目录

**[简介](#_Toc8111_WPSOffice_Level1)** **[2](#_Toc8111_WPSOffice_Level1)**

**[解析规则](#_Toc17291_WPSOffice_Level1)** **[2](#_Toc17291_WPSOffice_Level1)**

**[标识符&对应作用](#_Toc13440_WPSOffice_Level1)** **[2](#_Toc13440_WPSOffice_Level1)**

**[支持类型&Main中对应实例函数](#_Toc295_WPSOffice_Level1)** **[2](#_Toc295_WPSOffice_Level1)**

**[具体实现](#_Toc13004_WPSOffice_Level1)** **[3](#_Toc13004_WPSOffice_Level1)**

**[基础类型&字符串](#_Toc13442_WPSOffice_Level1)** **[4](#_Toc13442_WPSOffice_Level1)**

**[list枚举类型](#_Toc4893_WPSOffice_Level1)** **[4](#_Toc4893_WPSOffice_Level1)**

**[map枚举类型](#_Toc5382_WPSOffice_Level1)** **[5](#_Toc5382_WPSOffice_Level1)**

**[object枚举类型](#_Toc6995_WPSOffice_Level1)** **[7](#_Toc6995_WPSOffice_Level1)**

**[ARRAY\_TYPE枚举类型](#_Toc29887_WPSOffice_Level1)** **[9](#_Toc29887_WPSOffice_Level1)**

## 

## 简介

自定义通用解析规则，适用于游戏中的各种数值配置，主要为策划提供新的配置方式，进而解决json配置的臃肿和不方便。

## 解析规则

使用了java的反射机制，根据配置中字符串的位置反射为对应对象的对象成员，再根据不同标识符解析为对应的类型，各种组合对象之间递归调用。

## 标识符&对应作用

- 对象成员分隔符，格式 1-2

, array和list分隔符，格式1,2,3....

{} 对象分隔符，在最外围的可以不加，组合对象需加，格式{1-2-3}

: ; map类型使用，格式 key:value1;key:value2;

