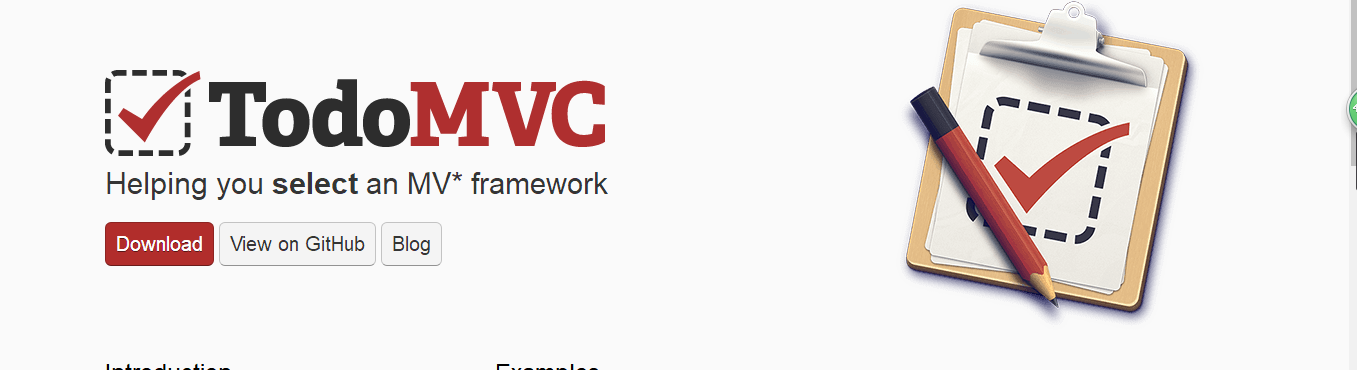
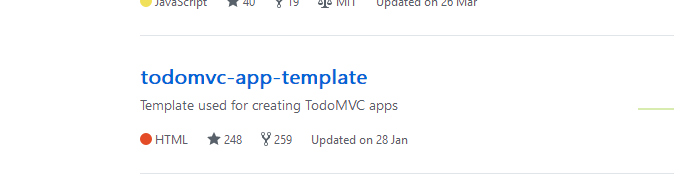
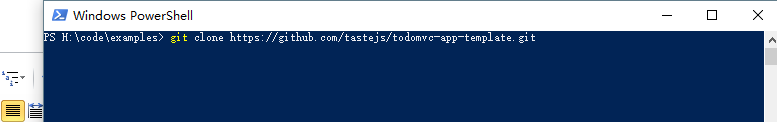
# todomvc.com

