IDCP-决策中心

系统使用说明书

# 版本信息

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 版本 | 修改日期 | 修订者 | 审核人 | 修改内容 |
| 1 | 1.0 | 2017-6-29 | 孟宇峰 |  | 初稿 |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
| 5 |  |  |  |  |  |
| 6 |  |  |  |  |  |

# 目录

[版本信息 2](#_Toc489531079)

[目录 3](#_Toc489531080)

[1. 引言 7](#_Toc489531081)

[1.1. 编写目的 7](#_Toc489531082)

[1.2. 读者对象 7](#_Toc489531083)

[1.3. 术语定义 7](#_Toc489531084)

[1.4. 参考资料 7](#_Toc489531085)

[2. 系统概述 8](#_Toc489531086)

[2.1. 项目背景 8](#_Toc489531087)

[2.2. 系统目标 8](#_Toc489531088)

[2.3. 条件与限制 8](#_Toc489531089)

[2.4. 推荐运行环境 8](#_Toc489531090)

[2.5. 推荐开发环境 8](#_Toc489531091)

[3. 系统登录 9](#_Toc489531092)

[3.1. 系统登录 9](#_Toc489531093)

[3.2. 系统首页 10](#_Toc489531094)

[4. 模型管理 10](#_Toc489531095)

[4.1. 模型查询 10](#_Toc489531096)

[4.2. 模型新增 11](#_Toc489531097)

[4.3. 模型维护-数据结构 12](#_Toc489531098)

[4.4. 模型维护-数据收集 13](#_Toc489531099)

[4.5. 模型维护-数据存储 14](#_Toc489531100)

[4.5.1. 查询页面 14](#_Toc489531101)

[4.5.2. 新增/维护页面 14](#_Toc489531102)

[4.6. 扩展参数 15](#_Toc489531103)

[4.6.1. 查询页面 15](#_Toc489531104)

[4.6.2. 新增/维护页面 16](#_Toc489531105)

[4.7. 数据项新增 17](#_Toc489531106)

[4.8. 数据项维护 19](#_Toc489531107)

[4.8.1. 数据结构 19](#_Toc489531108)

[4.8.2. 计算函数 20](#_Toc489531109)

[4.9. 数据项删除 21](#_Toc489531110)

[5. 数据收集工具 22](#_Toc489531111)

[5.1. JDBC工具 22](#_Toc489531112)

[5.2. Redis缓存工具 22](#_Toc489531113)

[6. 数据存储工具 23](#_Toc489531114)

[6.1. JDBC工具 23](#_Toc489531115)

[6.2. Redis缓存工具 23](#_Toc489531116)

[7. 计算函数 23](#_Toc489531117)

[7.1. 字符串类 23](#_Toc489531118)

[7.1.1. 时间字符串转格式函数-DateStringFormat 23](#_Toc489531119)

[7.1.2. 字符串相等比较函数-Equals 24](#_Toc489531120)

[7.1.3. 字符串不相等比较函数-NotEquals 24](#_Toc489531121)

[7.1.4. 字符串空判断函数-IsBlank 24](#_Toc489531122)

[7.1.5. 字符串非空判断函数-IsNotBlank 24](#_Toc489531123)

[7.1.6. 字符串替换函数-Replace 25](#_Toc489531124)

[7.1.7. 字符串分割函数-Split 25](#_Toc489531125)

[7.1.8. 字符串截取函数-SubStringAfter 25](#_Toc489531126)

[7.1.9. 字符串截取函数-SubStringAfterLast 26](#_Toc489531127)

[7.1.10. 字符串截取函数-SubStringBefore 26](#_Toc489531128)

[7.1.11. 字符串截取函数-SubStringBeforeLast 26](#_Toc489531129)

[7.1.12. 字符串截取函数-SubStringBetween 26](#_Toc489531130)

[7.1.13. 字符串截取函数-SubStringsBetween 27](#_Toc489531131)

[7.1.14. 字符串截取函数-SubString 27](#_Toc489531132)

[7.1.15. 字符串转布尔类型函数-ToBoolean 27](#_Toc489531133)

[7.1.16. 字符串转时间类型函数-ToDate 28](#_Toc489531134)

[7.1.17. 对象转字符串类型函数-ToString 28](#_Toc489531135)

[7.1.18. 字符串去空格函数-Trim 28](#_Toc489531136)

[7.1.19. 字符串去空格函数-TrimToEmpty 28](#_Toc489531137)

[7.2. 日期类 29](#_Toc489531138)

[7.2.1. 时间比较函数-After 29](#_Toc489531139)

[7.2.2. 时间比较函数-NotAfter 29](#_Toc489531140)

[7.2.3. 时间比较函数-Before 30](#_Toc489531141)

[7.2.4. 时间比较函数-NotBefore 30](#_Toc489531142)

[7.2.5. 时间比较函数-BetweenEquals 31](#_Toc489531143)

[7.2.6. 时间比较函数-NotBetweenEquals 31](#_Toc489531144)

[7.2.7. 时间比较函数-Between 32](#_Toc489531145)

[7.2.8. 时间比较函数-NotBetween 32](#_Toc489531146)

[7.2.9. 时间比较函数-CurrentDate 33](#_Toc489531147)

[7.2.10. 时间格式化函数-Format 33](#_Toc489531148)

[7.3. 数字类 33](#_Toc489531149)

[7.3.1. 数字比较函数-BetweenEquals 33](#_Toc489531150)

[7.3.2. 数字比较函数-NotBetweenEquals 33](#_Toc489531151)

[7.3.3. 数字比较函数-Between 34](#_Toc489531152)

[7.3.4. 数字比较函数-NotBetween 34](#_Toc489531153)

[7.3.5. 数字加法函数-DecimalAdd 34](#_Toc489531154)

[7.3.6. 数字减法函数-DecimalSubtract 35](#_Toc489531155)

[7.3.7. 数字乘法函数-DecimalMultiply 35](#_Toc489531156)

[7.3.8. 数字除法函数-DecimalDivide 35](#_Toc489531157)

[7.3.9. 数字求幂函数-DecimalPow 35](#_Toc489531158)

[7.3.10. 数字格式化函数-Format 36](#_Toc489531159)

[7.3.11. 数字比较函数-LessEquals 36](#_Toc489531160)

[7.3.12. 数字比较函数-LessThen 36](#_Toc489531161)

[7.3.13. 数字比较函数-MoreEquals 36](#_Toc489531162)

[7.3.14. 数字比较函数-MoreThen 37](#_Toc489531163)

[7.3.15. 数字设置精度函数-SetScale 37](#_Toc489531164)

[7.3.16. 对象转数字函数-ToDecimal 37](#_Toc489531165)

[7.3.17. 对象转数字函数-ToDouble 38](#_Toc489531166)

[7.3.18. 对象转数字函数-ToFloat 38](#_Toc489531167)

[7.3.19. 对象转数字函数-ToInteter 38](#_Toc489531168)

[7.3.20. 对象转数字函数-ToLong 38](#_Toc489531169)

[7.4. 集合类 38](#_Toc489531170)

[7.4.1. 集合空判断函数-IsEmpty 39](#_Toc489531171)

[7.4.2. 集合空判断函数-IsNotEmpty 39](#_Toc489531172)

