## 一、基本简介

### 1、AngularJs

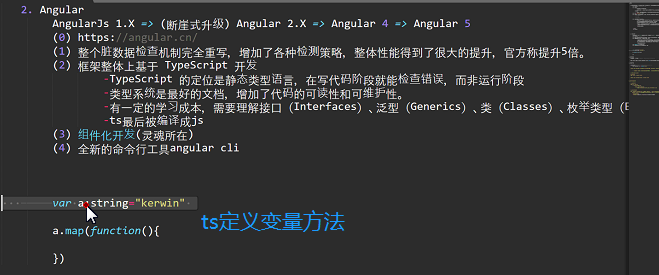


**解释**：SPA： single page application

MVW：model view whatever

更新数据原理：脏捡

### 2、断崖式升级：Angular



（1）typescript与js区别

A：typescript（微软）：定位式静态语言，在写代码的时候就能提示你出现了错误

A：js是解释型的动态语言，编写过程中不会提示错误，只有在解析的过程中才会提醒

B：ts最后被编译成js，写法自由

## 二、环境搭建



1. 全局安装脚手架一次

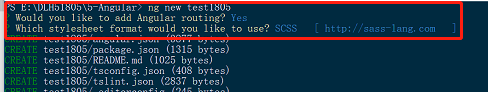
https://github.com/angular/angular-cli

npm install -g @angular/cli

1. 在文件夹上打开命令行，创建项目

ng new PROJECT-NAME --skip-install (默认npm安装太慢，可以取消，用cnpm 安装)

