法。

组件中自定义标签名的大小写问题

在大多数浏览器解析HTML时，不区分标签的大小写。自定义标签中的大写字母会被自动转换为小写字母，如`<First></First>`会被转换为`<first></first>`标签，因此导致与注册时提供的标签名不一致，出现组件未正确注册的错误。

如果使用自定义组件时，代码写在template里，在解析时就会被当做JS解析，是区分大小写的。

在脚手架中使用时，代码一定会写在template里，推荐使用大驼峰命名方式。比如用户信息为`UserInfo`。

使用脚手架创建项目

安装NodeJS

脚手架的安装及运行均依赖NodeJS，可以在命令行键入`node -v`来查看是否安装NodeJS。

下载NodeJS

https://nodejs.org/en/

配置npm全局路径

```
npm config set prefix "D:\nodejs\node_global"
npm config set cache "D:\nodejs\node_cache"
```
之后将第一个路径同时配置到环境变量`path`中。

配置淘宝npm镜像

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

安装vue-cli

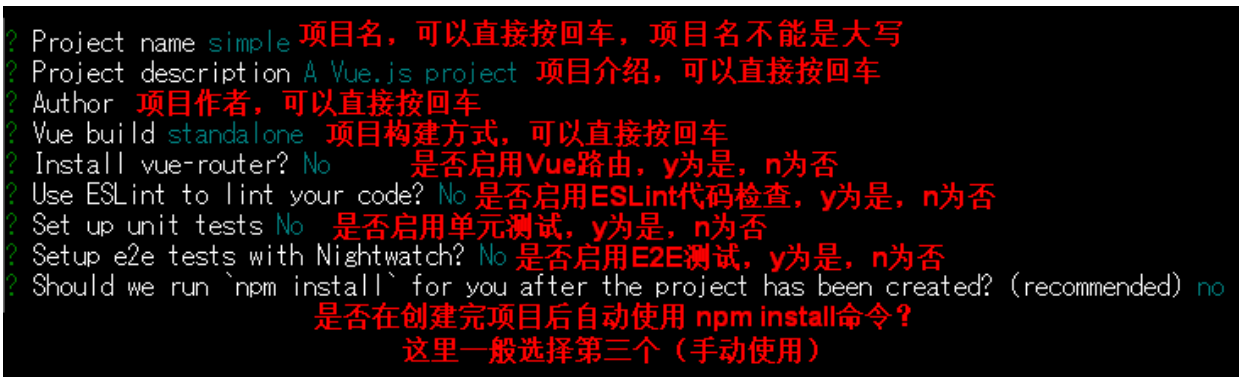
在命令行中执行`npm install -g vue-cli`。

创建模板项目

```
//vue init 模板名 项目名

vue init webpack simple
```

模板名可以在 https://github.com/vuejs-templates 查看，推荐使用webpack。



之后使用`cd 项目名`进入项目根目录，使用`npm install`来自动安装项目依赖。

其他常用命令

项目根目录下执行`npm run dev`可以测试运行项目，执行`npm run build`可以正式编译项目到dist目录下。

目录结构

单文件组件

```
<template>
    <!--template以下才是组件的HTML模板，仍然只能有一个根标签-->
    <div>
        我是一个单文件组件
    </div>
</template>

<script>
export default {
    //无需写template，data仍然以函数返回，其他属性照旧
    data(){
```

```
            return {
            }
        },
        props:[],
        methods:{
        }
    }
    </script>

    <!--这个scoped可以让这些样式仅在当前组件生效-->
    <style scoped>

    </style>
```

在脚手架中引入全局样式

> 如果我们有些样式是全局生效的，比如reset.css，或者手动对html、body标签进行样式调节，就不能把它们放在单文件组件里。

在`main.js`中：

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

//CSS文件也可以像模块一样引入
import "./assets/css/reset.css"
import "./assets/css/style.css"

Vue.config.productionTip = false

//...省略部分无关代码
```

Vue脚手架提供的便利

> Vue脚手架提供了一套比较完整的webpack配置，使我们可以将自己的精力专注于代码书写。

1. 使用单文件组件技术，可以极大的提高组件的使用便捷性。（实质是一种webpack的loader）
2. 使用样式作用域技术，可以在很大程度上避免组件间样式污染。（实质是一种webpack的插件）

3. 使用Babel自动对所有JS代码进行编译，使得项目中可以随意使用ES6的语法。（实质是webpack的Babel插件）
4. 使用webpack技术，对代码进行打包压缩，同时提供对JS模块化支持。（实质是webpack的技术）
5. 引入单文件组件时，可以不写`.vue`扩展名，可以在模块路径中使用`@`指向src目录。（实质是webpack的配置）
6. 当代码被修改时，页面自动刷新，实现热部署。（功能由webpack-dev-server提供）

补充：将Vue脚手架生成的文件部署到Tomcat中

1. 修改`/config/index.js`文件中`build`属性的`assetsPublicPath`属性（约46行）。

原内容：

```
  assetsPublicPath: '/',
```

修改为：

```
  assetsPublicPath: './',
```

2. 修改`/build/util.js`文件中约45行的内容。

原内容：

```
  // Extract CSS when that option is specified
  // (which is the case during production build)
  if (options.extract) {
    return ExtractTextPlugin.extract({
      use: loaders,
      fallback: 'vue-style-loader'
    })
  } else {
    return ['vue-style-loader'].concat(loaders)
  }
```

修改为：

```
  // Extract CSS when that option is specified
  // (which is the case during production build)
  if (options.extract) {
    return ExtractTextPlugin.extract({
      use: loaders,
      fallback: 'vue-style-loader',
```

```
      //其实就是添加了这个属性
      publicPath:'../../'
    })
  } else {
    return ['vue-style-loader'].concat(loaders)
  }
```

补充：在Vue脚手架中使用jquery、bootstrap等其他框架

jquery

在项目根目录安装jquery `npm install jquery`（注意大小写，jquery和jQuery不一样）。

build目录下的webpack.base.conf.js
开头加入

```
const webpack = require("webpack");
```

module.exports的最后加入（注意JS代码格式，不要忘了写逗号）

```
,plugins: [
    new webpack.ProvidePlugin({
        jQuery: "jquery",
        $: "jquery"
    })
]
```

之后，需要使用jquery的组件中导入jquery，如

```
<script>
    import $ from 'jquery'
    export default{
        data(){
            return {
                users:null
            }
        },
        created(){
            $.get("http://127.0.0.1:3000/all_user",function(data){
                this.users = data;
            }.bind(this))
        }
```

```
    }
</script>
```

bootstrap

1. 将`bootstrap.bundle.min.js`、`bootstrap.min.css`和`jquery.min.js`放在**static**目录下。
2. 在**index.html**中引入：

```
<!--必须要以/开头，否则非一级路由将有路径问题-->
<link rel="stylesheet" href="/static/bootstrap.min.css">
<!--必须要以/开头，否则非一级路由将有路径问题-->
<script src="/static/jquery.min.js"></script>

<script src="/static/bootstrap.bundle.min.js"></script>
```