
回到首页

芋道源码 —— 知识星球

我是一段不羁的公告！
记得给芳芳这 3 个项目加油，添加一个 STAR 噢。
https://github.com/YunaiV/SpringBoot-Labs
https://github.com/YunaiV/onemall
https://github.com/YunaiV/ruoyi-vue-pro
2018-01-23
数据库实体设计

数据库实体设计 —— 商品（1.2）之商品详情

> 芳芳目前正在做一个开源的电商项目，胖友可以 star 下。
> https://gitee.com/zhijiantianya/onemall

1. 概述

本文主要分享商品模块的商品详情的数据库实体设计。

基于 有赞云提供的商品API 和 有赞微商城的商品管理 逆向猜测数据库实体。

> 【护脸旁白】
> 笔者非电商行业出身 && 非有赞工程师，所以有错误或不合理的地方，烦请斧正和探讨。
> 有赞是个各方面都很 NICE 的公司，推荐 。

2. 背景了解

参考 《如何使用商品页模版》 。

当我们打开某个商品时，商品详情页示意图如下：

基本信息区：显示商品主页、价格等信息
顶部模板内容
商品详情区，每个商品独立编辑
底部模板内容

所以商品详情页涉及商品的详情和商品引用的模板。

2.1 界面

1. 商城端-商品详情页

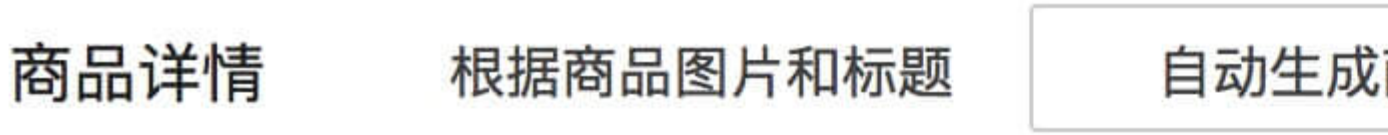
2. 商城端-商品页模板编辑页

3. 商城端-商品详情编辑页

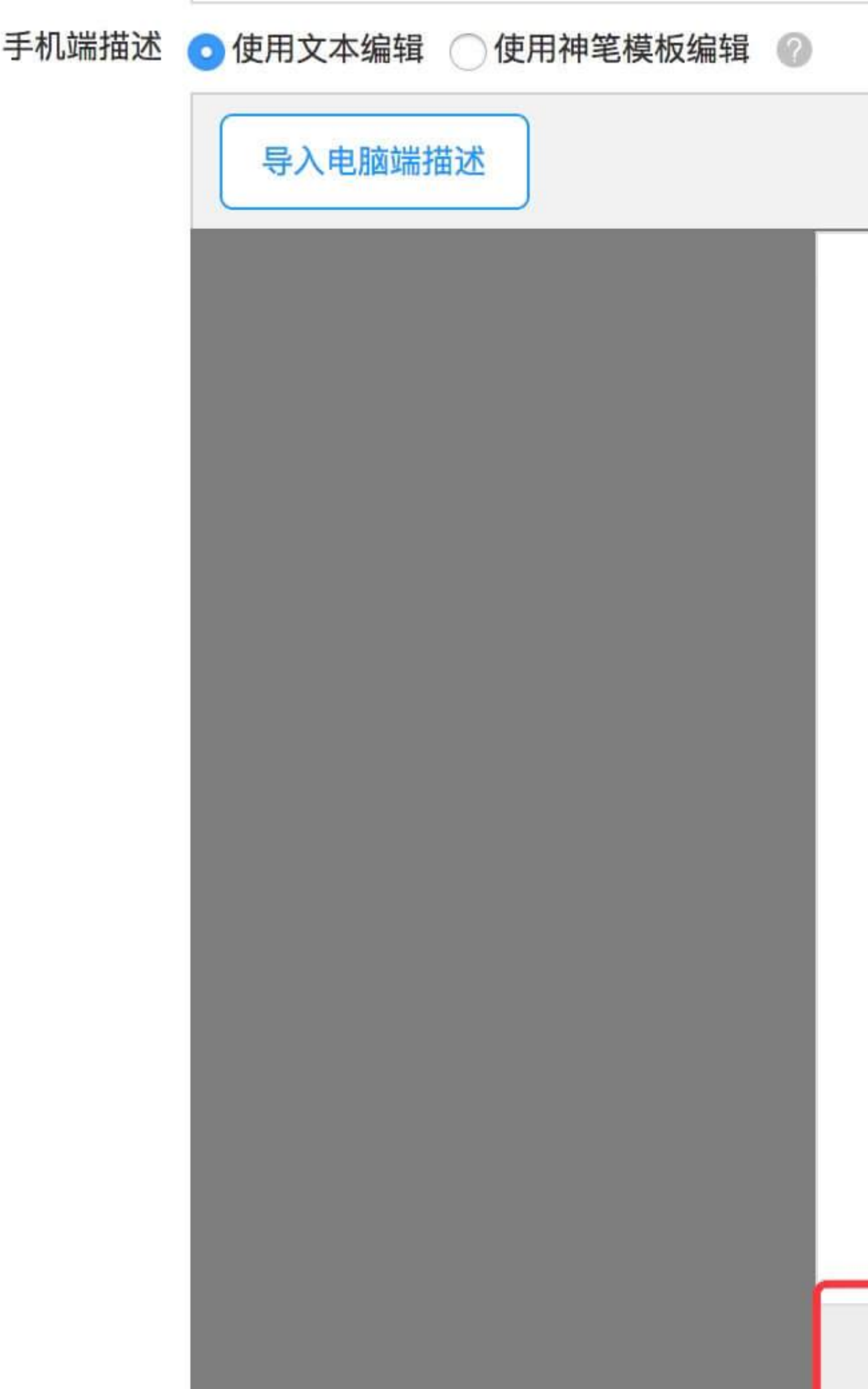
我们可以看出，有赞在常用的富文本的组件上，增加了大量易用的组件，提升商家运营的能力和空间。这点笔者觉得，非常值得借鉴与学习。

