Spring mvc在使用jackson将数据转换为json在前端显示时会存在时区问题。即：后端数据库记录时间为北京时间：2016-10-04 16:22:39，当我们将时区转换为UTC+2:00 开罗时间时 反馈到前端可能变成2016-10-04 08:22:39 时间根据时区进行调整。

解决方案：

**public** **class** JsonJodaDateTimeSerializer **extends** JsonSerializer<Date> {

@Override

**public** **void** serialize(Date value, JsonGenerator gen, SerializerProvider serializers)

**throws** IOException, JsonProcessingException {

SimpleDateFormat format = **new** SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

String formattedDate = format.format(value);

gen.writeString(formattedDate);

}

}

@JsonSerialize(using = JsonJodaDateTimeSerializer.**class**)

**private** Date jsonjodaDate;

可保证前后端数据一致

### 如果在后端根据时区修改时间值可以保证反馈到前端的时间戳一致，但是前端还是会根据时区来转换时间戳，以下是我不同时区后台处理时区的方式

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 北京时间【后端】  （UTC + 8：00） | 北京时间【前端】  （UTC + 8：00） | 【后端】  （UTC + 2：00） | 【前端】  （UTC + 2：00） |
| 时间不转换 | 1475569359633 | 2016-10-04 16:22:39 | 1475590959633 | 2016-10-04 16:22:39 |
| 时间根据时区转换一次 | 1475569359633 | 2016-10-04 16:22:39 | 1475569359633 | 2016-10-04 10:22:39 |

所以唯一要注意的是前后端时区是否一致，如果不一致则需要在前端或者后端处理时间。

前端处理方式：

var d = new Date(1475569359633);

//得到1970年一月一日到现在的秒数

var len = d.getTime();

//本地时间与GMT时间的时间偏移差

var offset = d.getTimezoneOffset() \* 60000;

//得到现在的格林尼治时间

var utcTime = len + offset;

var dd = new Date(utcTime + 3600000 \* 8);

后端处理方式：

SimpleDateFormat format = **new** SimpleDateFormat("yyyy-MM-dd-HH-mm-ss-SSS");

String[] timeAry = format.format(d).split("-");

Calendar calendar = Calendar.*getInstance*(TimeZone.*getTimeZone*("GMT+8"));

calendar.set(Integer.*parseInt*(timeAry[0]), Integer.*parseInt*(timeAry[1])-1,Integer.*parseInt*(timeAry[2]),

Integer.*parseInt*(timeAry[3]), Integer.*parseInt*(timeAry[4]), Integer.*parseInt*(timeAry[5]));

calendar.set(Calendar.***MILLISECOND***, Integer.*parseInt*(timeAry[6]));

Date cDate = calendar.getTime();

SOME BACKGROUD

Although GMT and UTC share the same current time in practice, there is a basic difference between the two:

* **[GMT is a time zone](https://www.timeanddate.com/time/zones/gmt)** officially used in some European and African countries. The time can be displayed using both the 24-hour format (0 - 24) or the 12-hour format (1 - 12 am/pm).
* **[UTC is not a time zone](https://www.timeanddate.com/time/aboututc.html)**, but a time standard that is the basis for civil time and time zones worldwide. This means that no country or territory officially uses UTC as a local time.

## UTC, GMT and Daylight Saving Time

Neither UTC nor GMT ever change for [Daylight Saving Time](https://www.timeanddate.com/time/dst/) (DST). However, some of the countries that use GMT switch to different time zones during their DST period.

某些情况下需要考虑DST带来的影响。

比如：1986年至1991年，中华人民共和国在全国范围实行了六年夏令时

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:SS");

String sTime = "1991-04-07 00:00:00";

Date time = sdf.parse(sTime);

Calendar cd = Calendar.getInstance();

cd.setTime(time);

cd.add(Calendar.DATE, 7);

time = cd.getTime();

System.out.println(sdf.format(time));

在运行后，没有得到预期的

1991-04-14 00:00:00

而是

1991-04-14 01:00:00

凭空多出来了1小时