[7.4.3. 集合去空函数-RemoveNull 39](#_Toc489531173)

[7.5. 表达式类 40](#_Toc489531174)

[7.5.1. 正则字符串查找函数-Find 40](#_Toc489531175)

[7.5.2. 正则字符串匹配函数-Matches 40](#_Toc489531176)

[7.5.3. 布尔值取反函数-Negate 40](#_Toc489531177)

[7.6. 字典类 40](#_Toc489531178)

[7.6.1. 业务字典转换函数-BusiCodeConvert 41](#_Toc489531179)

[7.6.2. 业务字典转换规则 41](#_Toc489531180)

[8. Dashboard 42](#_Toc489531181)

[9. 模型设计 43](#_Toc489531182)

[9.1. 模型树 43](#_Toc489531183)

[9.2. 模型 46](#_Toc489531184)

[9.3. 包 47](#_Toc489531185)

[9.4. 类 48](#_Toc489531186)

[9.5. 域 51](#_Toc489531187)

[10. 规则编辑 53](#_Toc489531188)

[10.1. 参数 53](#_Toc489531189)

[10.2. 规则 55](#_Toc489531190)

[10.3. 模板 56](#_Toc489531191)

[11. 规则管控 57](#_Toc489531192)

[11.1. 版本管理 57](#_Toc489531193)

[11.2. 发布管理 58](#_Toc489531194)

[11.3. 规则测试 59](#_Toc489531195)

[11.4. 规则监控 65](#_Toc489531196)

[12. 系统配置 68](#_Toc489531197)

[13. 附录 69](#_Toc489531198)

[13.1. 时间格式字符串 69](#_Toc489531199)

[13.2. 正则表达式 72](#_Toc489531200)

[13.2.1. 正则表达式的构造摘要 72](#_Toc489531201)

[13.2.2. 反斜线、转义和引用 75](#_Toc489531202)

[13.2.3. 字符类 76](#_Toc489531203)

[13.2.4. 行结束符 76](#_Toc489531204)

[13.2.5. 组和捕获 77](#_Toc489531205)

[13.2.6. Unicode 支持 77](#_Toc489531206)

[13.2.7. 与 Perl 5 相比较 78](#_Toc489531207)

[13.3. 数字格式字符串 79](#_Toc489531208)

[13.3.1. 模式 79](#_Toc489531209)

[13.3.2. 特殊模式字符 80](#_Toc489531210)

[13.3.3. 科学计数法 81](#_Toc489531211)

[13.3.4. 舍入 82](#_Toc489531212)

[13.3.5. 阿拉伯数字 82](#_Toc489531213)

[13.3.6. 特殊值 82](#_Toc489531214)

# 引言

## 编写目的

## 读者对象

使用数据引擎进行二次应用开发的开发人员、设计人员和需求分析人员。

## 术语定义

|  |  |
| --- | --- |
| **术语/缩写** | **解释** |
| 决策中心 | 整合了数据引擎和规则引擎优势的系统，完成规则调用过程中数据的自主获取。 |
| 数据引擎 | 重新设计开发的的数据引擎模块，负责数据定义、数据获取、数据销毁等数据生命周期的管理。 |
| 规则引擎 | 使用开源规则内核、二次封装的规则引擎模块，主要负责规则生命周期的管理 |
| 元数据 | 描述数据的数据。这里指整个元数据模型，其中包括数据定义、数据收集、数据存储、和计算函数等配置 |
| 数据定义 | 指数据内容的定义，指明了数据所包含的属性以及数据类型等关键信息。 |
| 数据收集 | 定义了数据获取的来源和方式。 |
| 数据存储 | 定义了数据存储的位置和形式。支持定义多级存储。 |
| 计算函数 | 允许配置数据在收集后，进行二次内存计算的函数。 |

## 参考资料

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **文件编号** | **文献名称** | **作 者** | **出版单位** | **发行日期** |
|  | 决策中心20170427.pptx |  |  |  |
|  | 设计文档-数据引擎.docx |  |  |  |

