# JsonCreator使用说明

## 原理介绍

实体转换JSON系统中最核心的功能,定义的表达式基本就是为了实体转换为JSON而设计的。怎么样去实现实体转JSON呢？这就涉及到JSON实质的问题，JSON是对象还是字符串？

**（1）设计思想**

在Javascript中这个问题不是很明显，因为它的eval可以直接到字符串转换为对象，但是在JAVA等面向对象的语言中，这个问题就很明显。如果是对象的话，那么需要先设计JSON的一些类，并通过这些类来构建JSON对象，然后实现实体与JSON对象之间的数据迁移。如果是字符串，那么就直接把实体对象转换为字符串。

在JAVA来讲，JSON的最重要作用就是数据传输的格式传输前台，这个传输肯定是字符串。而JSON对象其意义并不大，因为JAVA的操作一般都是基于更有实际的意义的实体来进行操作，而没有人愿意基于通用的JSON进行操作，这个操作更难不直观，而Javascript其实没有所谓的真正意义的对象，所有对象其实都是JSON对象，对于对象的操作也就是对JSON对象进行操作，这也是Javascript之所以难于使用的一个原因。

在Java中实体转换JSON实际上就是实体转换为JSON字符串，现行的JSON库的做法有两种，一种以JSON Lib为代表，标准的对象的思想，在类库中创建JSON的一些类，并根据这些类创建对象，把实体的属性及值迁移到JSON对象中去，最后再把JSON对象转换为JSON字符串。另外一种是jackson为代表，采用JSON字符串的思想，直接把实体转换为JSON字符串。所以其运行的速度是JSON Lib的百倍以上。

Jackson的缺点在于不能灵活地配置转换过程，也就是实体中有什么属性，基本上都得转换，对于采用AJAX的数据传输的应用系统来讲，这个是开发人员不能忍受的。而JSON Lib是可以进行配置转换过程，因为你需要什么样的属性，你就可以在实体转JSON对象时进行手动处理，但是这个处理的编码量大，性能低下，因为其能解决问题，所以即JSON Lib不怎么更新，还是有很多用户。

本文采用可配置表达式进行JSON转换库的开发，就是要结合两种方式的优点，抛弃它们的缺点，开发一个高性能且灵活的JSON转换库。需要高性能，那么就肯定不能采用JSON Lib的那种方式，只能采用直接把实体转换为JSON字符串，但是在转换过程需要预留接口让用户进行相关的配置，这个接口就是可配置的表达式，用户只需要配置好表达式之后，系统在转换过程解析表达式，直接进行JSON字符串的转换。

**（2）核心实现流程**

实现JSON对象转换为JSON字符串是一个较为复杂的嵌套算法，解析过程中即有其共性的处理，也需要进行特殊的处理。首先对JSON字符串来讲，其就是key/value对，每个value中又可以包括不同的key/value。但是对数组来讲，其内部不是key/value对，而是value。还有一种情况就是对最外层的不是key/value对，而是value。Value的形式有对象｛｝和数组[]两种形式，其解析的核心算法实现流程如下图4.3所示。

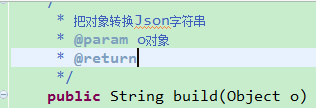


图4.3 实体转换Json核心流程

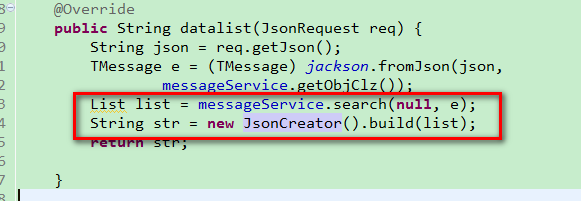
## 使用说明

### 2.1 build

#### 2.1.1 基本的build



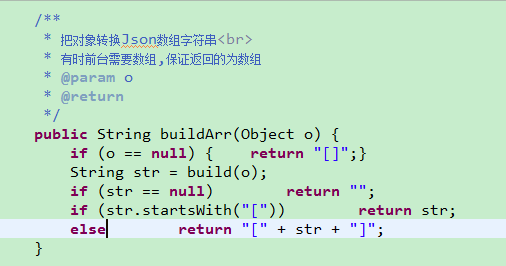
示例：

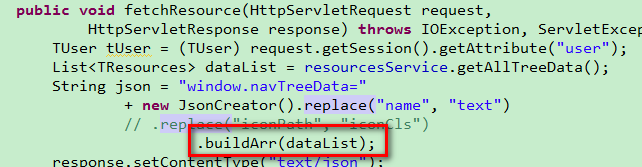


List list = messageService.search(null, e);