良心如笔者，如下是其他电商平台的商品详情编辑页：

微店

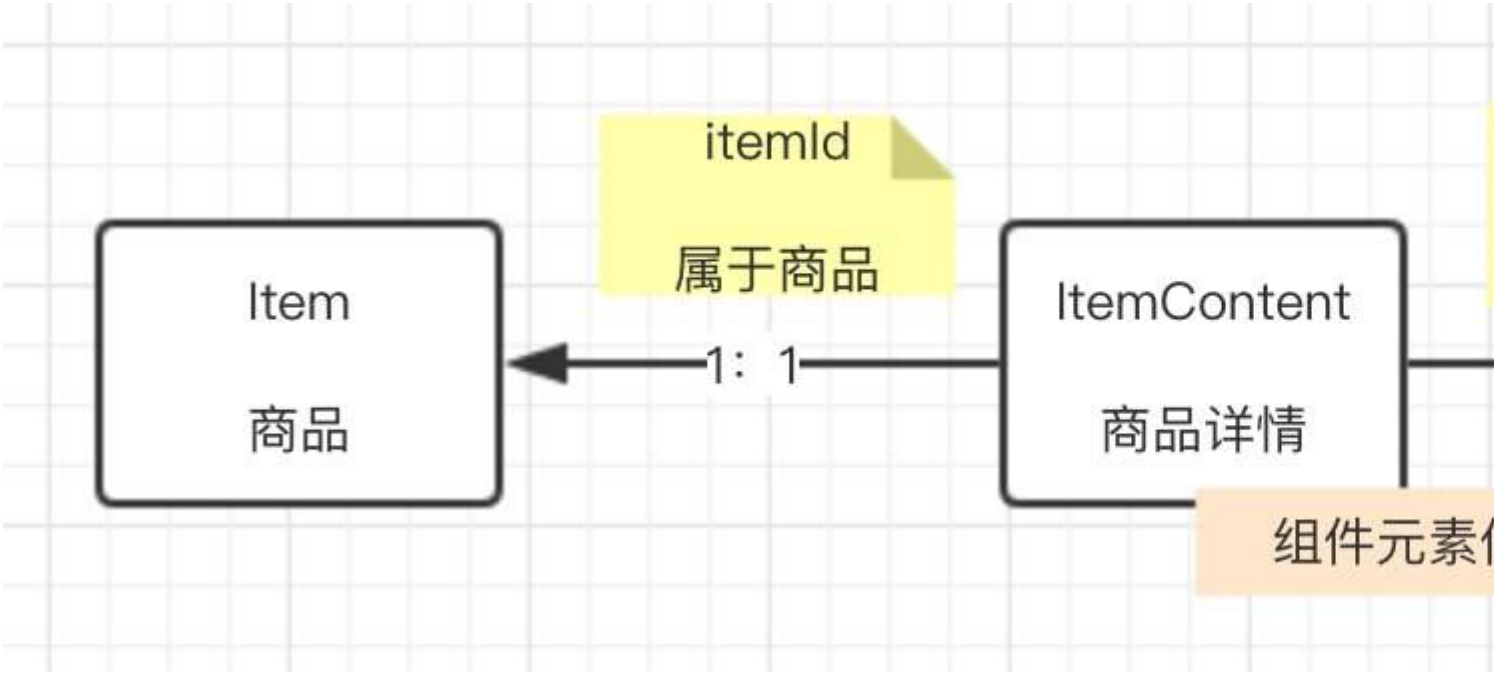


淘宝



3. 数据库实体

整体实体类关系如下图：



全部实体在 [Github 商品实体目录]() 下可见。

> 基情提示：
> 涉及的组件的 VO 类较多，可能比较像流水账。
> 胖友可以快速浏览，后面看看感兴趣的组件的 VO 类。

3.1 ItemTemplate

[ItemTemplate]()，商品页面模板。

```
/**
 * 编号 —— 普通版（内置）
 */
private static final Integer ID_STYLE_1 = - 1;
/**
 * 编号 —— 普通简洁版（内置）
 */
private static final Integer ID_SYTLE_2 = -2;

/**
 * 模板编号
 */
private Integer id;
/**
 * 别名
 *
 * 系统生成，作为唯一标识。例如，2fpa62tbms19h
 */
private String alias;
/**
```

```
     * 店铺编号
     */
    private Integer shopId;
    /**
     * 标题
     */
    private String title;
    /**
     * 是否展示
     */
    private Boolean isDisplay;
    /**
     * 状态
     *
     * 1-正常
     * 2-删除
     */
    private Integer status;
    /**
     * 模板元素数组
     *
     * JSON 格式，数组，每个元素为 {@link cn.iocoder.doraemon.itemgroup.item.entity.vo.AbstractItemTemplateElementVO} 的实
     */
    private String data;
    /**
     * 添加时间
     */
    private Date addTime;
    /**
     * 最后更新时间
     */
    private Date updateTime;
    /**
     * 删除时间
     */
    private Date deleteTime;
```

id ， 模板编号。系统内置了两套基础的模板：

- ID_STYLE_1 ， 普通版

- ID_SYTLE_2 ，普通简洁版

shopId ，店铺编号。每个店铺可配置属于自己的模板。
isDisplay ，是否展示。目前猜测（ 目前在接口上看到有该字段，管理页面里面实际不存在相关操作 ），可能内置的模板不展示。
status ，状态，包含正常和删除。笔者的习惯，对于未来可能出现新的状态的情况，笔者使用 Integer 而不 Boolean ，例如此处可能出现关闭状态（ status = 3 ）。
data ，模板元素数组，JSON 格式化成字符串（ 实际无根据元素进行检索，所以可以这么做 ）。其中，每个元素为 AbstractItemTemplateElementVO 的子类。

3.1.1 AbstractItemTemplateElementVO

AbstractItemTemplateElementVO ，商品页面模板元素 VO 抽象类。

```
/**
 * 类型
 */
private String type;
```

type ，类型。每种模板元素对应一个 type 枚举。笔者在实际抓取接口中发现，type = config 的情况，实际会对应多种模板元素，并且实际无共同性，所以笔者将它们分成两类：