# 系统概述

## 项目背景

## 系统目标

## 条件与限制

## 推荐运行环境

应用服务器硬件配置：4核CPU、16G内存以上配置

应用服务器软件：Linux操作系统、Redis缓存系统。

数据库服务器：8核CPU、32G内存以上配置

数据库服务器软件：mysql5.5+、orcale10+、db2 8.1+

用户端：双核CPU、4G内存、IE9+或Chrome浏览器。

## 推荐开发环境

硬件：双核CPU、4G内存

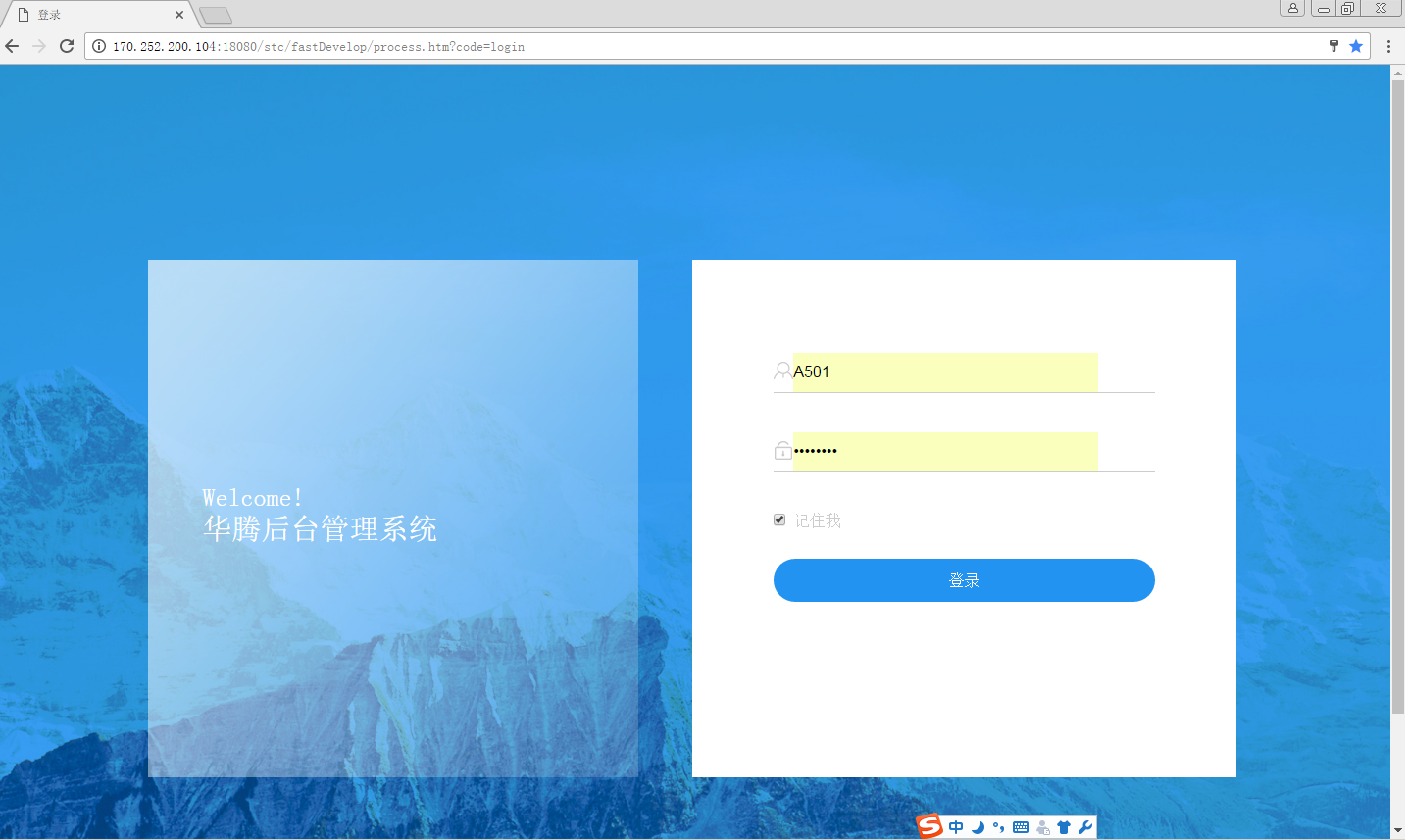
操作系统：windows

软件：Tomcat、eclipse、DbVisualizer

# 系统登录

## 系统登录

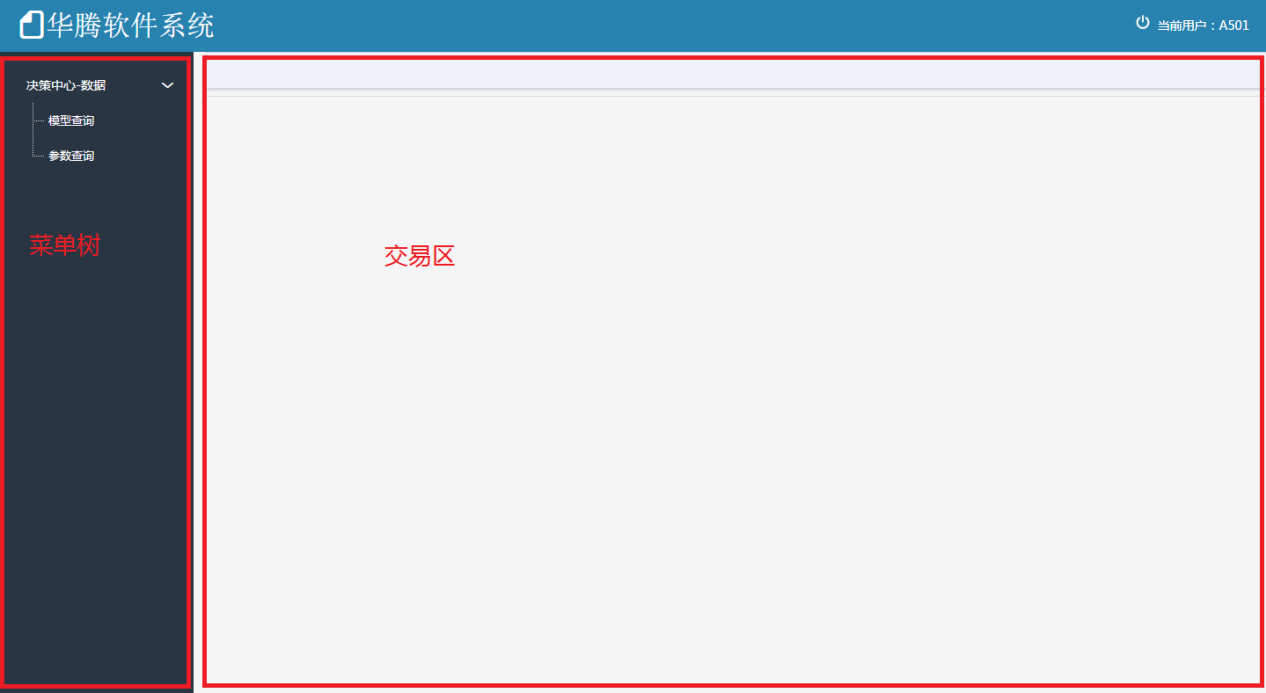
登录URL：<http://170.252.200.104:18080/stc/fastDevelop/process.htm?code=login>



用户名：A501

密码：88888888

## 系统首页



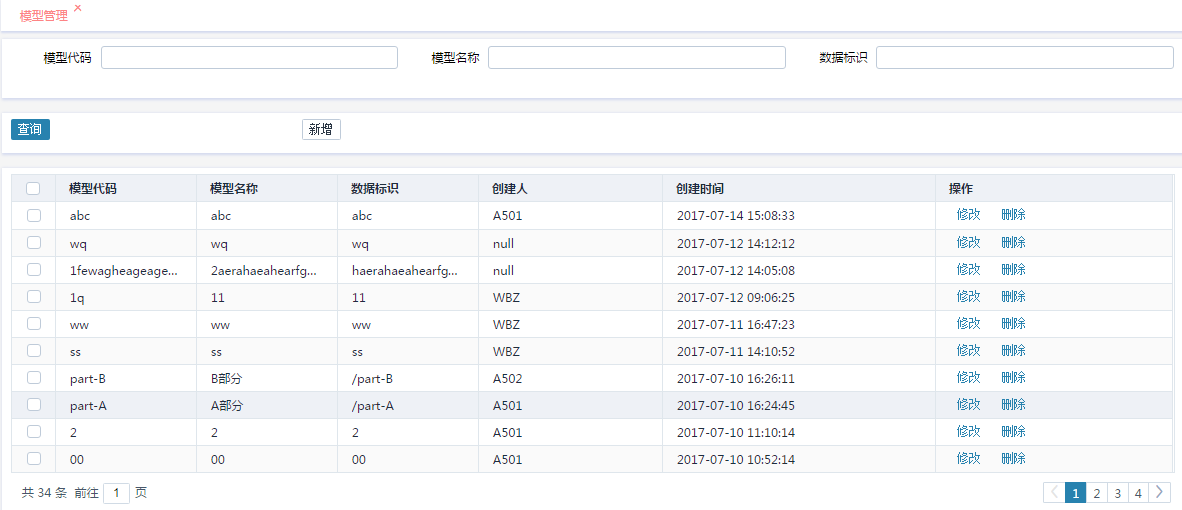
界面风格为左右分屏：

左侧：功能菜单树。登录角色所具备的全部功能菜单将会挂在树形控件上。

右侧：交易界面区。点击左侧交易菜单树上的功能时，右侧将会出现交易界面

# 模型管理

## 模型查询



**界面元素说明**：

|  |  |  |
| --- | --- | --- |
| **界面元素名称** | **界面元素类型** | **说明** |
| 模型代码 | 查询条件 |  |
| 模型名称 | 查询条件 | 支持模糊匹配 |
| 数据标示 | 查询条件 |  |
| 查询 | 按钮 |  |
| 新增 | 按钮 | 弹出新增模型界面 |
| 模型代码 | 列 |  |
| 模型名称 | 列 |  |
| 数据标示 | 列 |  |
| 模型类型 | 列 |  |
| 创建人 | 列 |  |
| 创建时间 | 列 |  |
| 修改 | 链接 | 进入模型维护界面 |
| 删除 | 链接 | 删除模型及相关全部数据 |