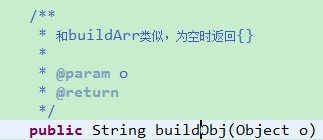
String str = new JsonCreator().build(list);

#### 2.1.2 buildArr

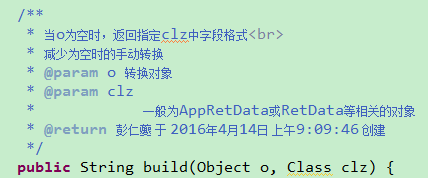




#### 2.1.3 buildObj

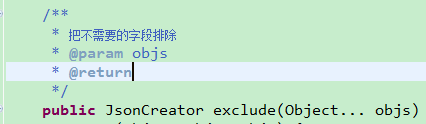


#### 2.1.3 build指定默认的类

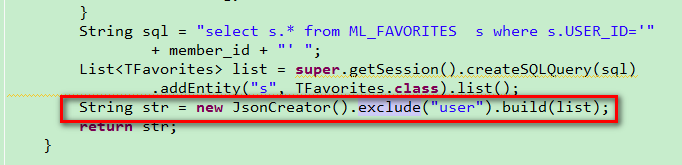


### 2.2 exclude

方法说明

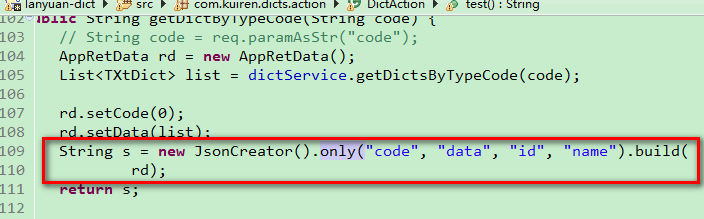


示例：



String str = new JsonCreator().exclude("user").build(list);

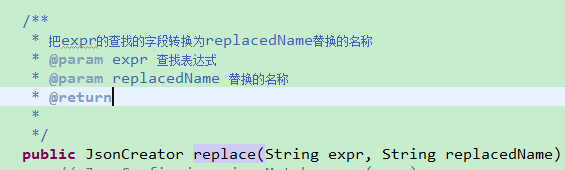
### 2.3 only



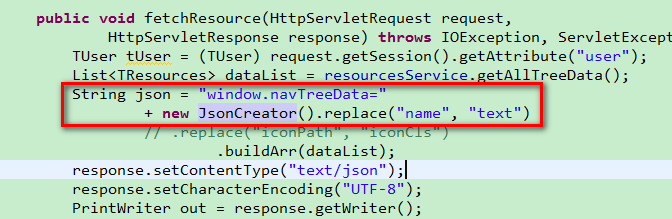
String s = **new** JsonCreator().only("code", "data", "id", "name").build(rd);

### 2.4 replace

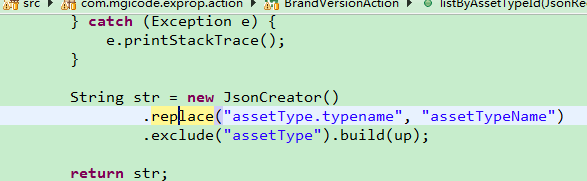
#### 2.4.1 replace基本



示例

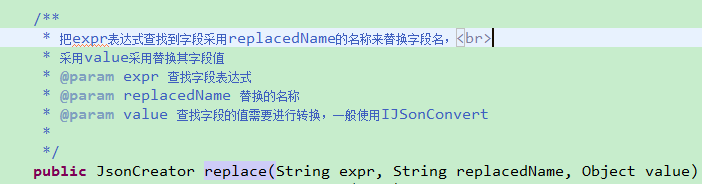


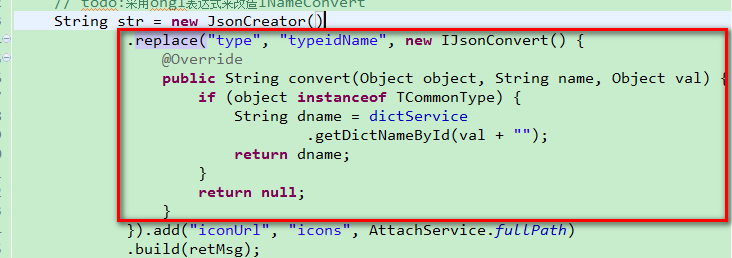
**new** JsonCreator().replace("name", "text").buildArr(dataList);