@ 多态分隔符，格式 XXX@{1-2-3}

## 支持类型&Main中对应实例函数

- 单对象解析 initSingle

- 对象数组解析 initArray

- 成员为对象 initMemberObject

- 多个成员为对象 initMemberObjects

- 数组成员为数组List initMemberList

- 成员为对象数组 initMemberArrObject

- 成员为对象数组，并且list里边的对象再包含一个list initMemberArrObjectList

- 数组成员为map initMemberMap

- 数组成员为map map.value为Object initMemberMapObjectMap

- 数组成员为map map.value为数组 initMemberMapArr

- 数组成员为map map.value为对象数组 并且其中的对象还会再嵌套一层 initMemberMapArr3

- 数组成员为map map.value为单对象 initMemberMapObject

- 数组成员为map map.value为数组对象 initMemberMapObjects

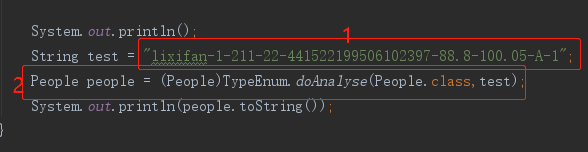
- 成员为array数组[] 基础类型 initArrayMember

- 成员为array数组[] 数组对象 initArrayObjectMember

- 多态 initDuotai

- 多态 initDuotai2

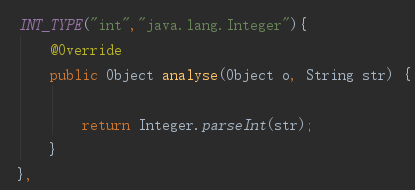
## 具体实现



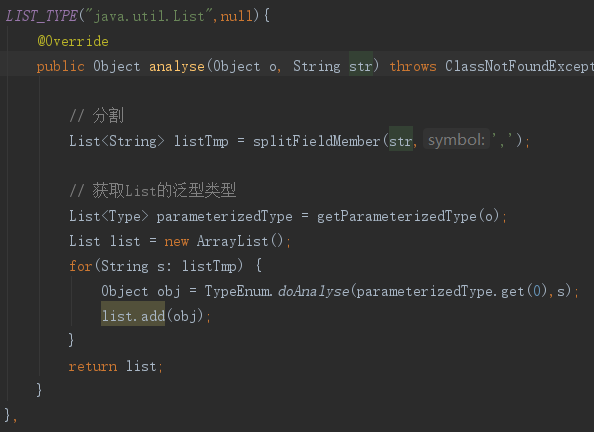
test字符串中 - 就是类中字数据成员的分割符，如lixifan对应着People类中的。

枚举TypeEnum支持八个基础类型和list、map以及Object和array，如：

### 基础类型&字符串

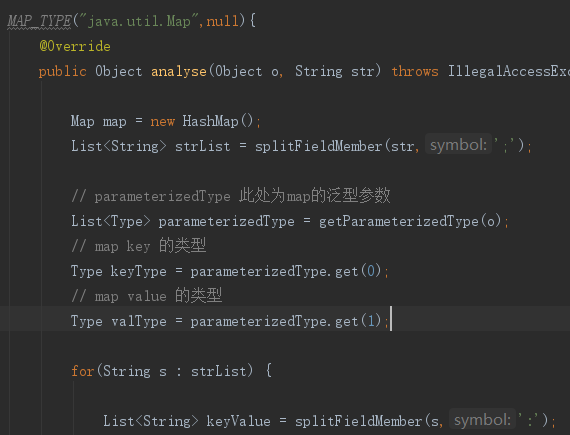


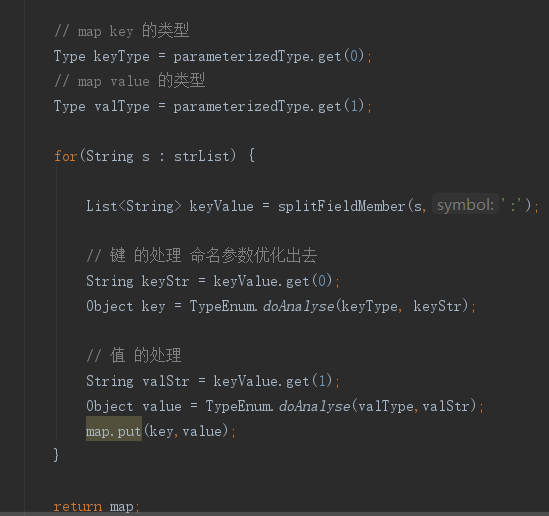
### list枚举类型



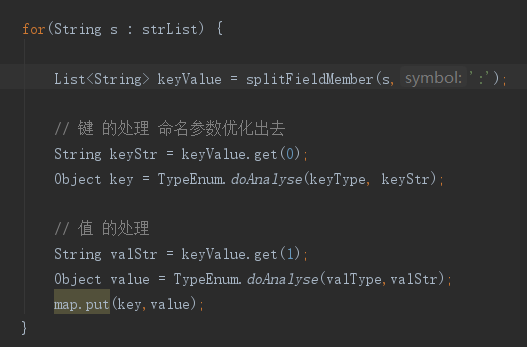
list和数组用 , 来处理，此处在分割好后，会根据List的泛型类型后调用TypeEnum.doAnalyse递归调用，再将拿到的object塞进list中，如List<Integer>拿到其中的Intege类型。

### map枚举类型





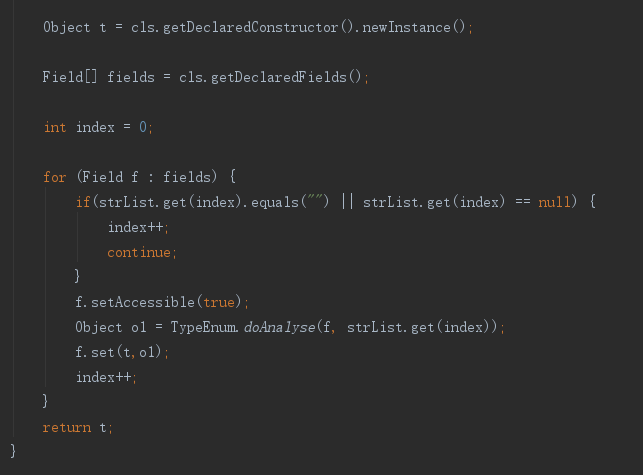
首先取出map的泛型参数类型，比如Map<Integer,Integer>这样，拿到其中key的类型和value的类型。