## 模型新增



**界面元素说明**：

|  |  |  |
| --- | --- | --- |
| **界面元素名称** | **界面元素类型** | **说明** |
| 模型代码 | 输入项 |  |
| 模型名称 | 输入项 |  |
| 数据标示 | 输入项 |  |
| 模型类型 | 下拉框 |  |
| 提交 | 按钮 | 保存新增的模型信息 |
| 返回 | 按钮 | 返回模型查询页面 |

## 模型维护-数据结构



**界面元素说明**：

|  |  |  |
| --- | --- | --- |
| **界面元素名称** | **界面元素类型** | **说明** |
| 返回 | 按钮 | 返回模型查询页面 |
| 模型代码 | 输入项 | 必输，建议是字母+数字的组合 |
| 模型名称 | 输入项 | 必输 |
| 数据标示 | 输入项 | 必输 |
| 模型类型 | 输入项 | 必输  简单类型：单一属性的对象  复杂类型：具备多个属性的对象 |
| 最少出现 | 输入项 | 允许为空：可能为null  必然出现：说明是必然存在的必输项 |
| 最多出现 | 输入项 | 最多出现一次：数据只能最多出现一次，不会是数组  数组：该数据对象允许出现多次 |
| 排序 | 输入项 | 影响在“模型查询”页面中顺序 |
| 数据处理 | 输入项 | 暂时无作用 |
| 描述 | 输入框 |  |
| 创建人 | 只读项 |  |
| 创建时间 | 只读项 |  |
| 更新人 | 只读项 |  |
| 更新时间 | 只读项 |  |
| 保存 | 按钮 | 保存模型信息 |
| 新增参数 | 按钮 | 进入新增模型扩展参数界面 |

## 模型维护-数据收集



**界面元素说明**：

|  |  |  |
| --- | --- | --- |
| **界面元素名称** | **界面元素类型** | **说明** |
| 收集实现 | 输入项 | 必输  填收集的实现类或工具的名称  （[详细请参考5-数据收集工具](#_数据收集工具)） |
| 提交 | 按钮 | 保存数据收集信息 |
| 删除 | 按钮 | 删除数据收集信息以及相关信息 |
| 新增 | 按钮 | [进入新增数据收集扩展参数界面](#_扩展参数) |

## 模型维护-数据存储

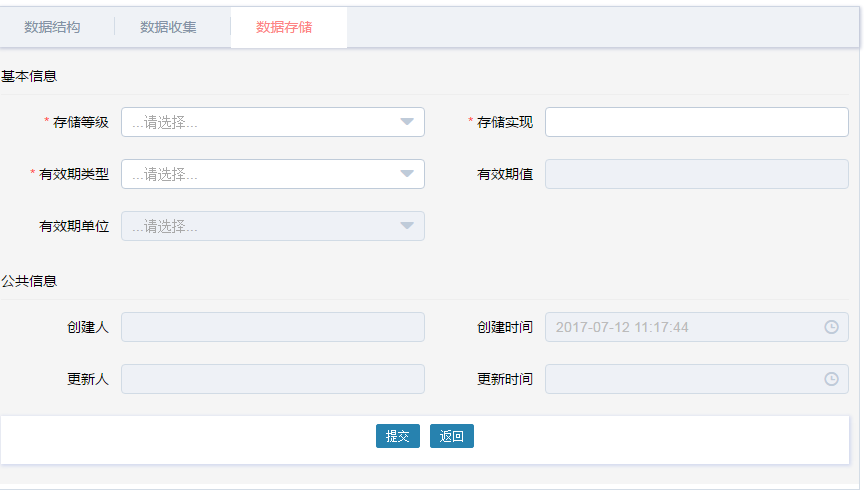
### 查询页面



**界面元素说明**：

|  |  |  |
| --- | --- | --- |
| **界面元素名称** | **界面元素类型** | **说明** |
| 新增 | 按钮 | [进入新增数据存储界面](#_新增/维护页面_3) |

### 新增/维护页面

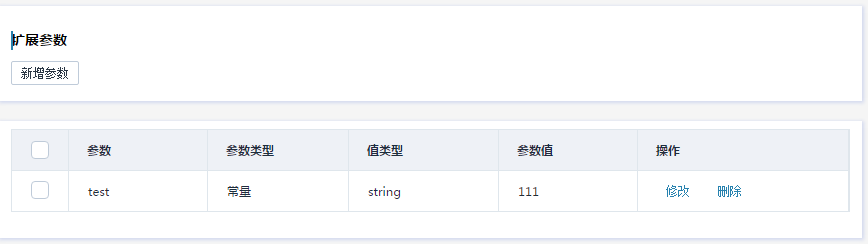


**界面元素说明**：

|  |  |  |  |
| --- | --- | --- | --- |
| **界面元素名称** | **界面元素类型** | **是否必输** | **说明** |
| 存储等级 | 下拉框 | 是 | 01.一级；02.二级；03.三级；04.四级；05.五级； |
| 存储实现 | 输入框 | 是 |  |
| 有效期类型 | 下拉框 | 是 | 01：临时；02：永久 |
| 有效期值 | 输入框 |  | 数据有效期为“临时”，则必输 |
| 有效期单位 | 下拉框 |  | 数据有效期为“临时”，则必输 |
| 创建人 | 输入框 |  |  |
| 创建时间 | 输入框 |  |  |
| 更新人 | 输入框 |  |  |
| 更新时间 | 输入框 |  |  |
| 提交 | 按钮 |  | 点击后触发页面唯一性、必输、长度校验，通过校验后，传递数据到后台服务器，保存入数据库“【数据存储配置表】” |
| 返回 | 按钮 |  | [返回数据存储查询页面](#_查询页面) |

## 扩展参数

### 查询页面



**界面元素说明**：

|  |  |  |
| --- | --- | --- |
| **界面元素名称** | **界面元素类型** | **说明** |
| 新增参数 | 按钮 | [点击后，进入新增扩展参数的界面](#_新增/维护页面) |
| 参数名称 | 只读表格列 |  |
| 参数来源类型 | 只读表格列 | 常量：静态常量数据  数据标示引用：引用其他模型数据  上下文引用：请求MAP对象中的数据项 |
| 参数数据类型 | 只读表格列 | 指明参数的数据类型：  String：字符串  Int：整数型  Date：日期型  Timestamp：时间戳  Boolean：布尔型  Long: 长整形  Decimal: 大数字型  Float: 浮点型  Double: 双精度浮点型  Clob: 文本大字段（不建议使用）  Blob: 二进制大字段（不建议使用） |
| 参数值 | 只读表格列 | 参数数据 |
| 修改 | 按钮 | [点击后，进入修改扩展参数的界面](#_新增/维护页面) |
| 删除 | 按钮 | 点击，可以删除扩展参数 |

