链滴

FastJson 解决 long 类型在前端界面展示精度丢失问题

作者：kangaroo1122

原文链接：https://ld246.com/article/1630569973733

来源网站：链滴

前后端交互的时候，数据的ID字段采用的雪花ID，Long类型，返回给前端时，由于数值过大，会导精度丢失，后面几位会变成0，这时候就需要把ID字段转成String类型的返回给前端页面。

目前有三种方式可以实现这个功能：

方式一、@JSONField

采用@JSONField注解，在ID字段上加上如下的注解，即可返回前端字符串的ID数据

```
@JSONField(serializeUsing = ToStringSerializer.class)
```

缺点：很明显，每个返回实体 model，只要有ID，就需要添加注解，太过于繁琐

方式二、Long.class转成String.class

在全局配置中，将Long类型的字段转成String类型

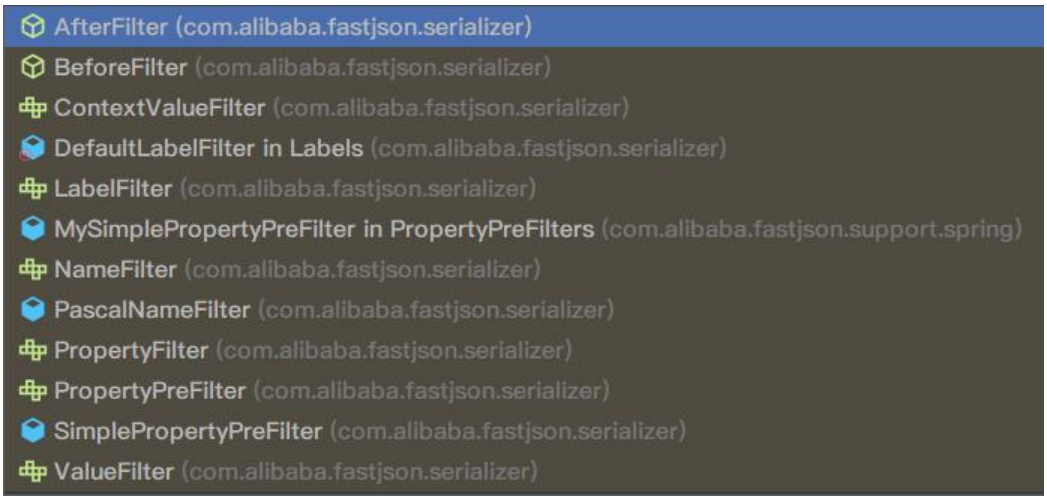
```
@Configuration
public class CustomFastJsonConfig {
    @Bean
    FastJsonHttpMessageConverter fastJsonHttpMessageConverter() {
        //1.需要定义一个convert转换消息的对象
        FastJsonHttpMessageConverter converter = new FastJsonHttpMessageConverter();

        //2.添加fastJson的配置信息
        FastJsonConfig fastJsonConfig = new FastJsonConfig();
        //3.设置Long为字符串
        SerializeConfig serializeConfig = SerializeConfig.globalInstance;
        serializeConfig.put(Long.class, ToStringSerializer.instance);
        serializeConfig.put(Long.TYPE, ToStringSerializer.instance);
        fastJsonConfig.setSerializeConfig(serializeConfig);
```

```
        //4.在convert中添加配置信息.
        converter.setFastJsonConfig(fastJsonConfig);
        return converter;
    }
}
```

缺点：也很明显，后端返回的数据，只要是Long类型的字段，都会被转成String返回，导致扩大了转范围

方式三、SerializeFilter

利用FastJson内置的SerializeFilter，有很多，如下



- PropertyPreFilter 根据PropertyName判断是否序列化；
- PropertyFilter 根据PropertyName和PropertyValue来判断是否序列化；
- NameFilter 修改Key，如果需要修改Key，process返回值则可；
- ValueFilter 修改Value；
- BeforeFilter 序列化时在最前添加内容；
- AfterFilter 序列化时在最后添加内容。

这里主要用到的SerializeFilter为ValueFilter，如下：

```
@Configuration
public class CustomFastJsonConfig {
    @Bean
    FastJsonHttpMessageConverter fastJsonHttpMessageConverter() {
        //1.需要定义一个convert转换消息的对象
        FastJsonHttpMessageConverter converter = new FastJsonHttpMessageConverter();

        //2.添加fastJson的配置信息
        FastJsonConfig fastJsonConfig = new FastJsonConfig();
        //3.设置id字段为字符串
        fastJsonConfig.setSerializeFilters((ValueFilter) (object, name, value) -> {
            if ("id".equalsIgnoreCase(name)){
                return value + "";
            }
```

```
            return value;
        });

        //4.在convert中添加配置信息.
        converter.setFastJsonConfig(fastJsonConfig);
        return converter;
    }
}
```

很明显，这个处理方式是最好的，这里是将ID字段转成String，需要转换其他字段时，只需要新增相的逻辑判断即可

完美解决。