- 通用组件：例如

- 模块独有组件：例如

友情提示
通用组件，在 「3.3 AbstractItemTemplateElementVO」 分享
模块独有组件，在各自模块的文章小节里分享。

3.1.2 ConfigElementVO

ConfigElementVO ，配置元素 VO 。

```
private static final String TYPE = "config";

/**
 * 标题
 */
private String title;
/**
 * 模板样式
 *
 * 0-普通版
 * 1-普通简洁版
 */
private Integer templateStyle;
```

type = config 。
title ，标题。和 ItemTemplate 的 title 字段一致。
templateStyle ，模板样式。

3.1.3 GoodsTemplateSplitElementVO

GoodsTemplateSplitElementVO ，商品详情区元素 VO 。

```
private static final String TYPE = "goods_template_split";
```

type = goods_template_split 。
作为商品详情区的占位元素。

3.2 ItemContent

ItemContent ， 商品详情。

```
/**
 * Item 编号
 *
 * {@link Item#id}
 */
private Integer id;
/**
 * 商品描述。
 *
 * 字数要大于5个字符，小于25000个字符 ， 受违禁词控制
 */
private String desc;
/**
 * 商品页模板编号
 */
private Integer templateId;
/**
 * 模板元素数组
 *
 * JSON 格式，数组，每个元素为 {@link cn.iocoder.doraemon.itemgroup.item.entity.vo.AbstractItemTemplateElementVO} 的实
 */
private String data;
```

id ， Item 编号，1：1 指向对应的 Item 。
desc ， 商品描述，HTML 富文本。考虑到数据库性能，单独成 ItemContent 表。
templateId ， ItemTemplate 编号。
data ， 模板元素数组，JSON 格式化成字符串(实际无根据元素进行检索，所以可以这么做)。其中，每个元素为 AbstractItemTemplateElementVO 的子类。

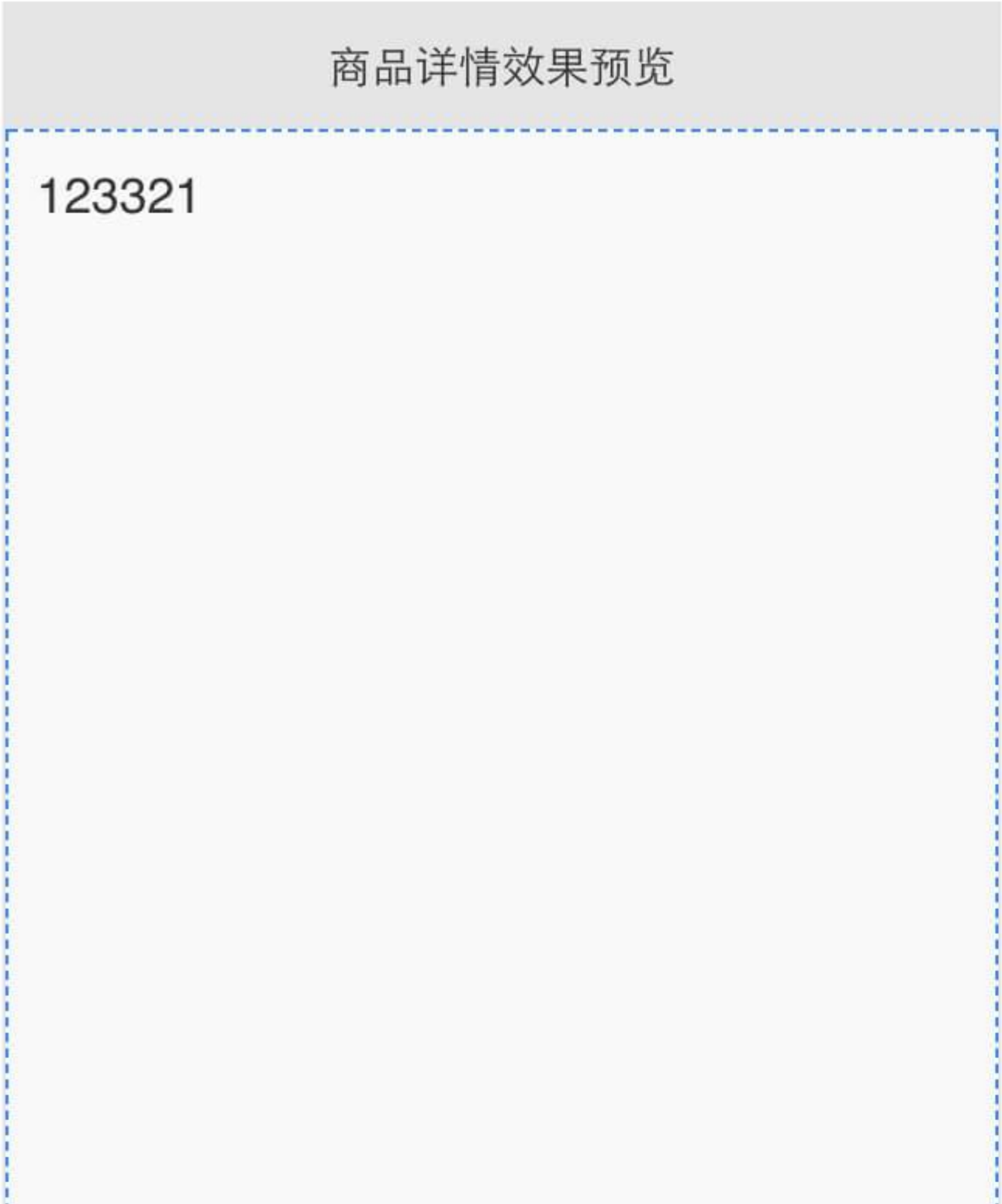
3.2.1 ConfigElementVO

ConfigElementVO ，配置元素 VO 。

```
private static final String TYPE = "config";

/**
 * 商品描述
 */
private String content;
```