### 新增/维护页面

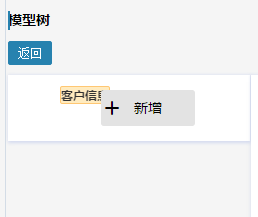


**界面元素说明**：

|  |  |  |
| --- | --- | --- |
| **界面元素名称** | **界面元素类型** | **说明** |
| 参数名称 | 输入项 | 必输 |
| 参数来源类型 | 输入项 | 必输  常量：静态常量数据  数据标示引用：引用其他模型数据  上下文引用：请求MAP对象中的数据项 |
| 参数值 |  |  |
| 参数数据类型 | 输入项 | 指明参数的数据类型：  String：字符串  Int：整数型  Date：日期型  Timestamp：时间戳  Boolean：布尔型  Long: 长整形  Decimal: 大数字型  Float: 浮点型  Double: 双精度浮点型  Clob: 文本大字段（不建议使用）  Blob: 二进制大字段（不建议使用） |
| 描述 | 输入框 |  |
| 创建人 | 只读项 |  |
| 创建时间 | 只读项 |  |
| 更新人 | 只读项 |  |
| 更新时间 | 只读项 |  |
| 保存 | 按钮 | 保存参数信息 |

## 数据项新增

首先在模型树，鼠标右键点击需要新增数据项的节点。



在右键菜单中，点击“新增”按钮。



**界面元素说明**：

|  |  |  |
| --- | --- | --- |
| **界面元素名称** | **界面元素类型** | **说明** |
| 数据项代码 | 输入项 | 必输，建议是字母+数字的组合 |
| 数据项名称 | 输入项 | 必输 |
| 数据项类型 | 输入项 | 必输  简单类型：最终的数据，不具备子数据项  复杂类型：还具备一个或多个子数据项 |
| 最少出现 | 输入项 | 允许为空：可能为null  必然出现：说明是必然存在的必输项 |
| 最多出现 | 输入项 | 最多出现一次：数据只能最多出现一次，不会是数组  数组：该数据对象允许出现多次 |
| 数据类型 | 输入项 | 当“数据项类型”时，必输  String：字符串  Int：整数型  Date：日期型  Timestamp：时间戳  Boolean：布尔型  Long: 长整形  Decimal: 大数字型  Float: 浮点型  Double: 双精度浮点型  Clob: 文本大字段（不建议使用）  Blob: 二进制大字段（不建议使用） |
| 数据处理 | 输入项 | 暂时无作用 |
| 长度 | 输入项 |  |
| 精度 | 输入项 | 当“数据类型”为Decimal: 大数字型、Float: 浮点型、Double: 双精度浮点型时，才有意义，标示小数点后面多少位。其他数据类型，该项数据默认为0 |
| 排序 | 输入项 | 影响数据项在“模型树”中相同层次的顺序 |
| 业务字典 | 输入项 |  |
| 描述 | 输入框 |  |
| 创建人 | 只读项 |  |
| 创建时间 | 只读项 |  |
| 更新人 | 只读项 |  |
| 更新时间 | 只读项 |  |
| 保存 | 按钮 | 保存模型信息 |
| 新增参数 | 按钮 | 进入新增数据项扩展参数界面 |

## 数据项维护

### 数据结构



### 计算函数

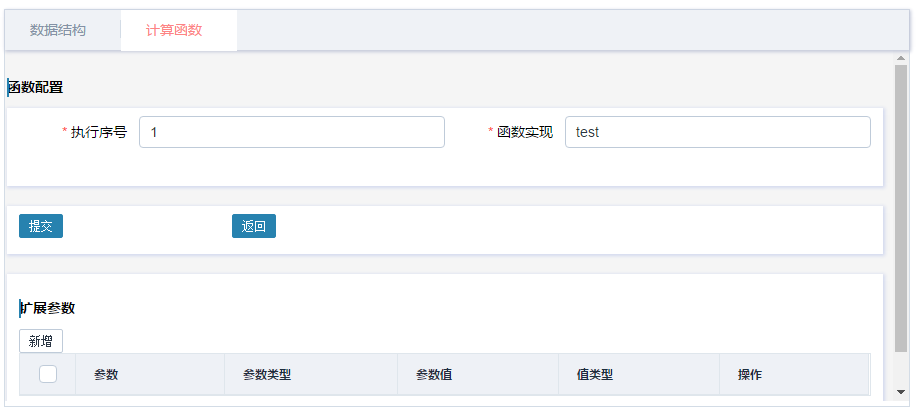
##### 查询页面



**界面元素说明**：

|  |  |  |
| --- | --- | --- |
| **界面元素名称** | **界面元素类型** | **说明** |
| 新增 | 按钮 | [点击后，进入新增计算函数的界面](#_新增/维护页面_1) |
| 执行序号 | 只读表格列 |  |
| 函数实现 | 只读表格列 |  |
| 更新人 | 只读表格列 |  |
| 修改 | 按钮 | [点击后，进入修改计算函数的界面](#_新增/维护页面_1) |
| 删除 | 按钮 | 点击，可以删除计算函数 |

##### 新增/维护页面

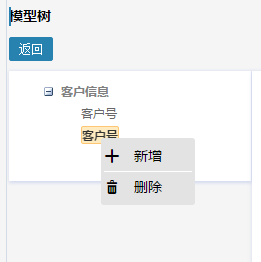


**界面元素说明**：

|  |  |  |
| --- | --- | --- |
| **界面元素名称** | **界面元素类型** | **说明** |
| 新增 | 按钮 | [点击后，进入新增计算函数的界面](#_新增/维护页面_1) |
| 执行序号 | 输入项 | 必输 |
| 函数实现 | 输入项 | 必输  [具体请参考7-计算函数](#_计算函数) |
| 提交 | 按钮 | 保存计算函数配置 |
| 返回 | 按钮 | 返回计算函数页面 |
| 新增 | 按钮 | [点击，可以进入新增扩展参数界面](#_新增/维护页面) |

## 数据项删除

在模型树上，右键单击需要删除的数据项：



然后在右键菜单中，点击“删除”按钮。

# 数据收集工具

## JDBC工具

一种从关系型数据库，使用JDBC方式收集数据的工具

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.service.collect.impl.CollectServiceByJDBCImpl | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| jndiName | 例如：  java:comp/env/jdbc/stc | 容器中配置的数据源JNDI |
| selectSql | select CASE\_ID AS caseId, TEL\_NO AS telNo, CERT\_NO AS custId from TBL\_STC\_CUST\_INFO WHERE CUST\_NAME = ? | 数据库查询SQL |
| selectParam0 |  | 查询SQL中第一个“？”参数值 |
| selectParam1 |  | 查询SQL中第二个“？”参数值 |
| 。。。。 |  | 。。。。 |
| selectParamX |  | 查询SQL中第X-1个“？”参数值 |
|  |  |  |

## Redis缓存工具

一种从Redis缓存中获取数据的工具，key为：数据标示+业务标示。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.service.collect.impl.CollectServiceByRedisImpl | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

# 数据存储工具

## JDBC工具

一种从关系型数据库，使用JDBC方式存储数据的工具

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.service.store.impl.DBStoreService | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

## Redis缓存工具

一种从Redis缓存中存储数据的工具，key为：数据标示+业务标示。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.service.store.impl.RedisStoreService | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

# 计算函数

## 字符串类

### 时间字符串转格式函数-DateStringFormat

从一种个是的时间字符串，转换格式成另一种时间字符串格式

例如：“2010-01-02”转为“1月2日2010年”

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.DateStringFormatFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| format | 例如：yyyy-MM-dd | 配置时间格式字符串 |
| targetFormat | 例如：yyyy-MM-dd | 配置时间格式字符串 |
|  |  |  |