在github上面可以找到 做这个练习的的模板文件:项目文件为：

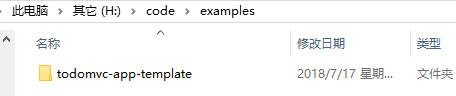


上面的样式都写好了 我们只需要实现他的功能就可以了

# angular clone项目



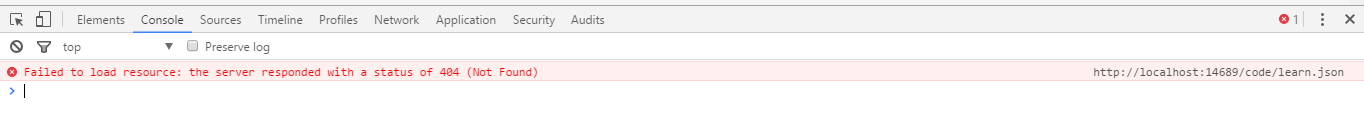
它会在当前文件夹下面新建一个与url地址里面项目名相同的文件夹。



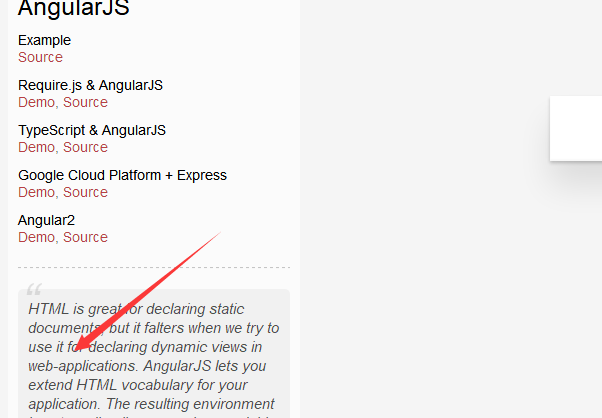
在项目文件夹下执行npm install 命令 下载依赖



在浏览器中访问发现报错



原因是learn.josn 是用来官方展示的时候才要用这个东西

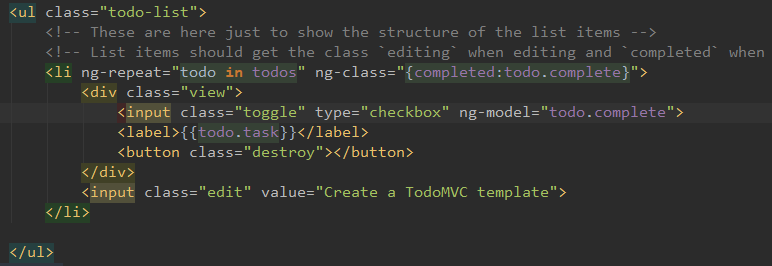


在页面怎么取消这个请求呢？把页面中的这个请求去掉就可以了





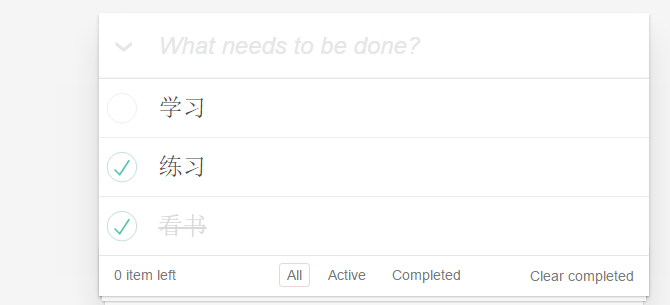
node\_modules/todomves-common可以卸载掉 这些是对在官网展示是说要到的一些样式和js。



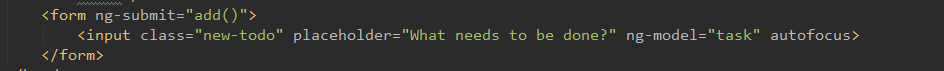
ng-class="{completed:todo.complete}" 如果todo.complete为true则显示completed这个类名，否则不显示。

这里用ng-model不用ng-checked的原因是 前者是双向的数据绑定而后者是单向的数据绑定。但是这里的功能要求是选中为完成，未选中表现为未完成状态。

ng-ckcecked表现为：选中了 但是todo.complete的值没有变成true



增加数据：这里用form标签包裹的原因是为了实现当敲回车的时候能够自动提交。在form表单中 submit提交可以实现当敲回车自动提交的功能。



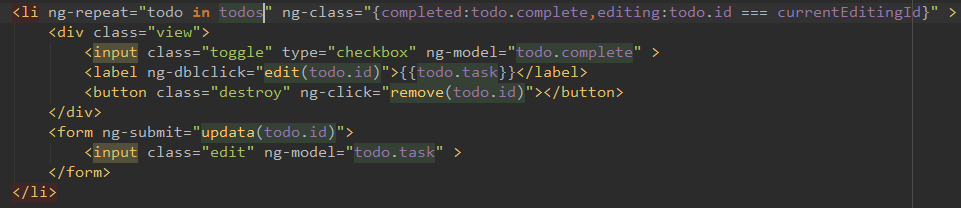
全选：ng-checked=”checkALl” 当checkALl为true时 被选中 否则不选中。



删除：执行一个点击事件，通过传递参数的形式，通过id知道是删除那个任务。



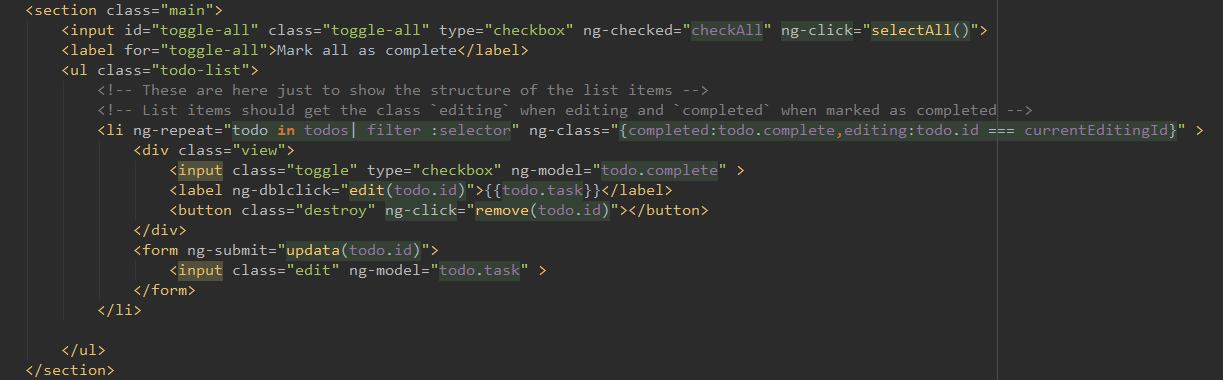
双击编辑：当双击任务的时候，通过id知道是那个任务进行编辑，就在那个元素上面加上editing的类名。

