**一、、spring boot 自动更新静态文件**

https://blog.csdn.net/qq\_15870133/article/details/78732680

**二、resful的get请求，默认不支持从客户端传递json数据到服务端**

**三、spring mvc拦截器和filter过滤器在跨域时，仅靠addCorsMappings设置是不行的（该设置只能用于普通Controller），还必须设置corsFilter并将其设置为过滤器中优先级最高**

**四、配置跨域设置后，session也可以在不同域名间传递，解决了前后端分离时登陆状态的保持。**

服务端： .allowCredentials(true)

客户端： $.ajaxSetup({

xhrFields: {

withCredentials: true//要进行跨域必须设置，但在服务端也必须允许。这样可以保持登录状态

},

});

**五、获取项目的根目录（物理路径）**

String basepath=ClassUtils.getDefaultClassLoader().getResource("").getPath();

说明：这里不能用用

rquest.getSession().getServletContext().getRealPath("/")

来获取路径，因为这是spring boot内置tomcat的原因，每次启动项目他的Tomcat的路径都不相同，是虚拟的路径，而且很多，

过一段时间它可以自动清楚这些不用的文件夹，这个方法在spring boot里面不好用，也是刚要使用一个图片，

放在tomcat服务器上可以访问到，但是重启就访问不到了，

**六、ueditor与本项目spring boot2的整合（跨域）-ljy根据实际情况记录（耗时三天，踩无数坑得出）**

参考此地址进行设置（但有许多小坑需要自己处理）：https://blog.csdn.net/qq\_33745799/article/details/70031641

**6.1 服务端设置**

（1）从服务器上面下载ueditor的源码（java部分），将所有源码拷贝到项目中

本项目源码地址：com.baidu.ueditor

（2）在maven中下载ueditor源码所需要的包（注意不要使用网上的ueditor-spring-boot-starter，容易出错而且没什么用）

<dependency>

<groupId>commons-io</groupId>

<artifactId>commons-io</artifactId>

<version>2.4</version>

</dependency>

<dependency>

<groupId>commons-codec</groupId>

<artifactId>commons-codec</artifactId>

<version>1.10</version>

</dependency>

<!-- https://mvnrepository.com/artifact/commons-fileupload/commons-fileupload -->

<dependency>

<groupId>commons-fileupload</groupId>

<artifactId>commons-fileupload</artifactId>

<version>1.3.1</version>

</dependency>

（3）解决ueditor的跨域问题，必须要对本项目的CorsConfig做好跨域设置，建议除域名外用\*代替

说明：本项目跨域类：com.lcvc.ebuy\_springboot.config.CorsConfig

（4）在ueditor的config.json拷贝到项目resources里面，为了方便管理

详细路径：resources/static/plugins/ueditor/config.json

（5）在config.json中，根据项目对上传文件的管理进行设置

（6）参照上述网址攻略：创建UEditorController类，用于取代ueditor的demo服务端接口controller.jsp页面

说明：该类作为ueditor访问的唯一服务端地址（与controller.jsp作用一致），这个接口会根据客户端的动作（上传图片、文件或是显示文件列表）来决定调用剩余的类。

（7）参照上述网址攻略：修改ConfigManager类的getConfigPath方法，重新指向服务端路径的config.json，使其配置生效

20191226更改：不再重新指向，因为spring boot打包为jar后内部路径已经改变（之前为war），推荐使用spring boot自带的ClassPathResource类来读取内部文件，但是使用该类也无法获取文件的真实路径（会出错），故废弃路径，直接读取文件。

（8）参照上述网址攻略：修改BinaryUploader类，因为SpringMVC框架对含字节流的request进行了处理，此处传的是处理过的request，故获取不到字节流。此时采用SpringMVC框架的解析器multipartResolver。

正常情况下：设置好该步后图片就可以正常上传，但是返回客户端的图片会无法显示。

（9）大坑（ljy自己解决，用了三天尝试各种方案后作出当前决定）：在实际操作中，出现回显路径问题（如果按实际项目需求存储在E盘或D盘下则不会出现此类问题），在UEditorController类处理结果的返回值中（String exec = new ActionEnter(request, rootPath).exec();）进行如下处理

（1）图片上传后的回显路径会包括spring boot的static，但是spring boot中直接访问是不需要static的，故在最后的回显地址中将static/替换掉