type = config 。
content ，标题。和 ItemContent 的 desc 字段一致。
富文本编辑器基于 ueditor 定制。



3.3 AbstractItemTemplateElementVO

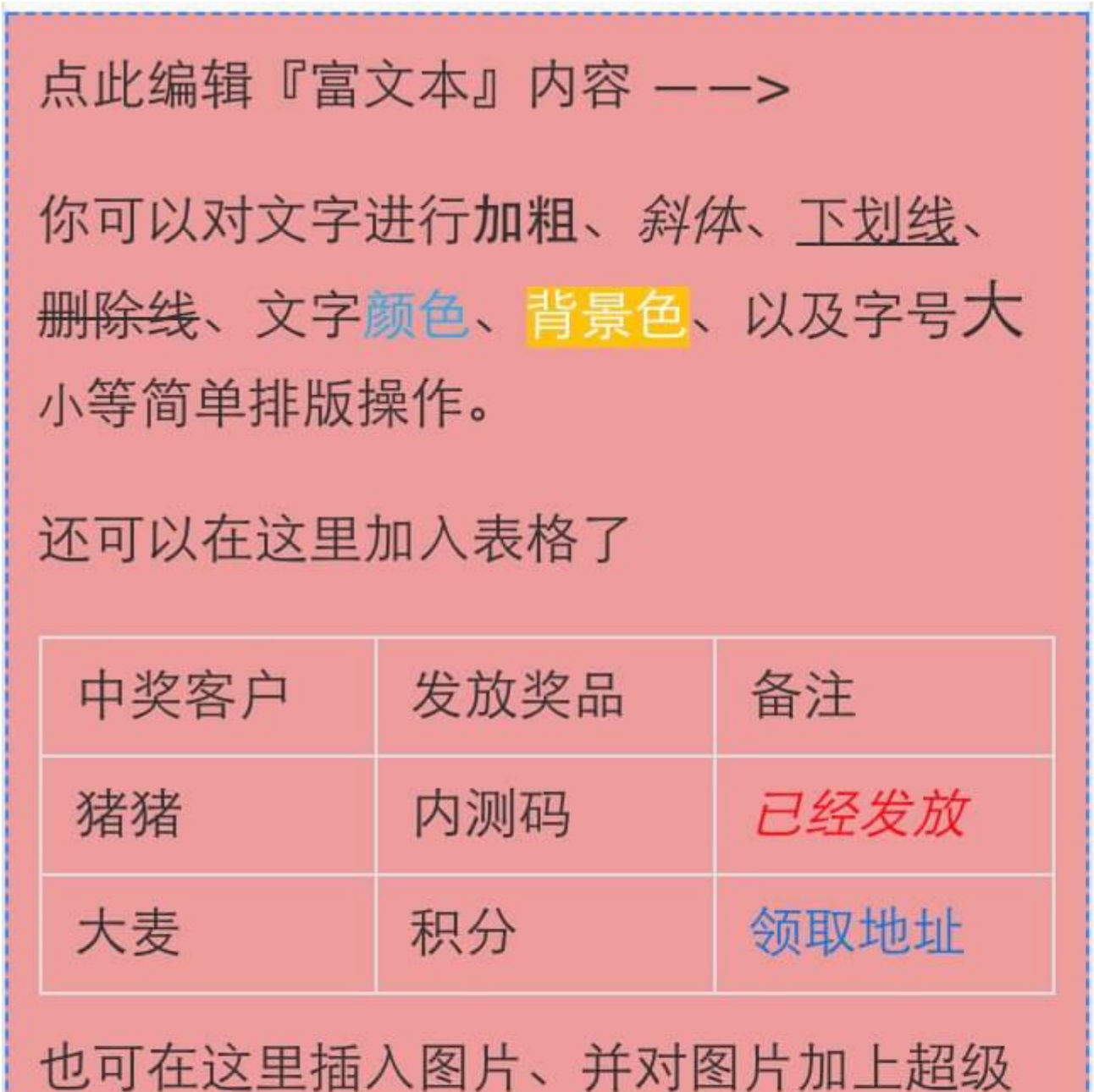
3.3.1 ItemTemplateRichTextElementVO

ItemTemplateRichTextElementVO ， 富文本元素 VO 。

```
private static final String TYPE = "rich_text";

/**
 * 是否全屏
 */
private Boolean fullscreen;
/**
 * 富文本
 */
private String content;
/**
 * 背景色
 */
private String color;
```

type = rich_text 。



3.3.2 ItemTemplateImageAdElementVO

ItemTemplateImageAdElementVO ，图片广告元素 VO 。

```
/**
 * 显示方式
 *
 * 0-折叠轮播
 * 1-分开显示
 */
private Integer showMethod;
/**
 * 显示大小
 *
 * 0-大图
 * 1-小图
 */
private Integer size;
/**
 * 元素高度
 */
private Integer height;
/**
 * 选项元素数组
 *
 * JSON 格式，数组，每个元素为 {@link SelectionElementVO}
 */
private String subEntrys;
```

type = rich_text 。

subEntrys ，SelectionElementVO 数组，JSON 格式化成字符串。代码如下：

- linkTitle 等为冗余字段。验证测试过程如下：1）选择一个商品作为 SelectionElementVO ；2）修改商品的标题；3）SelectionElementVO 的链接标题未变。
  - 当然，实际上也没必要更新，因为页面不展示这些字段对应的内容。
  - 另外，下面我们看到的 SelectionElementVO 也是相似情况。

3.3.3 ItemTemplateCubeElementVO

ItemTemplateCubeElementVO ，魔方元素 VO 。

```
private static final String TYPE = "cube2";

/**
 * 魔方高度
 */
private Integer layoutHeight;
/**
 * 魔方宽度
 */
private Integer layoutWidth;
/**
 * 选项元素数组
 *
 * JSON 格式，数组，每个元素为 {@link SelectionElementVO}
 */
private String subEntrys;
/**
 * 生成的魔方的 HTML 内容
 *
 * 例如：
 * <tr>
 *      <td class="not-empty cols-1 rows-3 " colspan="1" rowspan="3" data-index="0">
 *          <a href="https://h5.youzan.com/v2/showcase/feature?alias=akn9lyHRly"> <img src="https://img.yzcdn.cn/uplo
 *      </td>
 *      <td class="not-empty cols-2 rows-3 " colspan="2" rowspan="3" data-index="1">
 *          <img src="https://img.yzcdn.cn/upload_files/2018/01/17/FvK2lCbM-sk7lPJ57VwVrVb3AYzL.jpeg!730x0.jpg" />
 *      </td>
 *      <td class="not-empty cols-1 rows-3 " colspan="1" rowspan="3" data-index="2">
 *          <img src="https://img.yzcdn.cn/upload_files/2018/01/20/Fv-KKhg4sZ6Rrwve3_jiDVYQdMlO.jpg!730x0.jpg" />
 *      </td>
 * </tr>
 * <tr>
 *      <td id="sbtd" style="display:none;"></td></tr><tr><td id="sbtd" style="display:none;"></td>
 * </tr>
 * <tr>
 *      <td class="empty" data-x="0" data-y="3"></td>
 *      <td class="empty" data-x="1" data-y="3"></td>
 *      <td class="empty" data-x="2" data-y="3"></td><td class="empty" data-x="3" data-y="3"></td>
 * </tr>
 */
private String tableContent;
```