[**注意：时间格式字符串参考附录**](#_时间格式字符串)

### 字符串相等比较函数-Equals

两个字符串相等比较，如果相等返回true，否则返回false

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.EqualsFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| value | 例如：中国、上海 | 待比较的字符串 |
|  |  |  |

### 字符串不相等比较函数-NotEquals

两个字符串不相等比较，如果不相等返回true，否则返回false

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.NotEqualsFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| value | 例如：中国、上海 | 待比较的字符串 |
|  |  |  |

### 字符串空判断函数-IsBlank

判断字符串是否为空，如果是返回true，否则返回false。

空字符串包括如下情况：

1. null
2. “”
3. “ ” （引号内存在一个或者多个英文空格，中文全角空格除外）

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.IsBlankFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

### 字符串非空判断函数-IsNotBlank

判断字符串是否为非空，如果是返回true，否则返回false。

空字符串包括如下情况：

1. null
2. “”
3. “ ” （引号内存在一个或者多个英文空格，中文全角空格除外）

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.IsNotBlankFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

### 字符串替换函数-Replace

把字符串中指定“查找字符串”替换成“替换字符串”，最终返回数组

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.ReplaceFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| searchString | 例如：aaa | 查找字符串。 |
| replacement | 例如：bbb | 替换字符串。 |
|  |  |  |

### 字符串分割函数-Split

将字符串按指定的分隔符进行分隔，最终返回数组

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.SplitFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| separator | 例如：; | 分割符，如果不设置的话，默认为,（英文逗号）。 |
|  |  |  |

### 字符串截取函数-SubStringAfter

截取字符串，指定字符（第一个匹配）之后的内容

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.SubStringAfterFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| separator | 例如：; | 分割符，如果不设置的话，默认为,（英文逗号）。 |
|  |  |  |

### 字符串截取函数-SubStringAfterLast

截取字符串，指定字符（最后一个匹配）之后的内容

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.SubStringAfterLastFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| separator | 例如：; | 分割符，如果不设置的话，默认为,（英文逗号）。 |
|  |  |  |

### 字符串截取函数-SubStringBefore

截取字符串，指定字符（第一个匹配）之前的内容

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.SubStringBeforeFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| separator | 例如：; | 分割符，如果不设置的话，默认为,（英文逗号）。 |
|  |  |  |

### 字符串截取函数-SubStringBeforeLast

截取字符串，指定字符（最后一个匹配）之前的内容

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.SubStringBeforeLastFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| separator | 例如：; | 分割符，如果不设置的话，默认为,（英文逗号）。 |
|  |  |  |

### 字符串截取函数-SubStringBetween

截取字符串，指定字符之间的内容

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.SubStringBetweenFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| separator | 例如：{ | 开始字符串。 |
| separatorEnd | 例如：} | 结束字符串 |

### 字符串截取函数-SubStringsBetween

截取字符串，指定字符之间的内容，返回字符串数组对象

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.SubStringsBetweenFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| separator | 例如：{ | 开始字符串。 |
| separatorEnd | 例如：} | 结束字符串 |

### 字符串截取函数-SubString

截取字符串，指定下标之间的内容，返回截取后的字符串

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.SubStringFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| start | 例如：0 | 开始下标，从0开始。  如果不设置，默认为0。 |
| end | 例如：20 | 结束下标  如果不设置，默认到字符串结尾。 |

### 字符串转布尔类型函数-ToBoolean

将字符串转换为boolean类型数据

True的场景（忽略大小写）：

1. true
2. 1
3. Y
4. YES
5. ON

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.ToBooleanFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

### 字符串转时间类型函数-ToDate

将字符串转换为Date类型数据

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.ToDateFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| format | 例如：yyyy-MM-dd | 配置时间格式字符串 |
|  |  |  |

[**注意：时间格式字符串参考附录**](#_时间格式字符串)

### 对象转字符串类型函数-ToString

将对象转换为String类型数据

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.ToStringFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

### 字符串去空格函数-Trim

可以去除字符串开头和结尾的英文空格。

如何字符串是null的话，返回null。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.TrimFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

### 字符串去空格函数-TrimToEmpty

可以去除字符串开头和结尾的英文空格。

如何字符串是null的话，返回空字符串。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.string.TrimToEmptyFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

## 日期类

### 时间比较函数-After

判断时间是否在指定时间之后，如果是返回true，否则返回false。

支持的时间格式：

1. date类型
2. long类型
3. yyyyMMdd
4. yyyy-MM-dd
5. yyyy-MM-dd HH:mm:ss
6. yyyy-MM-dd HH:mm
7. yyyy-MM-dd HH:mm:ss.SSS

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.date.AfterFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| value | 例如：2010-10-10 | 待比较的时间字符串 |
|  |  |  |

### 时间比较函数-NotAfter

判断时间是否不在指定时间之后，如果是返回true，否则返回false。

支持的时间格式：

1. date类型
2. long类型
3. yyyyMMdd
4. yyyy-MM-dd
5. yyyy-MM-dd HH:mm:ss
6. yyyy-MM-dd HH:mm
7. yyyy-MM-dd HH:mm:ss.SSS

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.date.NotAfterFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| value | 例如：2010-10-10 | 待比较的时间字符串 |
|  |  |  |

### 时间比较函数-Before

判断时间是否在指定时间之前，如果是返回true，否则返回false。

支持的时间格式：

1. date类型
2. long类型
3. yyyyMMdd
4. yyyy-MM-dd
5. yyyy-MM-dd HH:mm:ss
6. yyyy-MM-dd HH:mm
7. yyyy-MM-dd HH:mm:ss.SSS

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.date.BeforeFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| value | 例如：2010-10-10 | 待比较的时间字符串 |
|  |  |  |

### 时间比较函数-NotBefore

判断时间是否不在指定时间之前，如果是返回true，否则返回false。

支持的时间格式：

1. date类型
2. long类型
3. yyyyMMdd
4. yyyy-MM-dd
5. yyyy-MM-dd HH:mm:ss
6. yyyy-MM-dd HH:mm
7. yyyy-MM-dd HH:mm:ss.SSS

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.date.NotBeforeFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| value | 例如：2010-10-10 | 待比较的时间字符串 |
|  |  |  |

### 时间比较函数-BetweenEquals

判断时间是否在指定时间之间，并且是min<=时间数据<=max，如果是返回true，否则返回false。

支持的时间格式：

1. date类型
2. long类型
3. yyyyMMdd
4. yyyy-MM-dd
5. yyyy-MM-dd HH:mm:ss
6. yyyy-MM-dd HH:mm
7. yyyy-MM-dd HH:mm:ss.SSS

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.date.BetweenEqualsFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| min | 例如：2010-10-10 | 待比较的时间字符串 |
| max | 例如：2011-10-10 | 待比较的时间字符串 |

### 时间比较函数-NotBetweenEquals

判断时间是否不在指定时间之间，并且是“时间数据”<min&&“时间数据”>max，如果是返回true，否则返回false。

支持的时间格式：

1. date类型
2. long类型
3. yyyyMMdd
4. yyyy-MM-dd
5. yyyy-MM-dd HH:mm:ss
6. yyyy-MM-dd HH:mm
7. yyyy-MM-dd HH:mm:ss.SSS