cnpm install



2、启动angular服务器 

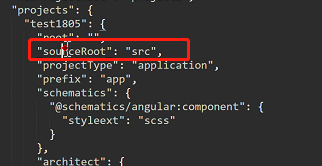
改变端口号方法：



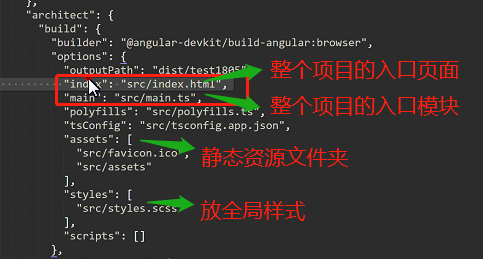
## 三．配置文件

在angular.json中

1、项目的核心入口文件



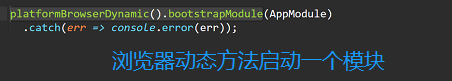
2、整个项目编译到dist下的tast1805下



3、后面的测试不用改

4、入口流程

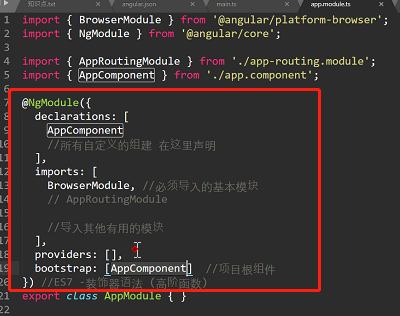
（1）在main.js中





AppModule是项目的根模块

（2）在app.moudle中



自动翻译代码，紧连着写，将navbar进行装饰成一个强大的类

**selector是组件的名字**

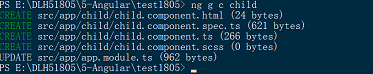


## 四、创建组件

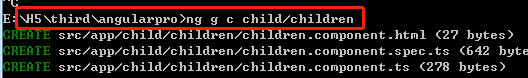
### （一）脚手架创建

1、脚手架命令行创建（每次创建的时候会自动给你声明好）

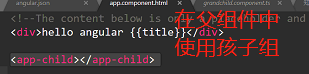
在app文件夹下创建组件，自动的找到src文件夹下的app来创建



在child文件夹下再创建一个组件



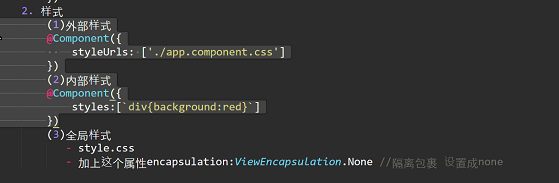
2、引入孩子组件



### （二）模板写法



### （三）样式





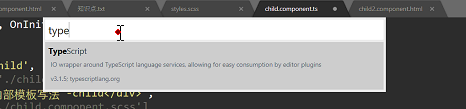
注意：1、样式不会发生覆盖

2、影响到其他选择器，去除包裹

encapsulation:ViewEncapsulation.None



3、安装typeScript会在编码时提示代码错误



4、专门开发typeScript的工具



### （四）Selector支持的写法



div里面写内容没用，会被渲染成组件的内容



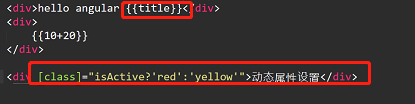


## 五、数据绑定

### （一）定义状态



### （二）使用状态



### （三）绑定事件



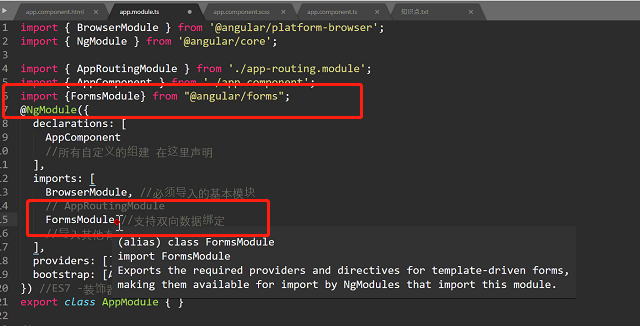


### （四）获取表单值

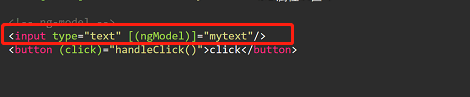
双向数据绑定

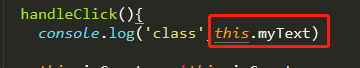
要实现双向事件绑定，要借助内置的模块





mytext是提前定义好的一个状态值





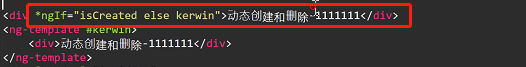
### （五）动态创建与删除

使用一个指令