之后便是拆分然后根据各自的类型调用Type.doAnalyse拿到对应的Object，再放入map中。

### object枚举类型

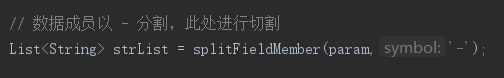




OBJECT\_TYPE的枚举实现会复杂一点

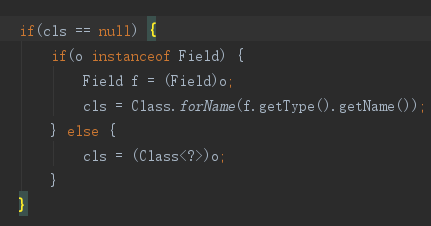


此处会优先处理掉带@的多态情况，取得多态的Class



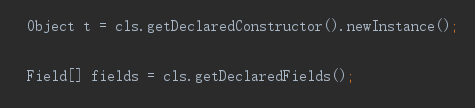
之后便是用 - 分割符对param进行拆分

如果不是多态的情况



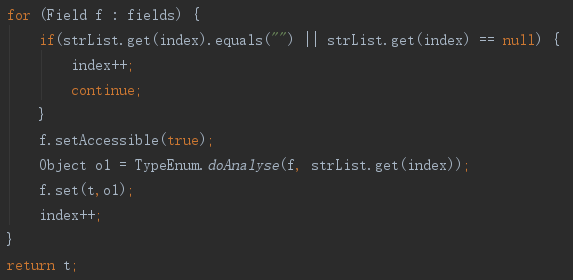
如果传过来的参数o是Field，则取出对应的Class类型，如果是直接传递class类型，则直接赋值。

再之后是反射实例化话后取出所有的Field

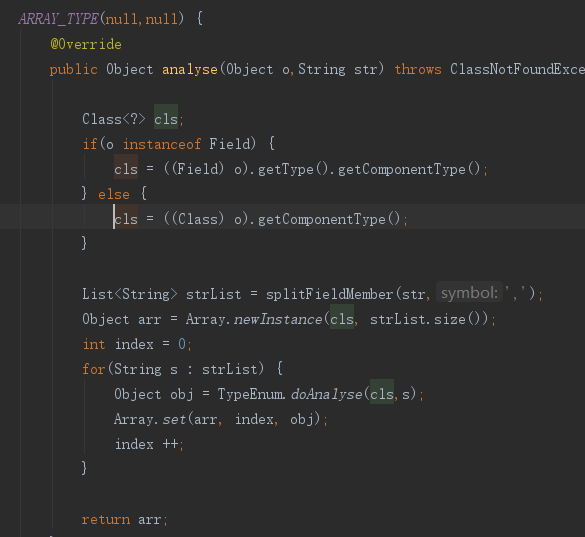


再接着便是遍历Fields，然后取出来后会先做一次判断，如果对应的Field要塞入的值是””或者为null，则放空，如果不是则调用指定的TypeEnum.doAnalyse拿到处理后的Object，

再塞入对应的Field中。



### **ARRAY\_TYPE枚举类型**



第一步是为了兼容People14[].class数组类和成员Field两种类型，之后便取出对应的ComponentType，即拿出PeoPle14这个类型，然后由于数组是用 , 进行划分的，所以会先根据 , 进行分割，之后便是调用Array.newInstance 实例化一个数组，然后遍历配置字符串的时候一个个根据类型获得递归调用TypeEnum.doAnalyse获得对应的Object，然后再set入数组中返回。