（2）在ueditor查看文件或图片目录时，图片会按照物理路径显示，导致图片无法显示。故将物理路径替换掉，改为相对路径

Ljy说明：如果需要对ueditor的上传规则重新设定（更复杂的规则），可以自己定义上传类（该类还有尝试后留下的代码，已注释），和普通的spring mvc上传处理方式一样，非常简单。但是如果想简单的使用ueditor的规则不想变动，但是原配置文件的改动又无法纠正结果的话，就只能简单粗暴的对返回路径进行处理了。

**6.2 客户端设置（前端）**

（1）将ueditor的文件夹导入前端项目中，可以将Jsp文件夹删除

注：ueditor文件夹不需导入服务端项目，服务端只需config.json文件

（2）修改ueditor.config.js文件，使其能够正确的访问服务端（上述第6步）

例如：

// 服务器统一请求接口路径

//, serverUrl: URL + "jsp/controller.jsp"

, serverUrl: "http://127.0.0.1:8088/ebuy\_springboot/api/ueditor"//注意：如果服务器更换务必更改。读取ueditor原controller.jsp的替代控制器

（3）剩下的按照编辑器常规方式调用即可

**6.3 引入spring security后出现的CORB（X-Content-Type-Options: nosniff）导致无法读取ueditor配置的问题-解决了2周**

（1）问题出现原因（引入spring security后出现）

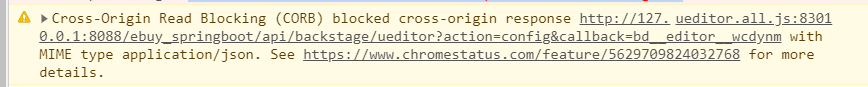
互联网上的资源有各种类型，通常浏览器会根据响应头的Content-Type字段来分辨它们的类型。例如：”text/html”代表html文档，”image/png”是PNG图片，”text/css”是CSS样式文档。然而，有些资源的Content-Type是错的或者未定义。这时，某些浏览器会启用MIME-sniffing来猜测该资源的类型，解析内容并执行。

简单的说，如果Web应用没有返回Content-Type，那么浏览器将拒绝加载相关资源。针对本项目而言：如果服务器发送响应头 “X-Content-Type-Options: nosniff”，则 script 和 styleSheet 元素会拒绝包含错误的 MIME 类型的响应。这是一种安全功能，有助于防止基于 MIME 类型混淆的攻击。

（2）疑惑：服务端已经设定了json类型

response.setContentType("application/json;charset=UTF-8");

但是依旧出现以下问题



（3）解决方式（具体参照项目里的com.lcvc.ebuy\_springboot.web.action.plugins.ueditor. UEditorController）

将json类型手动指定Content-Type为Javascript类型（"application/ecmascript"，"application/javascript"，"application/x-javascript"，"text/ecmascript"，"text/javascript"，"text/jscript"，"text/x-javascript"，"text/vbs"，"text/vbscript"）的一种即可

response.setContentType("text/javascript");

（4）特别说明：

建议后期的跨域的数据请求转到本站服务器（用Node.js），由本站服务器去做跨域请求，即跳过浏览器同源策略的限制

**6.4 缺陷（待解决）**

在使用ueditor多图上传（含视频、涂鸦）时，不知道什么原因无法将登陆状态的JSESSIONID正确值跨域传递到服务端，导致无法进行登陆状态的验证，影响实际项目开发。最后选了个个折中方案：因为ueditor每次调用上传组件前会先访问服务端配置，如果访问不了服务端配置无法调用相关组件，所以这里只要拦截了服务端配置就可以（暂行办法），其他的请求不拦截。

if(path.equals("/api/backstage/ueditor")){

String method=request.getParameter("action");

if(!method.equals("config")){

flag=true;//直接通过验证

}

}

但是这样是有问题的，比如用户已经注销了，但是只要不刷新页面，服务端配置还是一直保存，允许上传文件

**20191024更新解决办法（不完美）：**

在前端登陆成功后，将服务端的session对象的id存下来，然后在ueditor.config.js的地址上加上jsessionid传递到服务端，就可以保存登陆状态了，如下图

 , serverUrl: server+"api/backstage/ueditor;jsessionid="+JSESSIONID+"?"//注意：如果服务器更换务必更改。读取ueditor原controller.jsp的替代控制器

但是存在问题，比如账户注销再登陆后，如果不刷新首页就无法重新获取最新的jsessionid

**20200130出现问题：**

在使用vue单独做前端项目的过程中发现，之前无法获取jsessionid的Ueditor上传文件配置，layui开发的项目又自动获取到了cookie，非常诡异，因为开发环境和运行环境都围边，所以**20191024更新解决办法（不完美）**中的设置暂时取消。