type = cube2 。

可以在魔方中添加指定大小的方块

subEntrys ， SelectionElementVO 数组，JSON 格式化成字符串。

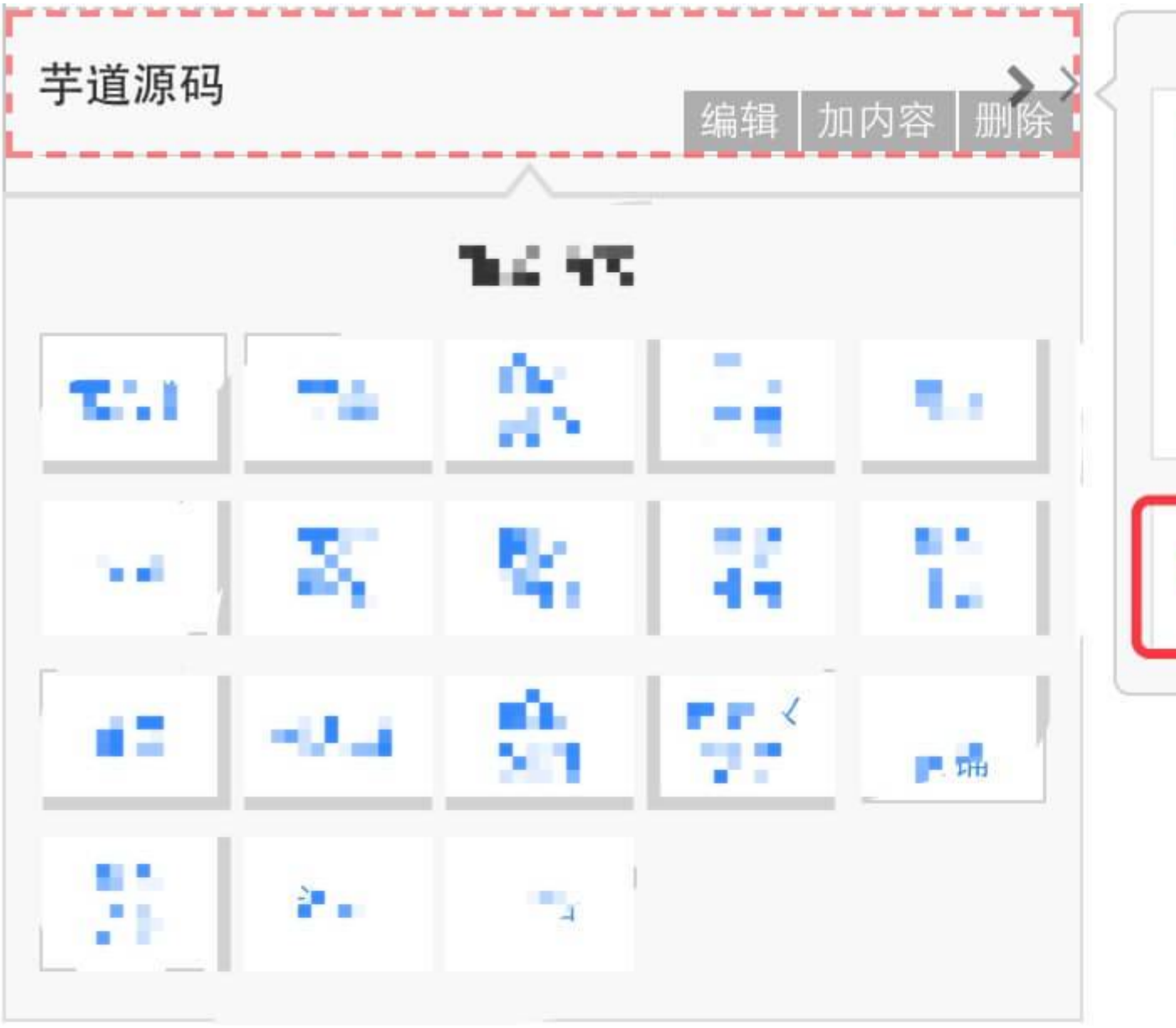
3.3.4 ItemTemplateTextNavElementVO

ItemTemplateTextNavElementVO ， 文本导航元素 VO 。

```
private static final String TYPE = "text_nav";

/**
 * 展示方式
 *
 * 0-默认（目前就这一种）
 */
private Integer showMethod;
/**
 * 选项元素数组
 *
 * JSON 格式，数组，每个元素为 {@link SelectionElementVO}
 */
private String subEntrys;
```

type = text_nav 。



subEntrys ， SelectionElementVO 数组，JSON 格式化成字符串。

3.3.5 ItemTemplateImageNavElementVO

ItemTemplateImageNavElementVO ，图片导航元素 VO 。

```
private static final String TYPE = "nav";

/**
 * 选项元素数组
 *
 * JSON 格式，数组，每个元素为 {@link ItemTemplateTextNavElementVO.SelectionElementVO}
 */
private String subEntrys;
```

type = nav 。

subEntrys ， SelectionElementVO 数组，JSON 格式化成字符串。

3.3.6 ItemTemplateShowCaseElementVO

ItemTemplateShowCaseElementVO ，橱窗元素 VO 。

```
private static final String TYPE = "showcase";

/**
 * 橱窗标题名
 */
private String title;
/**
 * 显示方式
 *
 * 0-保留
 * 1-三列
 */
private Integer mode;
/**
 * 图片间隙
 *
 * 0-保留
 * 1-消除
 */
private Integer withoutSpace;
/**
 * 内容区标题
 */
private String bodyTitle;
/**
 * 内容区说明
 */
private String bodyDesc;
/**
 * 选项元素数组
 *
 * JSON 格式，数组，每个元素为 {@link SelectionElementVO}
 */
private String subEntrys;
```

type = nav 。

subEntrys ， SelectionElementVO 数组，JSON 格式化成字符串。

3.3.7 ItemTemplateTitleElementVO

ItemTemplateTitleElementVO ， 商品页面模板标题元素 VO 。

```
private static final String TYPE = "title";

/**
 * 标题模板
 *
 * 1-传统样式
 *    使用 {@link #subEntrys} 字段
 * 2-模仿微信图文页样式
 *    使用 "wx_" 前缀的字段们
 */
private Integer titleTemplate;
/**
 * 标题名
 */
private String title;
/**
 * 副标题
 */
private String subTitile;
/**
 * 展示方式
 *
 * 0-居左显示
 * 1-居中显示
 * 2-居右显示
 */
private Integer showMethod;
/**
 * 背景色
 */
private String color;
/**
 * 选项元素数组
 *
 * JSON 格式，数组，每个元素为 {@link SelectionElementVO}
 */
private String subEntrys;
/**
 * wx 作者
 */
private String wxTitleAuthor;
/**
 * wx 日期
 */
private Date wxTitleDate;
/**
 * wx 链接标题
 */
private String wxTitleLink;
/**
 * wx 标题链接类型
 *
 * 0-引导关注
```

```
 * 1-其他链接
 */
private Integer wxTitleLinkType;
/**
 * wx 链接
 */
private WxLink wxLink;
```

type = title 。

该元素有两种模板样式，根据 titleTemplate 判断样式：

- 传统样式，使用 title、subTitle、showMethod、color、subEntrys 等字段。

- 模仿微信图文页样式，使用 `title`、`wx_` 前缀等字段们。

[SelectionElementVO]
[WxLinkVO]

