springBoot，默认使用的json解析框架是Jackson。

这里配置使用fastjson替换掉jackson

pom.xml中添加依赖

<dependency>  
 <groupId>com.alibaba</groupId>  
 <artifactId>fastjson</artifactId>  
 <version>1.2.23</version>  
</dependency>

配置fastjson有两种方式

第一种方式：

① 启动类继承 WebMvcConfigurerAdapter

②覆盖configureMessageConverters方法

@SpringBootApplication  
public class SpringbootFastjsonApplication extends WebMvcConfigurerAdapter{  
  
 // 方法一：extends WebMvcConfigurerAdapter  
 @Override  
 public void configureMessageConverters(List<HttpMessageConverter<?>> converters) {  
 super.configureMessageConverters(converters);  
  
 //1、先定义一个convert转换消息的对象  
 FastJsonHttpMessageConverter fastConverter = new FastJsonHttpMessageConverter();  
 //2、添加fastjson的配置信息，比如是否要格式化返回的json数据；  
 FastJsonConfig fastJsonConfig = new FastJsonConfig();  
 fastJsonConfig.setSerializerFeatures(  
 //是否需要格式化  
 SerializerFeature.*PrettyFormat* );  
 //附加：处理中文乱码（后期添加）  
 List<MediaType> fastMedisTypes=new ArrayList<MediaType>();  
 fastMedisTypes.add(MediaType.*APPLICATION\_JSON\_UTF8*);  
 fastConverter.setSupportedMediaTypes(fastMedisTypes);  
 //3、在convert中添加配置信息  
 fastConverter.setFastJsonConfig(fastJsonConfig);  
 //4、将convert添加到converters  
 converters.add(fastConverter);  
 }  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(SpringbootFastjsonApplication.class, args);  
 }  
}

第二种方式：

直接注册HttpMessageConverts的Bean

@Configuration  
@ConditionalOnClass({JSON.class})//判断JSON这个类文件是否存在，存在才会创建配置。  
public class FastJsonHttpMessageConvertersConfiguration {  
 @Bean  
 public HttpMessageConverters fastJsonHttpMessageConverters(){  
 //1、先定义一个convert转换消息的对象  
 FastJsonHttpMessageConverter fastConverter = new FastJsonHttpMessageConverter();  
 //2、添加fastjson的配置信息，比如是否要格式化返回的json数据；  
 FastJsonConfig fastJsonConfig = new FastJsonConfig();  
 fastJsonConfig.setSerializerFeatures(  
 SerializerFeature.*PrettyFormat*,  
 SerializerFeature.*WriteClassName* );  
 //附加：处理中文乱码  
 List<MediaType> fastMedisTypes = new ArrayList<MediaType>();  
 fastMedisTypes.add(MediaType.*APPLICATION\_JSON\_UTF8*);  
 fastConverter.setSupportedMediaTypes(fastMedisTypes);  
 //3、在convert中添加配置信息  
 fastConverter.setFastJsonConfig(fastJsonConfig);  
  
 HttpMessageConverter<?> converter = fastConverter;  
  
 return new HttpMessageConverters(converter);  
 }  
}