if-else分支



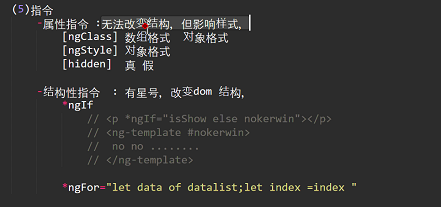
### （六）动态隐藏与显示

属性来对待

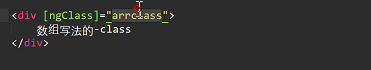




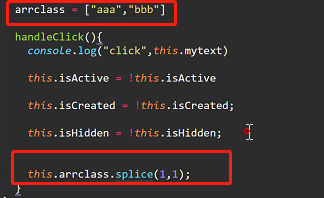
**（七）指令**



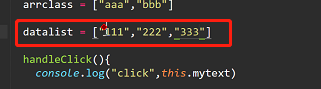
1. 数组写法

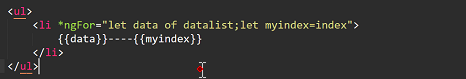


删除bbb样式



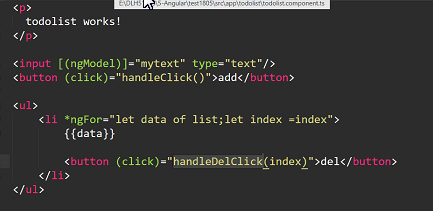
**（八）遍历**





**注意：只支持let换成var不好使**

**（九）ToDoList**





## 六、管道-数据格式化处理



1. 应用

后端返回给你一个时间戳的日期格式

| ： 管道符，将这个数据加工一下

（2）自己可以定义格式

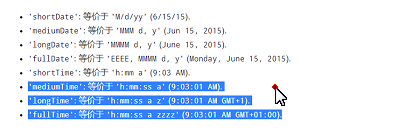
两个y会定义成18，文档中date有更多解释



显示结果



（3）快速写法



1. 大小写转换



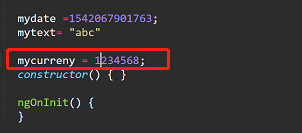




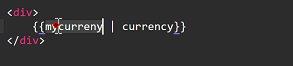
页面效果



1. 货币（默认美元）



在html中



页面显示：

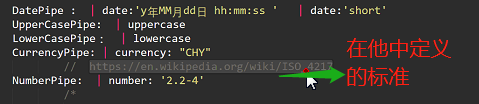


（1）支持传参：



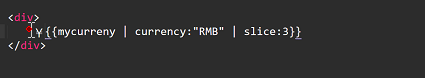
页面显示





（2）数据链

管道链 ,我们可以将多个管道连接在一起，组成管道链对数据进行处理。

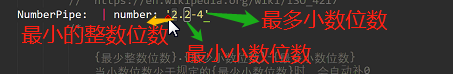


slice可以传多个参数

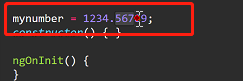
**后面接着写“：”**

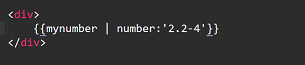
1. 数字管道，保留小数

最多保留几位，最少保留几位



定义一个状态：





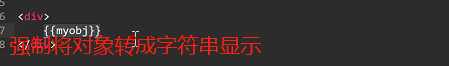
显示结果：



（1） 只保留两位小数



5、转化数据格式



将数据转换成json

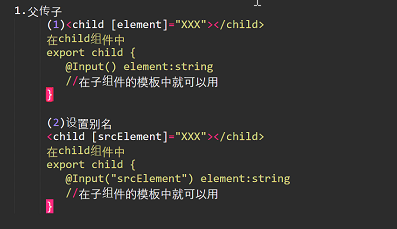


显示页面：



## 七、组件通信

### （一）父传子

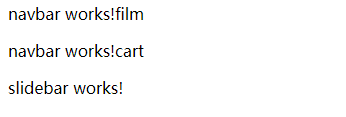


代码实现：





显示结果：



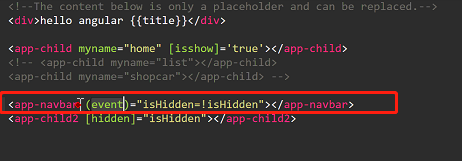