3.3.8 ItemTemplateAudioElementVO

[ItemTemplateAudioElementVO] ，语音元素 VO 。

```
private static final String TYPE = "audio";
```

```
/**
 * 样式
 *
 * 0-模仿微信对话样式
 * 1-简易音乐播放器
 */
private Integer style;
/**
 * 语音
 */
private String audio;
/**
 * 气泡头像
 */
private String avatar;
/**
 * 气泡位置
 *
 * 0-居左
 * 1-居右
 */
private Integer bubble;
/**
 * 是否循环
 *
 * 0-否
 * 1-是
 */
private Integer loop;
/**
 * 播放
 *
 * 0-暂停后再恢复播放时，从头开始
 * 1-暂停后再恢复播放时，从暂停位置开始
 */
private Integer reload;
/**
 * 标题
 */
private String title;
```

type = audio 。

该元素有两种模板样式，根据 style 判断样式：

- 模仿微信对话样式

- 简易音乐播放器

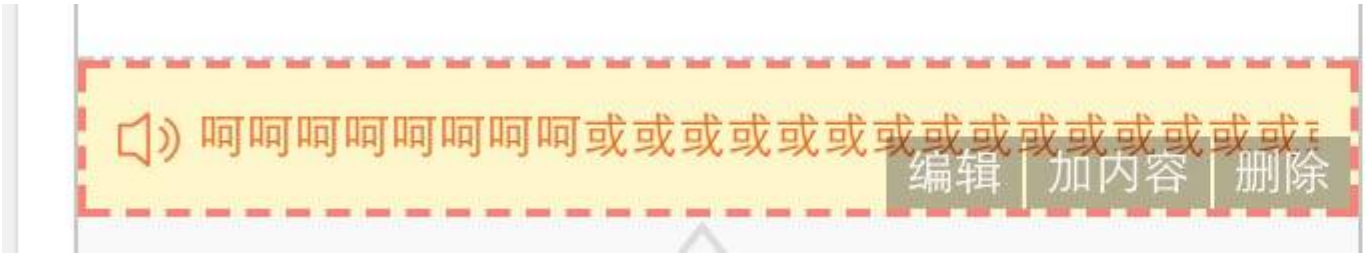
3.3.9 ItemTemplateNoticeElementVO

ItemTemplateNoticeElementVO ，通知元素 VO 。

```
private static final String TYPE = "notice";

/**
 * 内容
 */
private String content;
```

type = notice 。



3.3.10 ItemTemplateLineElementVO

ItemTemplateLineElementVO ， 商品页面模板辅助线元素 VO 。

```
private static final String TYPE = "line";

/**
 * 背景色
 */
private String color;
/**
 * 样式
 *
 * solid 实线
 * dashed 虚线
 * dotted 点线
 */
private String lineType;
/**
 * 是否左右留白
 */
private Boolean hasPadding;
```

type = line 。

3.3.11 ItemTemplateWhiteElementVO

ItemTemplateWhiteElementVO ， 辅助空白元素 VO 。

```
private static final String TYPE = "while";

/**
 * 空白高度，单位：像素
 */
private Integer height;
```

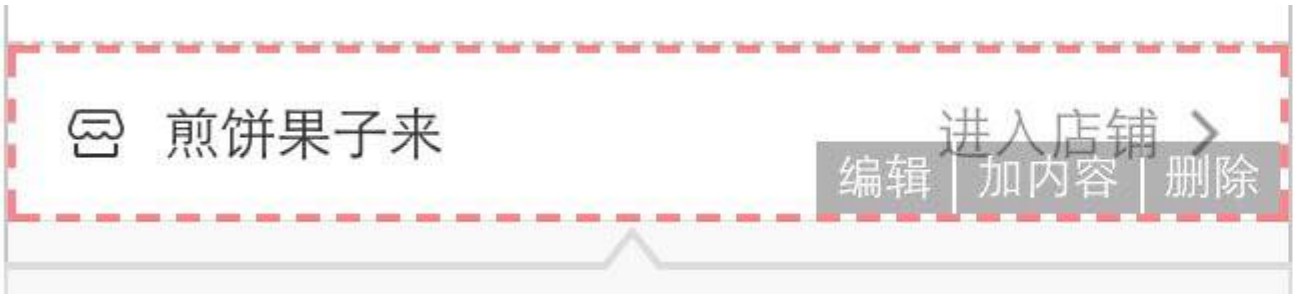
type = white 。

3.3.12 ItemTemplateStoreElementVO

ItemTemplateStoreElementVO ， 进入店铺元素 VO 。

```
private static final String TYPE = "store";
```

type = store 。



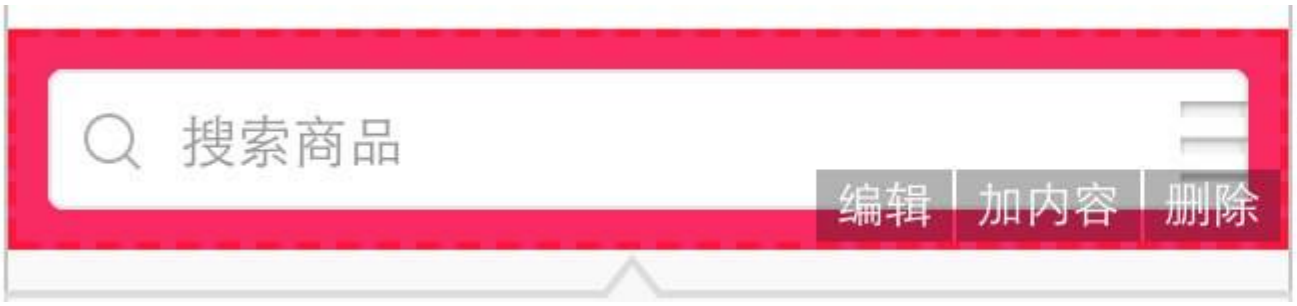
3.3.13 ItemTemplateSearchElementVO

ItemTemplateSearchElementVO ， 搜索商品元素 VO 。

```
private static final String TYPE = "search";

/**
 * 背景色
 */
private String color;
```

