# springboot项目上线启动后无法访问到controller

问题描述：springboot项目本地运行一切正常，但是在部署到阿里云ecs服务器上并且成功启动后，在外部无法的请求无法到达controller；

关于SpringBoot应用挂了很久之后，会发生Invalid character found in method name. HTTP method names must be tokens的问题。

解决方案：

方案1（实在没办法的时候再用）：重启服务器；

方案2（待测试）：先重启服务器，然后修改设置一下application.yml文件的tomcat下的max-http-header-size即可。

server:

port: 5555

servlet:

context-path: /xxxx

tomcat:

remote-ip-header: x-forward-for

uri-encoding: UTF-8

max-threads: 1000

max-http-header-size: 8096

详细内容参考博客<https://blog.csdn.net/moshowgame/article/details/82597326>

# 前端（easyUI）传数据到Java后台（springboot）出现以下错误：

Content type 'application/x-www-form-urlencoded;charset=UTF-8' not supported

解决方案：当前端请求的Content-Type是Json时，可以用@RequestBody这个注解来解决。@RequestParam 底层是通过request.getParameter方式获得参数的，换句话说，@RequestParam 和request.getParameter是同一回事。因为使用request.getParameter()方式获取参数，可以处理get 方式中queryString的值，也可以处理post方式中 body data的值。所以，@RequestParam可以处理get 方式中queryString的值，也可以处理post方式中 body data的值。@RequestParam用来处理Content-Type: 为 application/x-www-form-urlencoded编码的内容，提交方式GET、POST。

@RequestBody接受的是一个json对象的字符串，而不是Json对象，在请求时往往都是Json对象，用JSON.stringify(data)的方式就能将对象变成json字符串。

总结：

前端请求传Json对象则后端使用@RequestParam；

前端请求传Json对象的字符串则后端使用@RequestBody。

参考博客：https://blog.csdn.net/feiyst/article/details/88431621

记录时间：2020-01-10 00:03

# mybatis基于注解实现关联查询

//one to one

@Select("select \* from user where userId = #{userId}")

@Results({

@Result(id=true,column="userId",property="userId"),

@Result(column="userName",property="userName"),

@Result(column="userPassword",property="userPassword"),

@Result(column="userAge",property="userAge"),

@Result(column="userEmail",property="userEmail"),

@Result(column="userId"(需要传递的参数),property="userAddress",one=@One(select="com.cr.mapper.AddressMapper.getAddressByUserId",fetchType= FetchType.EAGER（表示该实体对象采用急加载模式）))

})

public User getUserById(int userId);

//one to many

@Select("select \* from user where userId=#{userId}")

@Results({

@Result(id=true,column="userId",property="userId"),

@Result(column="userName",property="userName"),

@Result(column="userPassword",property="userPassword"),

@Result(column="userAge",property="userAge"),

@Result(column="userEmail",property="userEmail"),

@Result(column="userId",property="books",many=@Many(select="com.cr.mapper.BookMapper.getBooksByUserId",fetchType= FetchType.LAZY（表示该实体对象采用懒加载模式）))

})

public User getUserAndBooksById(int userId);

附加：懒加载和急加载的区别：

1、FetchType.LAZY：懒加载，加载一个实体时，定义懒加载的属性不会马上从数据库中加载。

2、FetchType.EAGER：急加载，加载一个实体时，定义急加载的属性会立即从数据库中加载。

3、比方User类有两个属性，name跟address，就像百度知道，登录后用户名是需要显示出来的，此属性用到的几率极大，要马上到数据库查，用急加载；

而用户地址大多数情况下不需要显示出来，只有在查看用户资料是才需要显示，需要用了才查数据库，用懒加载就好了。所以，并不是一登录就把用户

的所有资料都加载到对象中，于是有了这两种加载模式。

# 方法异常捕获后关于数据库事务的处理

问题描述：

加了数据库事务的方法，在其内部捕获异常后事务回滚失效；

解决方案：在catch代码块里面抛出异常，或者在里面手动控制回滚;

记录时间：2020-03-01