**注意：**也可以将childname定义成与父组件的一样



### （二）子传父

1、点击按钮控制显示与隐藏

在父组件中，自定义一个事件

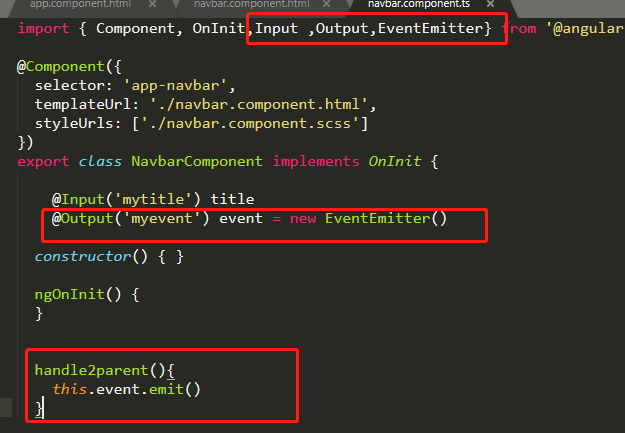


在子组件中：

点击按钮时，触发定义好的事件

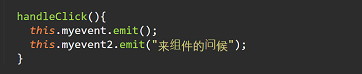
事件必须初始化

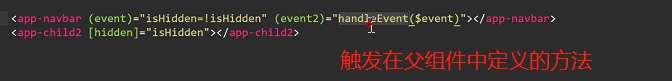




2、传值

子组件点击按钮，触发父组件的方法：

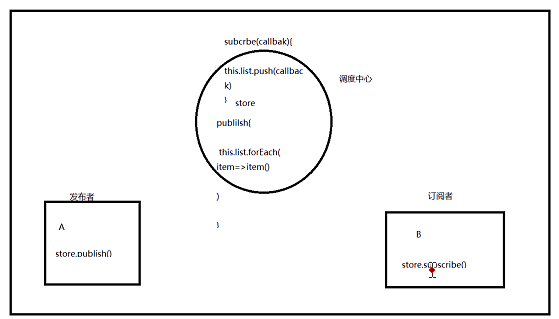




$event是传来的数据

### （三）复杂非父子

#### 1、观察者模式------订阅发布模式



步骤：

1. 新建store文件夹，并创建index.ts文件



1. 订阅者





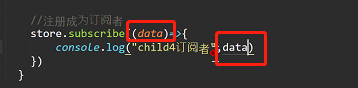
1. 发布者

点击按钮，分发

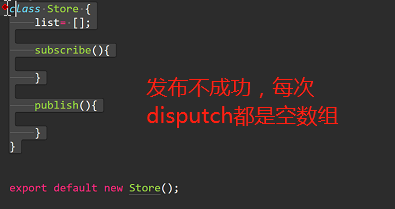


1. 发送真正的值





**类的写法，每次new一个新的对象**



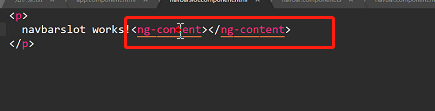
## 八、插槽

在父组件中能访问到父组件的作用域，在子组件中能访问子组件的作用域



案例：子传父





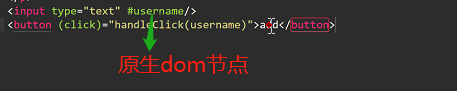
## 九、引用



### （一）应用

1. 获取输入框的值

（1）方法一



在组件中，无法直接通过this.username来拿值

（2）方法二，通过this.username来拿值

取到的依然是dom节点





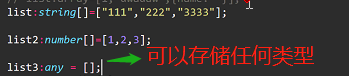
### （二）typeScript定义变量

1、如果最开始将title赋值为字符串，后来将它改成数字会报错。改不了类型。

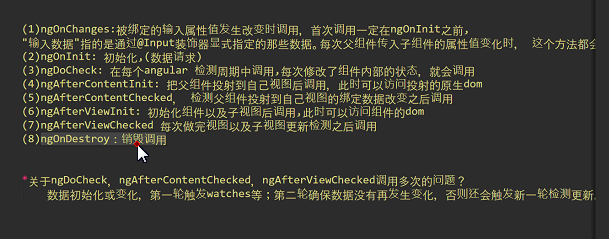
2、指定存储格式，可以不指定，会默任是第一次赋值内容的数据类型



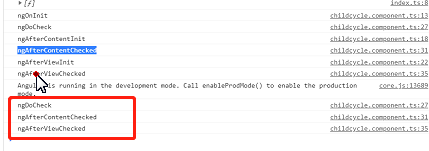
3、限制数组里只能存某种类型



## 十、生命周期



docheck：初始值



又走一遍为了确保数据没有再发生变化

## 十一、表单处理

### （一）表单驱动型表单校验

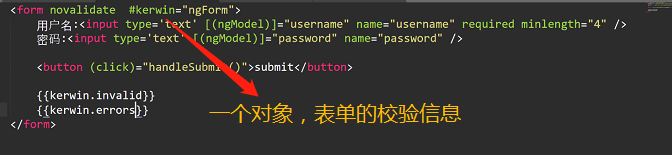
步骤：

1. novalidate 禁用浏览器默认验证

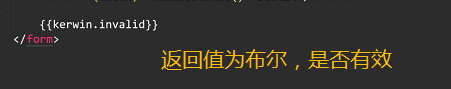
2. 引入 FormsModule

3. #kerwin= "ngForm" (使用 # 把表单指令对象导出到 kerwin 这个可引用变量中)

