通常而言map映射都应用在被映射的key的值并不多的情况下，如性别，学历，成为等。

注：

在使用mapValue处理机制之前一定要看key的值是不是非常多，如果很多那就

一定不要用mapValue处理机制，而是应该采用DTO或其他方式解决这个问题。

之所以强调当“key”值“非常多”就不要使用mapValue，是因为Dorado7必须把所有的键值对全部下载到浏览器中才能完成key/value的映射。下载的值过多势必导致网络开销和客户端的运算量负担加重。通常来讲，如果您的界面只有一个地方需要使用mapValue的，那么一两百的键值对并不是是大的问题，但是如果您的一个界面上有十几个需要使用mapValue地方，那么您就必须综合的来考虑所有这些键值对加在一起对造成多大的网络开销和客户端的运算量。

mapValue的处理机制通常有三种方法：

* xml配置
* json赋值
* provider赋值

## 1、性别（xml赋值）

本例性别的映射没有直接配置在视图模型中，而是直接配置到DataSet对应的DataType声明模块中，在model文件中Employee的sex的mapValue声明：

<PropertyDef name="sex" label="性别">

<Property name="mapValues">

<Collection>

<Entity>

<Property name="key">true</Property>

<Property name="value">Male</Property>

</Entity>

<Entity>

<Property name="key">false</Property>

<Property name="value">Female</Property>

</Entity>

</Collection>

</Property>

</PropertyDef>

## 2、称谓（provider赋值）

与性别不一样，本例中称谓被设计为可以动态维护的信息，这样我们就无法通过<Property name="mapValues">这种固化到xml中的形态进行定义，必须采用一种更灵活的形态动态的从第三方存储中(例如数据库)获取。我们再来看称谓的定义，本例中在视图模型中重新覆盖了原始Employee这个DataType中titleOfCourtesy的定义：

<DataType name="EmployeeX" parent="Employee">

<PropertyDef name="titleOfCourtesy" mapValues=

'${dorado.getDataProvider("employeeInterceptor#getTitlesOfCourtesy").getResult()}'/>

</DataType>

其中的mapValues属性通过视图模型所支持的EL表达式通过一个DataProvider对象提供数据，通过dataProvider服务定位表达式，可以找到提供数据的服务方法：

@DataProvider

public Map<String, String> getTitlesOfCourtesy() {

Map<String, String> mapValue = new LinkedHashMap<String, String>();

mapValue.put("Mr.", "Mister");

mapValue.put("Mrs.", "Mistress");

mapValue.put("Ms.", "Miss");

mapValue.put("Dr.", "Doctor");

return mapValue;

}

## 3、json赋值

称为的mapValues也可以用json赋值的方式，如：

dataSet1.get("dataType").getPropertyDef("titleOfCourtesy")

.set("mapping",

[

{key:"Mr.", value:"Mister"},

{key:"Mrs.", value:"Mistress"},

{key:"Ms." value:"Miss"},

{key:"Dr.", value:"Doctor"}

]

);

## 特别说明

“key”，“value”关键字是mapValue处理机制中Dorado的默认关键字，当实际使用中我们可以通过PropertyDef的keyPropertyDef的keyProperty和valueProperty属性改变这个规则。

<PropertyDef name="sex" label="性别">

<Property name="keyProperty">key1</Property>

<Property name="valueProperty">value1</Property>

<Property name="mapValues">

<Collection>

<Entity>

<Property name="key1">true</Property>

<Property name="value1">Male</Property>

</Entity>

<Entity>

<Property name="key1">false</Property>

<Property name="value1">Female</Property>

</Entity>

</Collection>

</Property>

</PropertyDef>