type = search 。



3.3.14 ItemTemplateGoodsElementVO

ItemTemplateGoodsElementVO ， 商品元素 VO 。

```
private static final String TYPE = "goods";

/**
 * 列表样式
 *
 * 0-大图
 * 1-小图
 * 2-一大两小
 * 3-详细列表
 */
private Integer size;
/**
```

```
 * 商品样式
 *
 * 0-卡片样式
 * 1-瀑布流
 * 2-极简样式
 * 3-促销
 */
private Integer sizeType;
/**
 * 是否显示购买按钮
 *
 * 0-不显示
 * 1-显示
 */
private Integer buyBtn;
/**
 * 显示购买按钮的样式
 *
 * 1-样式1
 * 2-样式2
 * 3-样式3
 * 4-样式4
 */
private Integer buyBtnType;
/**
 * 是否显示商品标题
 *
 * 0-不显示
 * 1-显示
 */
private Integer title;
/**
 * 是否显示商品简介
 *
 * 0-不显示
 * 1-显示
 */
private Integer showSubTitle;
/**
 * 是否显示商品价格
 *
 * 0-不显示
 * 1-显示
 */
private Integer price;
/**
 * 商品数组
 *
 * JSON 格式，数组，每个元素为 {@link Goods}
 */
private String goods;
```

`type = goods` 。

该元素有四种列表样式，根据 `size` 判断样式：

- 大图

- 小图


VV视界
VIP专享
乐视视频
芒果TV
优酷
爱奇艺
VIP小说
搜狐视频
脑子不太好使
￥18.80
Panda Gifts
熊猫有礼
熊猫有礼 临安新货山核桃仁 口口香酥脆，
￥88.00
编辑
加内容
删除
添加内容

- 一大两小


VV视界
VIP专享
优酷

- 详细列表

该元素有四种商品样式，根据 `sizeType` 判断样式。

- 此处笔者就不截图了。

`goods` ，Goods 数组，JSON 格式化成字符串。

- Goods 里有冗余字段，在商品更新后，冗余字段也会更新。验证测试过程如下：1）选择一个商品作为 Goods ；2）修改商品的图片；3）Goods 的图片变化。

3.3.15 ItemTemplateGoodsListElementVO

ItemTemplateGoodsListElementVO ，商品列表元素 VO 。

```
private static final String TYPE = "goods_list";

/**
 * 展示商品分组的商品数量
 */
private Integer goodsNumberType;
/**
 * 列表样式
 *
 * 0-大图
```

```
 * 1-小图
 * 2-一大两小
 * 3-详细列表
 */
private Integer size;
/**
 * 商品样式
 *
 * 0-卡片样式
 * 1-瀑布流
 * 2-极简样式
 * 3-促销
 */
private Integer sizeType;
/**
 * 是否显示购买按钮
 *
 * 0-不显示
 * 1-显示
 */
private Integer buyBtn;
/**
 * 显示购买按钮的样式
 *
 * 1-样式1
 * 2-样式2
 * 3-样式3
 * 4-样式4
 */
private Integer buyBtnType;
/**
 * 是否显示商品标题
 *
 * 0-不显示
 * 1-显示
 */
private Integer title;
/**
 * 是否显示商品简介
 *
 * 0-不显示
 * 1-显示
 */
private Integer showSubTitle;
/**
 * 是否显示商品价格
 *
 * 0-不显示
 * 1-显示
 */
private Integer price;
/**
 * 商品分组数组
 *
 * JSON 格式，数组，每个元素为 {@link GoodsTagElementVO}
 */
private String goods;
```

type = goods 。

整体字段类似 ItemTemplateGoodsElementVO 。

goods ，GoodsTagElementVO 数组，JSON 格式化成字符串。

- GoodsTagElementVO 里有冗余字段，在商品更新后，冗余字段不会更新。验证测试过程如下：1）选择一个商品分组作为 GoodsTagElementVO ；2）修改商品分组的名字；3）GoodsTagElementVO 的名字不变。
  - 当然，实际上也没必要更新，因为页面不展示这些字段对应的内容。
  - 另外，商品分组下的商品发生变化时，引用的内容会发生变化。

3.3.16-1 ItemTemplateGoodsTagListElementVO

ItemTemplateGoodsTagListElementVO ，商品分组元素 VO 的第一种。

```
private static final String TYPE = "tag_list";

/**
 * 商品分组数组
 *
 * JSON 格式，数组，每个元素为 {@link GoodsTagElementVO}
 */
private String goods;
```

type = tag_list 。

goods ，GoodsTagElementVO 数组，JSON 格式化成字符串。

- GoodsTagElementVO 里有冗余字段，在商品更新后，冗余字段会更新。验证测试过程如下：1）选择一个商品分组作为 GoodsTagElementVO ；2）修改商品分组的名字；3）GoodsTagElementVO 的名字改变。
  - 当然，实际上有必要更新，因为页面展示这些字段对应的内容。
  - 另外，商品分组下的商品发生变化时，引用的内容会发生变化。

3.3.16-2 ItemTemplateGoodsTagsElementVO

ItemTemplateGoodsTagsElementVO ，商品分组元素 VO 的第二种。

```
private static final String TYPE = "tags";

/**
 * 列表样式
 *
 * 0-大图
 * 1-小图
 * 2-一大两小
 * 3-详细列表
 */
private Integer size;
/**
 * 商品样式
 *
 * 0-卡片样式
 * 1-瀑布流
 * 2-极简样式
 * 3-促销
 */
private Integer sizeType;
/**
 * 是否显示购买按钮
 *
 * 0-不显示
 * 1-显示
 */
private Integer buyBtn;
/**
 * 显示购买按钮的样式
 *
 * 1-样式1
 * 2-样式2
 * 3-样式3
 * 4-样式4
 */
private Integer buyBtnType;
/**
 * 是否显示商品标题
 *
 * 0-不显示
 * 1-显示
 */
private Integer title;
/**
 * 是否显示商品简介
 *
 * 0-不显示
```

```
     * 1-显示
     */
    private Integer showSubTitle;
    /**
     * 是否显示商品价格
     *
     * 0-不显示
     * 1-显示
     */
    private Integer price;
    /**
     * 商品分组数组
     *
     * JSON 格式，数组，每个元素为 {@link GoodsTagElementVO}
     */
    private String goods;
```

type = tags 。

整体字段类似 ItemTemplateGoodsListElementVO 。

goods ，GoodsTagElementVO 数组，JSON 格式化成字符串。
    - GoodsTagElementVO 里有冗余字段，在商品更新后，冗余字段会更新。验证测试过程如下：1）选择一个商品分组作为 GoodsTagElementVO ；2）修改商品分组的名字；3）GoodsTagElementVO 的名字改变。
        - 当然，实际上有必要更新，因为页面展示这些字段对应的内容。
        - 另外，商品分组下的商品发生变化时，引用的内容会发生变化。