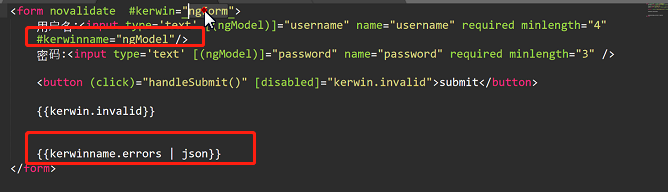
NgForm指令:Angular会在<form>标签上自动创建并附加一个NgForm指令。它会控制那些带有ngModel指令和name属性的元素，监听他们的属性（包括其有效性）。

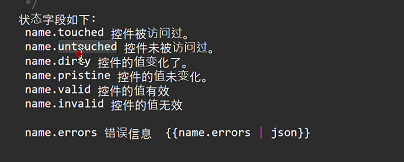


校验整个表单是否满足

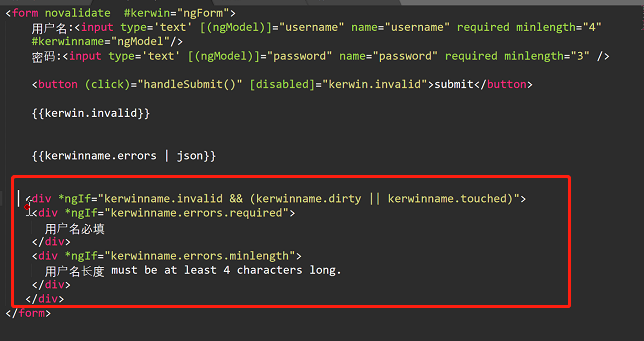


2、给每个表单加校验





根据校验提示信息



### （二）响应式表单验证



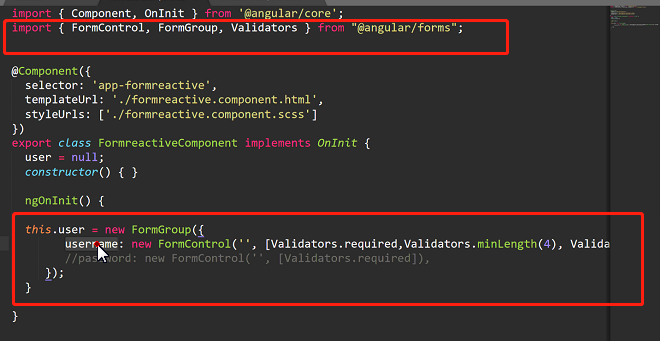
双向数据绑定不在好用了

步骤：

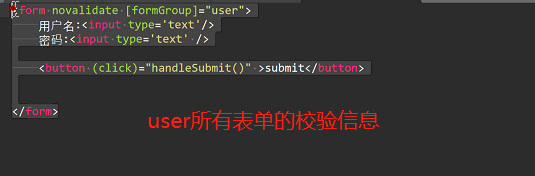
1、

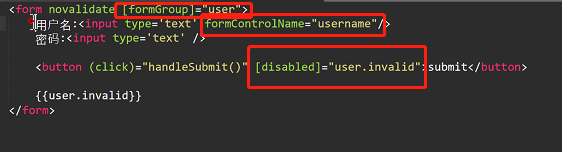


2、引入模块

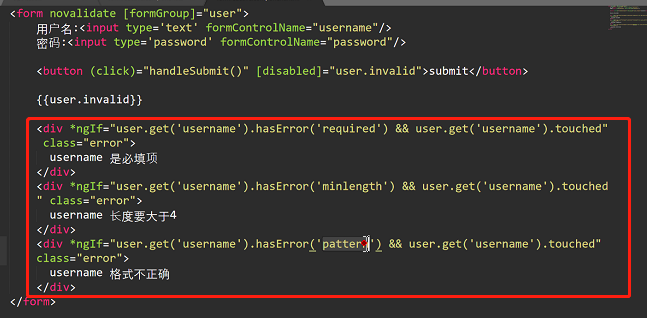


在表单上

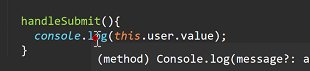




3、判断提示信息的显示与隐藏

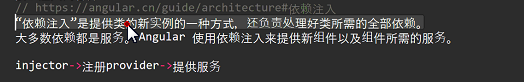


4、取到表单的值



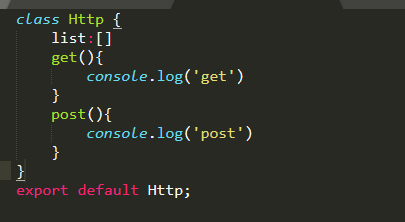
## 十二、依赖注入

### （一）内容介绍



1、使用

创建http.ts文件



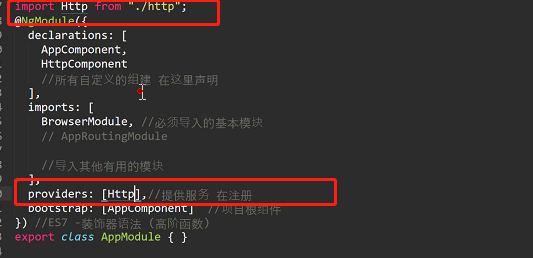