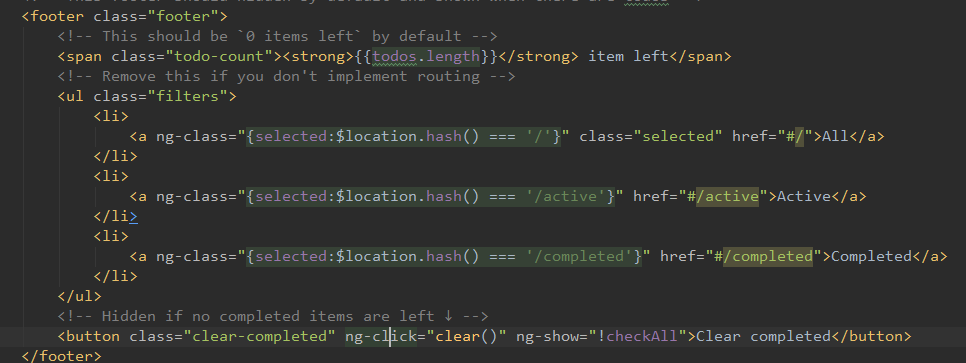


通过判断当前的id是否与正在编辑的id相同，决定editing是否显示。

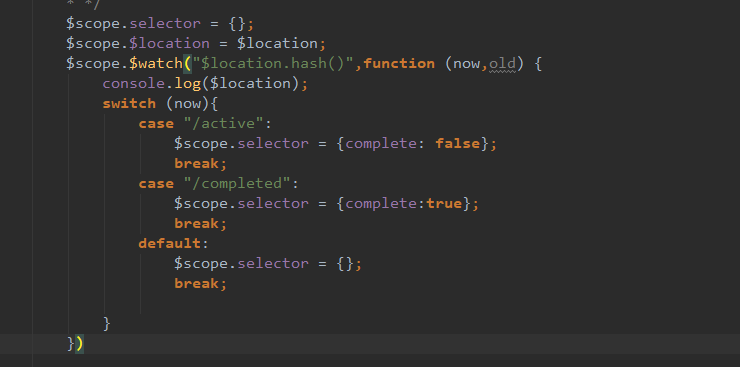
## 通过用filter 和switch 实现切换的功能。

html:



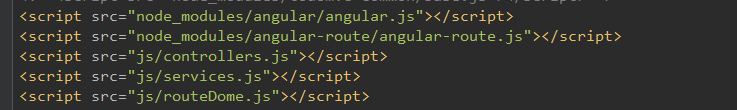


js



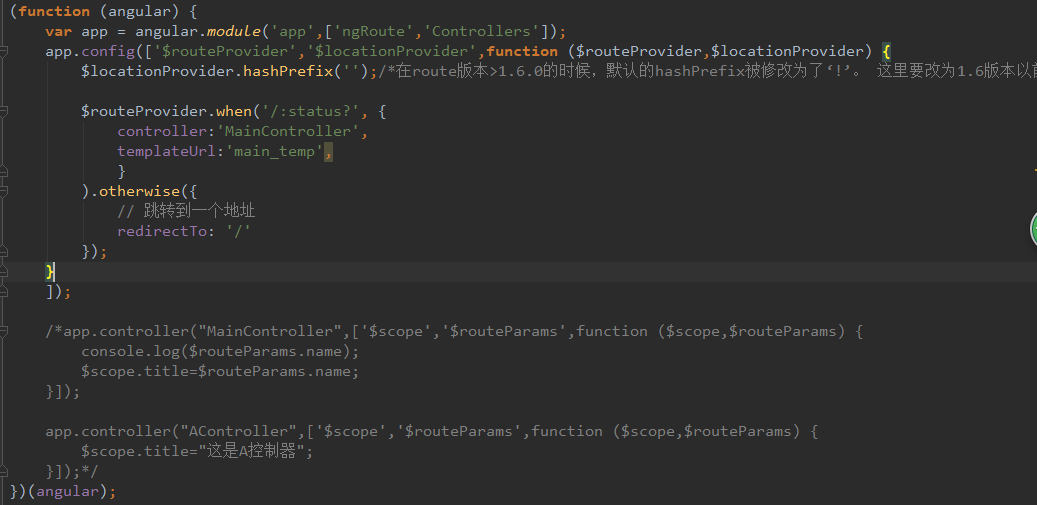
# 通过ng-route路由来实现切换的功能

首先要下载依赖的包 npm install angular-route,并在页面导入。



## routeDome.js

然后在routeDome.js文件主模块里面配置路由



主模块依赖ngRoute、Controllers 这两个模块，而Controllers这个模块为自己定义的模块。

通过$routeProvider来控制路由。

## Controllers.js

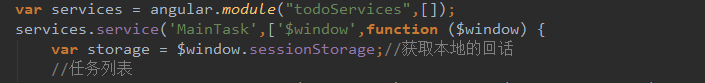
在Controllers.js这个文件里面定义一个新的模块，这个模块也依赖了一个新的模块todoServices，在controllers这个模块里面应该定义一些操作界面的方法，而不是直接操作数据的方法。