3.3.17 ItemTemplateLinkElementVO

ItemTemplateLinkElementVO ，关联链接元素 VO 的。

```
private static final String TYPE = "link";

/**
 * 选项元素数组
 *
 * JSON 格式，数组，每个元素为 {@link SelectionVO}
 */
private String subEntrys;
```

type = link 。

subEntrys ，SelectionVO 数组，JSON 格式化成字符串。
    - SelectionVO 里有冗余字段，在商品更新后，冗余字段不会更新。验证测试过程如下：1）选择一个商品分组作为 SelectionVO ；2）修改商品分组的名字；3）SelectionVO 的名字不变。
        - 当然，实际上也没必要更新，因为页面不展示这些字段对应的内容。
        - 另外，商品分组下的商品发生变化时，引用的内容会发生变化。

3.3.18 ItemTemplateComponentElementVO

ItemTemplateComponentElementVO ，自定义模块元素 VO 。

```
private static final String TYPE = "component";

/**
 * 自定义组件编号
 */
private Integer id;
/**
 * 标题
 */
private String title;
/**
 * 链接
 */
private String link;
```

type = component 。

自定义模块参见 《如何使用自定义页面模块？》 。功能上和 商品页面模板 类似，相对更加通用。

4. API

基于 有赞云提供的商品API ，整理如下 API 类。

4.1 ItemTemplateAPI

ItemTemplateAPI ，商品模板 API 。

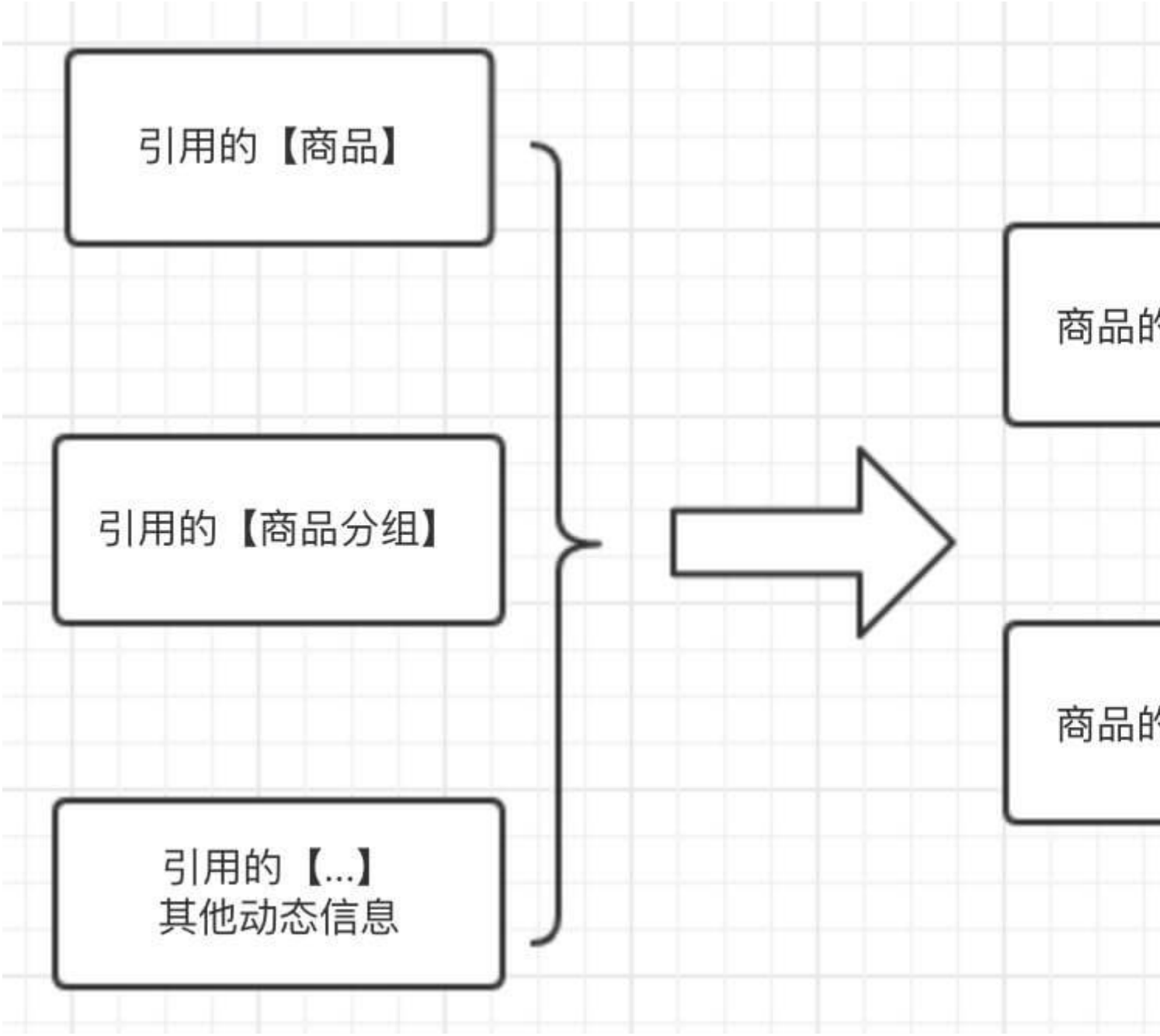
```
public interface ItemTemplateAPI {

    // 添加模板

    // 修改模板
```

5. 缓存

不同于我们常见的一些商品的描述，内容是静态 + 动态两部分组成。动态部分，例如引用的商品、商品分组等等，如下图所示：



TODO TODO 6004 ：商品详情缓存 。多层缓存，查询组成详情页。未来详细分享。

[《第七章 Web开发实战2——商品详情页》]

[《深入分布式缓存:从原理到实践》]

666. 彩蛋

有赞真是做的好细啊！写的手都累了！

关于 「5. 缓存」，欢迎胖友们一起讨论具体的实现方案。

文章目录

回到首页