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.date.NotBetweenEqualsFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| min | 例如：2010-10-10 | 待比较的时间字符串 |
| max | 例如：2011-10-10 | 待比较的时间字符串 |

### 时间比较函数-Between

判断时间是否不在指定时间之间，并且是min<时间数据<max，如果是返回true，否则返回false。

支持的时间格式：

1. date类型
2. long类型
3. yyyyMMdd
4. yyyy-MM-dd
5. yyyy-MM-dd HH:mm:ss
6. yyyy-MM-dd HH:mm
7. yyyy-MM-dd HH:mm:ss.SSS

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.date.BetweenFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| min | 例如：2010-10-10 | 待比较的时间字符串 |
| max | 例如：2011-10-10 | 待比较的时间字符串 |

### 时间比较函数-NotBetween

判断时间是否在指定时间之间，并且是“时间数据”<=min&&“时间数据”>=max，如果是返回true，否则返回false。

支持的时间格式：

1. date类型
2. long类型
3. yyyyMMdd
4. yyyy-MM-dd
5. yyyy-MM-dd HH:mm:ss
6. yyyy-MM-dd HH:mm
7. yyyy-MM-dd HH:mm:ss.SSS

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.date.NotBetweenFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| min | 例如：2010-10-10 | 待比较的时间字符串 |
| max | 例如：2011-10-10 | 待比较的时间字符串 |

### 时间比较函数-CurrentDate

获取当前服务器系统时间，返回Date对象

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.date.CurrentDateFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

### 时间格式化函数-Format

将时间对象格式成指定的时间格式。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.date.FormatFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| format | 例如：yyyy-MM-dd | 配置时间格式字符串，如果不配置，默认为yyyy-MM-dd |
|  |  |  |

[**注意：时间格式字符串参考附录**](#_时间格式字符串)

## 数字类

### 数字比较函数-BetweenEquals

判断数字是否在指定的范围内，并且是min<=“数据”<=max，如果是返回true，否则返回false。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.BetweenEqualsFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| min | 例如：1 | 待比较的最小值 |
| max | 例如：2 | 待比较的最大值 |
|  |  |  |

### 数字比较函数-NotBetweenEquals

判断数字是否不在指定的范围内，并且是“数据”<min&&“数据”>max，如果是返回true，否则返回false。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.NotBetweenEqualsFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| min | 例如：1 | 待比较的最小值 |
| max | 例如：2 | 待比较的最大值 |
|  |  |  |

### 数字比较函数-Between

判断数字是否在指定的范围内，并且是min<“数据”<max，如果是返回true，否则返回false。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.BetweenFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| min | 例如：1 | 待比较的最小值 |
| max | 例如：2 | 待比较的最大值 |
|  |  |  |

### 数字比较函数-NotBetween

判断数字是否不在指定的范围内，并且是“数据”<=min&&“数据”>=max，如果是返回true，否则返回false。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.NotBetweenFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| min | 例如：1 | 待比较的最小值 |
| max | 例如：2 | 待比较的最大值 |
|  |  |  |

### 数字加法函数-DecimalAdd

数字加法，返回Decimal类型数据。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.DecimalAddFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| value | 例如：1 | 待相加的数字 |
|  |  |  |

### 数字减法函数-DecimalSubtract

数字减法，返回Decimal类型数据。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.DecimalSubtractFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| value | 例如：1 | 减数 |
|  |  |  |

### 数字乘法函数-DecimalMultiply

数字乘法，返回Decimal类型数据。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.DecimalMultiplyFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| value | 例如：1 | 乘数 |
|  |  |  |

### 数字除法函数-DecimalDivide

数字除法，返回Decimal类型数据。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.DecimalDivideFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| value | 例如：1 | 除数 |
|  |  |  |

### 数字求幂函数-DecimalPow

数字求幂，返回Decimal类型数据。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.DecimalPowFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| value | 例如：1 | 幂 |
|  |  |  |

### 数字格式化函数-Format

将时间对象格式成指定的数字格式字符串。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.FormatFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| format | 例如：#,###,###,###,###.## | 配置时间格式字符串，如果不配置，默认为yyyy-MM-dd |
|  |  |  |

**[注意：数字格式字符串参考附录](#_数字格式字符串)**

### 数字比较函数-LessEquals

判断数字是否<=待比较数字，如果是返回true，否则返回false。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.LessEqualsFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| value | 例如：1 | 待比较的数字 |
|  |  |  |

### 数字比较函数-LessThen

判断数字是否<待比较数字，如果是返回true，否则返回false。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.LessThenFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| value | 例如：1 | 待比较的数字 |
|  |  |  |

### 数字比较函数-MoreEquals

判断数字是否>=待比较数字，如果是返回true，否则返回false。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.MoreEqualsFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| value | 例如：1 | 待比较的数字 |
|  |  |  |

### 数字比较函数-MoreThen

判断数字是否>待比较数字，如果是返回true，否则返回false。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.MoreThenFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| value | 例如：1 | 待比较的数字 |
|  |  |  |

### 数字设置精度函数-SetScale

将数字设置指定精度和舍入方式，返回Decimal数据。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.SetScaleFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| precision | 例如：2 | 精度，默认为0 |
| roundingMode | 例如：4 | 舍入模式，默认为四舍五入  0：ROUND\_UP  1：ROUND\_DOWN  2：ROUND\_CEILING  3：ROUND\_FLOOR  4：ROUND\_HALF\_UP（四舍五入）  5：ROUND\_HALF\_DOWN  6：ROUND\_HALF\_EVEN  7：ROUND\_UNNECESSARY |
|  |  |  |

### 对象转数字函数-ToDecimal

将对象转数字对象，返回Decimal数据。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.ToDecimalFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

### 对象转数字函数-ToDouble

将对象转数字对象，返回Double数据。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.ToDoubleFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

### 对象转数字函数-ToFloat

将对象转数字对象，返回Float数据。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.ToFloatFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

### 对象转数字函数-ToInteter

将对象转数字对象，返回Inteter数据。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.ToInteterFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

### 对象转数字函数-ToLong

将对象转数字对象，返回Long数据。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.number.ToLongFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

## 集合类

### 集合空判断函数-IsEmpty

判断集合是否为空，如果是返回true，否则返回false。

支持的集合类型：

1. 数组
2. Collection
3. Map

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.collection.IsEmptyFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

### 集合空判断函数-IsNotEmpty

判断集合是否为非空，如果是返回true，否则返回false。

支持的集合类型：

1. 数组
2. Collection
3. Map

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.collection.IsNotEmptyFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

### 集合去空函数-RemoveNull

去掉集合中的null对象，返回List<Object>对象。

支持的集合类型：

1. 数组
2. Collection

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.collection.RemoveNullFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

## 表达式类

### 正则字符串查找函数-Find

将字符串根据指定正则表达式查找，如果找到返回true，否则返回false。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.expression.FindFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| format | 例如：aaa | 正则表达式 |
|  |  |  |

**[注意：正则表达式请参考附录](#_正则表达式)**

### 正则字符串匹配函数-Matches

将字符串根据指定正则表达式匹配，如果找到返回true，否则返回false。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.expression.MatchesFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| format | 例如：aaa | 正则表达式 |
|  |  |  |