2、依赖注入帮你new好，不需要你自己来new

（1）让注入器知道你的服务

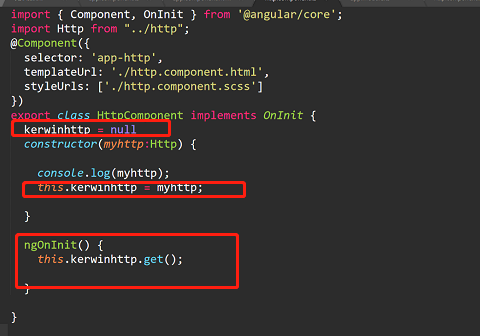


解决：配置供应商，将服务在根模块中注册，这样的话任何组件引入的都是同一个实例

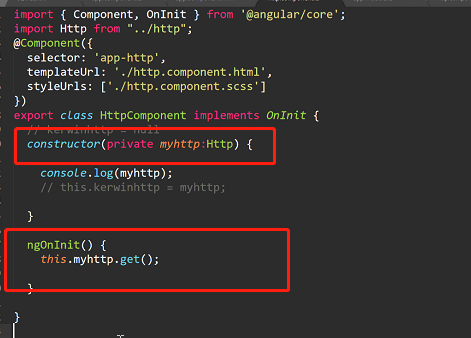


（2）在生命周期里面调用

方法一：



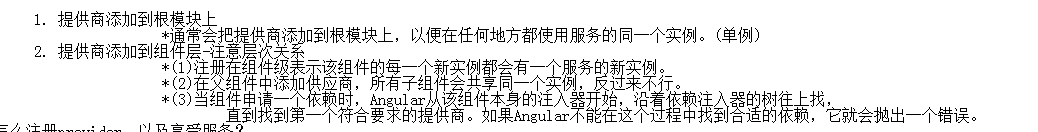
方法二：



### （二）优点



### （三）组件级服务：在组件中注册provider





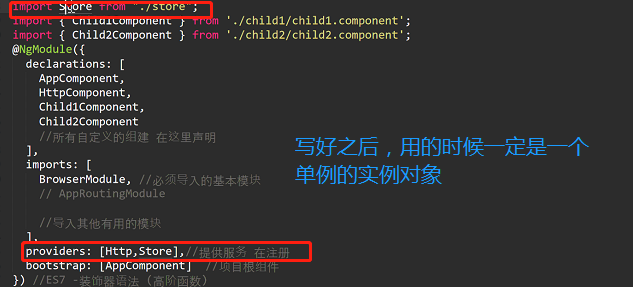
**注意：如果在根组件和自己的组件中都注册了服务，优先调用自己的服务**

### （四）通信

1、在store的index.ts文件夹中，定义一个类



2、在跟模块引入store服务



3、发布消息



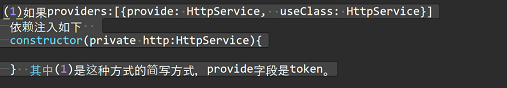
4、订阅者



注意：如果是组件级的服务，就实现不了通信了，就不是new出一个实例了

### （五）注册服务写法

（1）简写



（2）完整写法



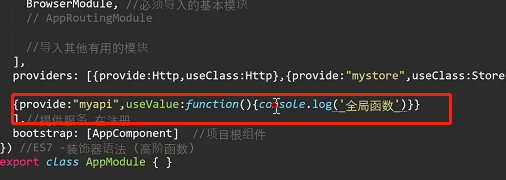
（3）在根组件引入store，子组件就不用引入的写法



缺点：重名会被覆盖，字符串不会检查



（4）定义一个全局函数



获取全局函数

### （六）特别的功能



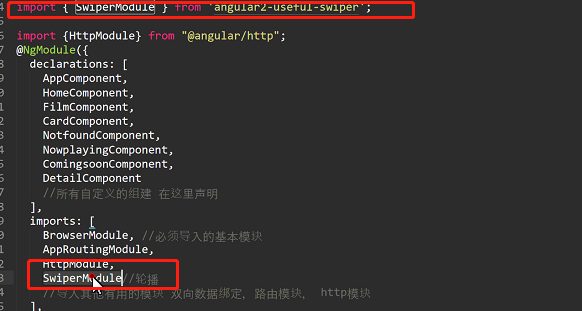