但是出现了另外的问题，在vue中使用的ueditor依旧不会获取到cookie，所以在vue中继续使用jsessionid成功处理session的问题。同时还有个更诡异的问题，在使用新的后台前端vue的同时，本来ueditor不会自动携带cookie的，但是如果使用layui打开ueditor，那么vue的ueditor也会获得layui中的cookie信息，导致自己创建的jsessionid失效，所以这两个项目在测试的时候不能同时运行。初步考虑是在同一个域名下的冲突。被这个莫名其妙的问题卡了三天，服了。

**七、关于在spring mvc中接收前端的JSON对象（关联属性时）的处理办法**

问题出现：首先，由于前后端分离，表单值的传递使用JSON而不是以往的$("#myForm").serialize()序列化传递，这导致了如下情况的出现

public class Product implements java.io.Serializable {

private Integer id;

private ProductType productType;//所属产品类别

private String name;//产品名称

……

}

从上述代码可以得知，productType这种关联属性，在以往用$("#myForm").serialize()传递时，只要在前端使用<input name="productType.id">传值，后端就可以自动将id值封装到productType属性中。但是在JSON传递时，如果还是采用<input name="productType.id">传值，那么传递过来的JSON值就是

{"productType.id":"6","name":"cc","orderNum":"100","price":"","originalPrice":"","linkUrl":""}

这样是无法封装到productType属性中的。

此时可以有两种做法，一种是在服务端接收ID值然后自己封装，但是这样会让接口显得比较混乱，因此使用第二种做法，就是在客户端自己封装关联属性再转换为JSON会更好。

例如：

var productType ={//创建所属产品类别对象

id: $("#productTypeId").val()//

};

data.field.productType=productType;//将产品类别对象组装到表单中

最后再通过

JSON.stringify(data.field)

获取最终的JSON值

{"name":"cc","orderNum":"100","price":"","originalPrice":"","linkUrl":"","productType":{"id":"6"}}

可以看到现在的JSON值才是能够正确表达关联对象的，在spring mvc中也能够正确的封装到productType属性中了。

**八.关于layui的一些问题**

1.layUi的select在赋初始值的时候，如果是逻辑类型，如下图

<select id="onSale" name="onSale" lay-verify="required">

<option value="">请选择</option>

<option value="true">上架</option>

<option value="false">下架</option>

</select>

这个时候如果从服务端传逻辑类型的数值过来，如true，并不能在自动赋值，必须要修改一下为下述代码：

form.val("myForm", {

,"click":product.click

,"onSale":product.onSale.toString()//转化为字符串，直接用布尔类型会无法设置

,"description":product.description

});

即加上.toString()转化为字符串即可

2.关于layui表单赋值的部分问题

（1）在某些特定情况下，如ueditor的图片上传组件赋值给表单，我用jquery的方法赋值

$("#pirUrl").attr("value", arg[0].src);

但是很奇怪的是，在产品添加页面可以操作，但产品编辑页面不行，所以为了防止类似的问题，可以采用layui的表单赋值方式来操作

form.val("myForm", {"picUrl":arg[0].src});

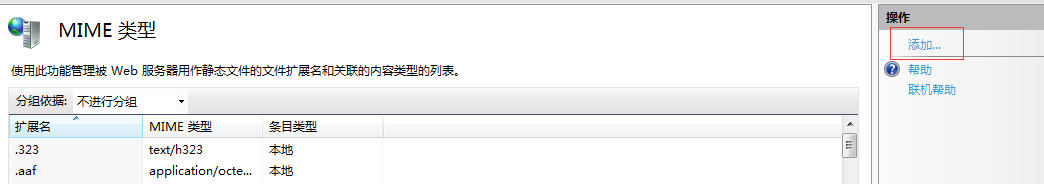
3.在IIS中发布项目，虽然不影响项目运行，但是提示ayui/font/iconfont.woff?v=240 net::ERR\_ABORTED 404 (Not Found)

原因：原因是 IIS 中未添加相应MIME类型，只需在 IIS 主页双击进入【MIME类型】，并添加相应扩展名及MIME类型对应关系即可。

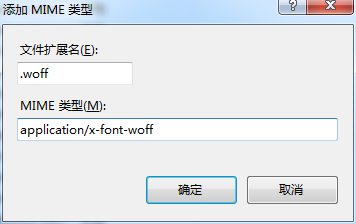
操作：



选择mime类型



关系键值对如下

：

svg image/svg+xml

.woff2 application/x-font-woff