在定义控制器的时候，通过依赖注入的方式把自定义的服务MainTask注入进去，这样就可以使用另外一个模块的方法了。



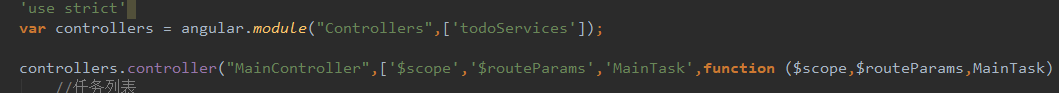
## services.js

在这个文件里面定义了一个新的模块，在模块里面定义了一个新的服务。服务的名字叫MainTask，这个服务依赖注入了$window。



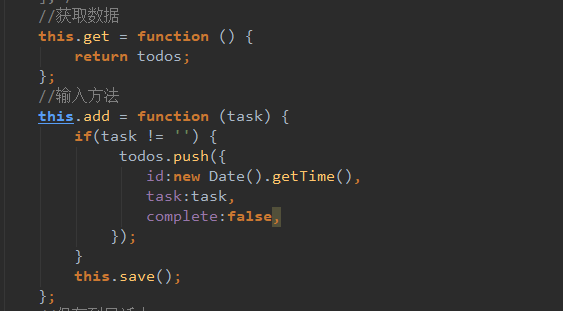
个人理解就是定义一个服务就是定义一个对象，service(参数1，参数2)里面的参数2就是对象的构造函数，而前面参数1就是构造函数的实例对象，而参数1就是实例对象的名字。

等同于 ： var MainTask = new 参数2;

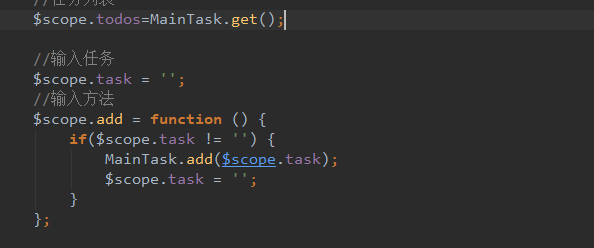


同时在Controllers.js 里面依赖该模块，并在定义控制器的时候，注入该对象MainTask，就可以在控制器里面调用该对象的方法。

在Maintask定义的方法



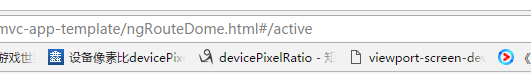
在controller里面调用方法

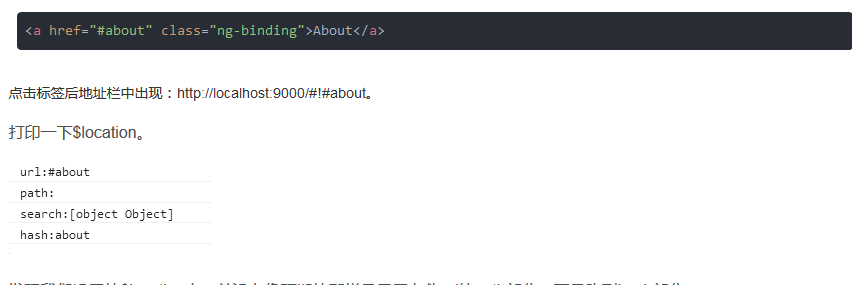


# 注：ngRoute路径出现#!#解决方案

在新版本的angular@1.6.0及以上版本时，配置ngRoute后，点击html#/a显示内容，点击超链接后url跳转去了html#!/a#%2Fa，

1. 旧版本（@1.5.9）就没有这个问题，url为html#/a。





发现我们设置的$location中，并没有像预期的那样是显示在伪url的path部分，而是跑到hash部分。

原因是：在route版本>1.6.0的时候，默认的hashPrefix被修改为了‘!’

相当于以下代码：locationProvider.hashPrefix('!');



参考：<https://blog.csdn.net/qq_28506819/article/details/72795851>

<https://blog.csdn.net/github_38469481/article/details/70598066>