想要获取兄弟组件的属性，可以将属性放到服务上

### 十三、路由

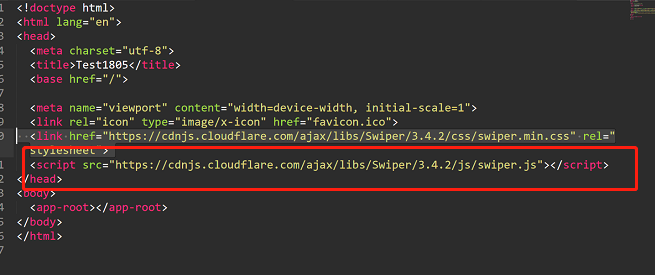
十四、轮播

1. 下载模块

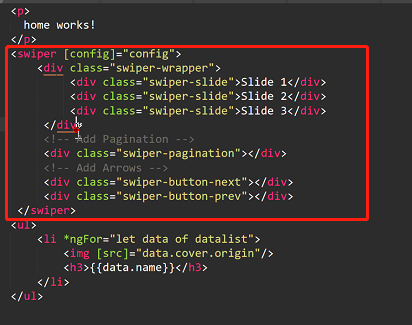
2、在跟模块中引入轮播



在index.html中引入样式



3、使用轮播



4、配置congif变量

自己封装swiper组件

## 拓展

1、sublime托文件时，不想要文件 settings

**"folder\_exclude\_patterns":**

**[**

**"node\_modules",**

**".svn",**

**".git"**

**],**

2、 轮播图原理

1.先让图片轮播起来。基本就是写一个 play函数里面加定时器，每次使图片的index对象加一，当index大于最大值时，设置index等于第 一张图片.这样轮播图就动起来了。

    2. 轮播图动起来基本就是只显示一张图片隐藏其他的图片。我上面使用的是opacity 。

    3. 图片下面的按钮。主要就是使下面的按钮与上面的图片一一对应。然后点击下面的按钮显示对应的图片。

     4. 左右的上一张和下一张按钮。点击左边的上一张按钮图片向前显示，实现原理就是使 index 对象减一。点击左边的下一张按钮图片向后显示，实现原理就是使 index 对象加一。

    5. 对应上一张和下一张连续点击的问题就是给这两个按钮加上延时器。

    6. 当鼠标放在轮播图区域时停止轮播，实现原理就是清除定时器，离开开始轮播就是加上定时器。