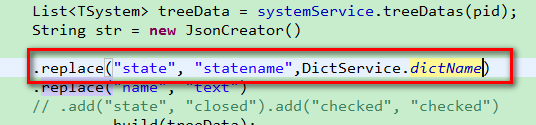


String str = **new** JsonCreator().replace("assetType.typename", "assetTypeName").exclude("assetType").build(up);

#### 2.4.2 replace的值可以动态转换



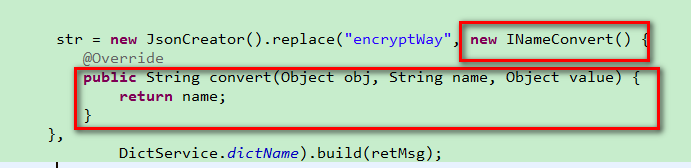




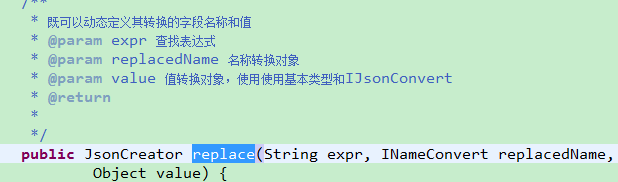
#### 2.4.3 replace名称可动态转换



示例：



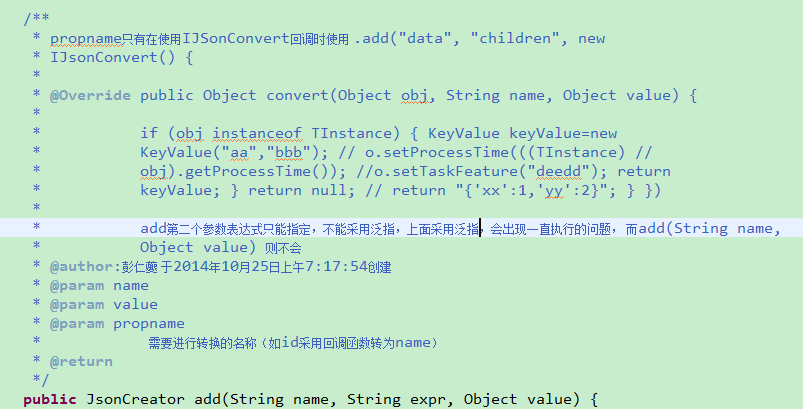
#### 2.4.4 replace名称和值都可动态转换

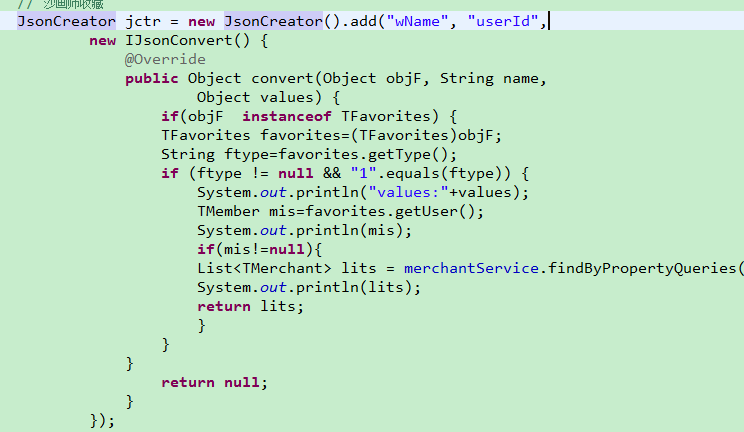


示例参考2.4.2和2.4.3

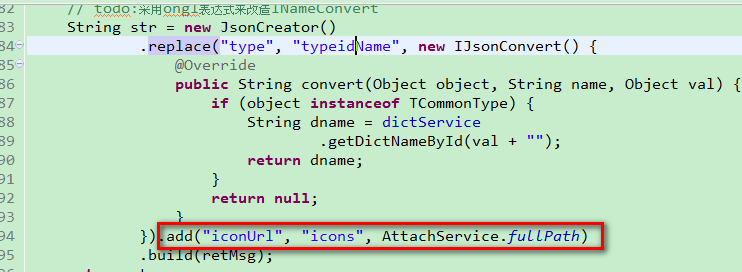
### 2.5 add

#### 2.5.1 动态值转换





#### 2.5.2 动态值转换简化版本



#### 2.5.3 动态直接值转换

有时需要在某个对象中加上其它的对象，直接采用2.5.1和2.5.2不行，转换时候会多“”，这时需要采用DirectConvert来封装一下。