**[注意：正则表达式请参考附录](#_正则表达式)**

### 布尔值取反函数-Negate

将布尔值取反。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.expression.NegateFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
|  |  |  |

## 字典类

### 业务字典转换函数-BusiCodeConvert

将业务字典根据指定规则转换。

|  |  |  |
| --- | --- | --- |
| **函数实现类** | com.huateng.stc.function.dict.BusiCodeConvertFunction | |
| **扩展参数配置如下：** | | |
| **参数名称** | **参数值** | **补充说明** |
| convertRule | 例如：test | 转换规则码 |
|  |  |  |

### 业务字典转换规则

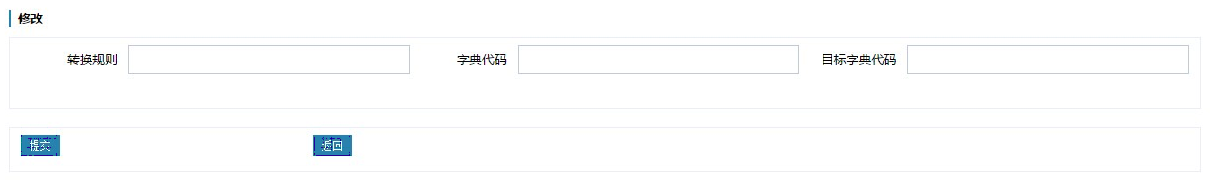
##### 查询页面



**界面元素说明**：

|  |  |  |
| --- | --- | --- |
| **界面元素名称** | **界面元素类型** | **说明** |
| 转换规则 | 查询条件 |  |
| 字典代码 | 查询条件 |  |
| 目标字典代码 | 查询条件 |  |
| 查询 | 按钮 |  |
| 新增 | 按钮 | [进入新增参数界面](#_新增/维护页面_2) |
| 转换规则 | 列 |  |
| 字典代码 | 列 |  |
| 目标字典代码 | 列 |  |
| 更新人 | 列 |  |
| 更新时间 | 列 |  |
| 修改 | 链接 | [进入修改参数界面](#_新增/维护页面_2) |
| 删除 | 链接 | 删除该笔参数 |

##### 新增/维护页面



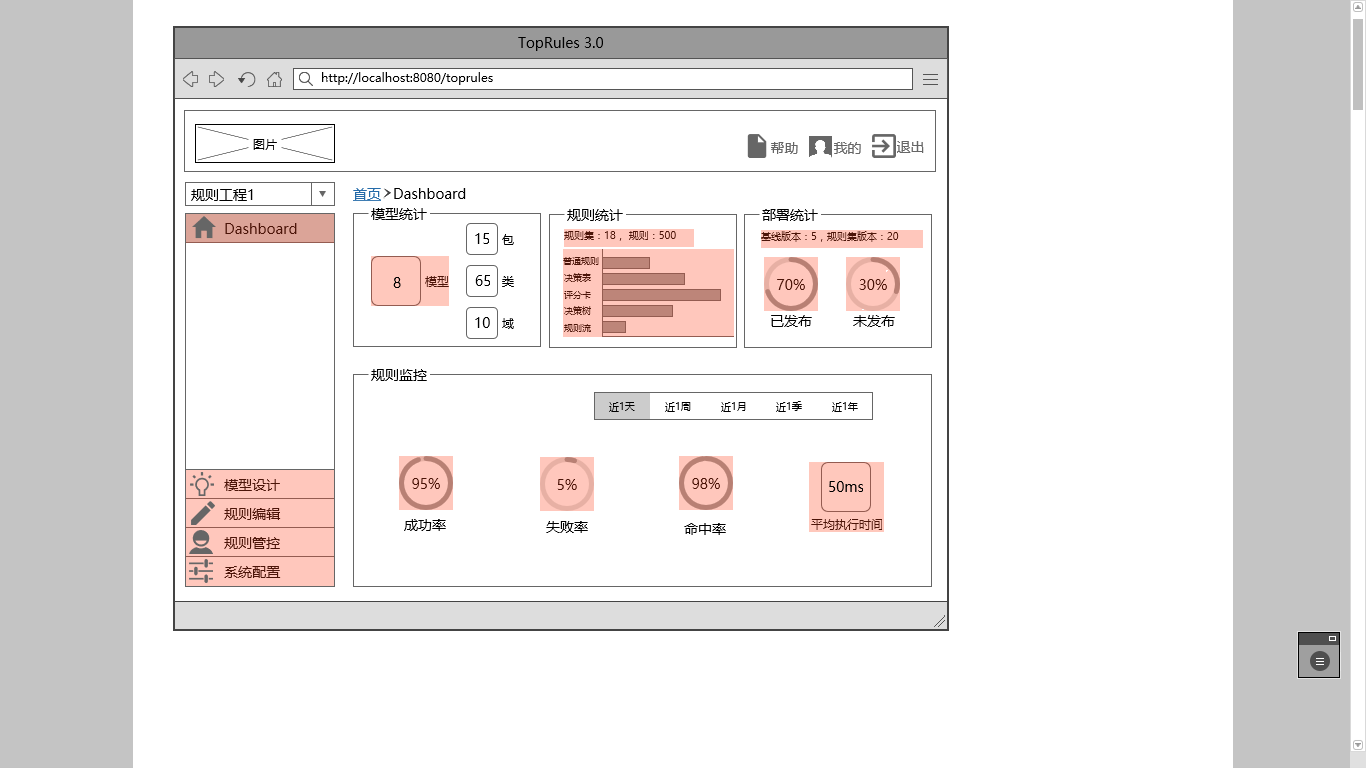
**界面元素说明**：

|  |  |  |
| --- | --- | --- |
| **界面元素名称** | **界面元素类型** | **说明** |
| 转换规则 | 输入项 |  |
| 字典代码 | 输入项 | 可以配置成：[DEFAULT]  作为默认转换规则 |
| 目标字典代码 | 输入项 |  |
| 提交 | 按钮 | 保存转换规则 |
| 返回 | 按钮 | 返回查询页面 |

# Dashboard

|  |  |
| --- | --- |
| 功能描述 | 从全局视角展示规则引擎管理平台的基本统计信息和运行监控信息 |
| 入口路径 | 登录成功后直接进入Dashboard页面 |
| 关键逻辑 | 左栏树形导航菜单上方为规则工程切换按钮（与现有flex版功能相同），点击该按钮弹出规则工程切换窗口，也可以进行规则工程增加、删除等维护功能；  模型统计中显示当前规则工程中的模型个数，所有模型中总的包个数、总的类个数和总的域个数；点击相应数字能跳转到对应的页面；  规则统计中显示当前规则工程中的总的规则集和规则个数，规则数再按普通规则、决策表、评分卡、决策树、规则流分别以柱状图显示；点击相应数字能跳转到对应的页面；  部署统计中显示当前规则工程中的总的基线版本数和规则集版本数，规则集版本数再按已发布和未发布分别统计百分比；点击相应数字能跳转到对应的页面；  规则监控中显示当前规则工程中所有已发布规则集最近一个时间段（按天、周、月、季、年切换）的成功率、失败率、命中率、平均执行时间等指标；点击相应数字能跳转到对应的页面。 |

（表3.1 Dashboard说明）



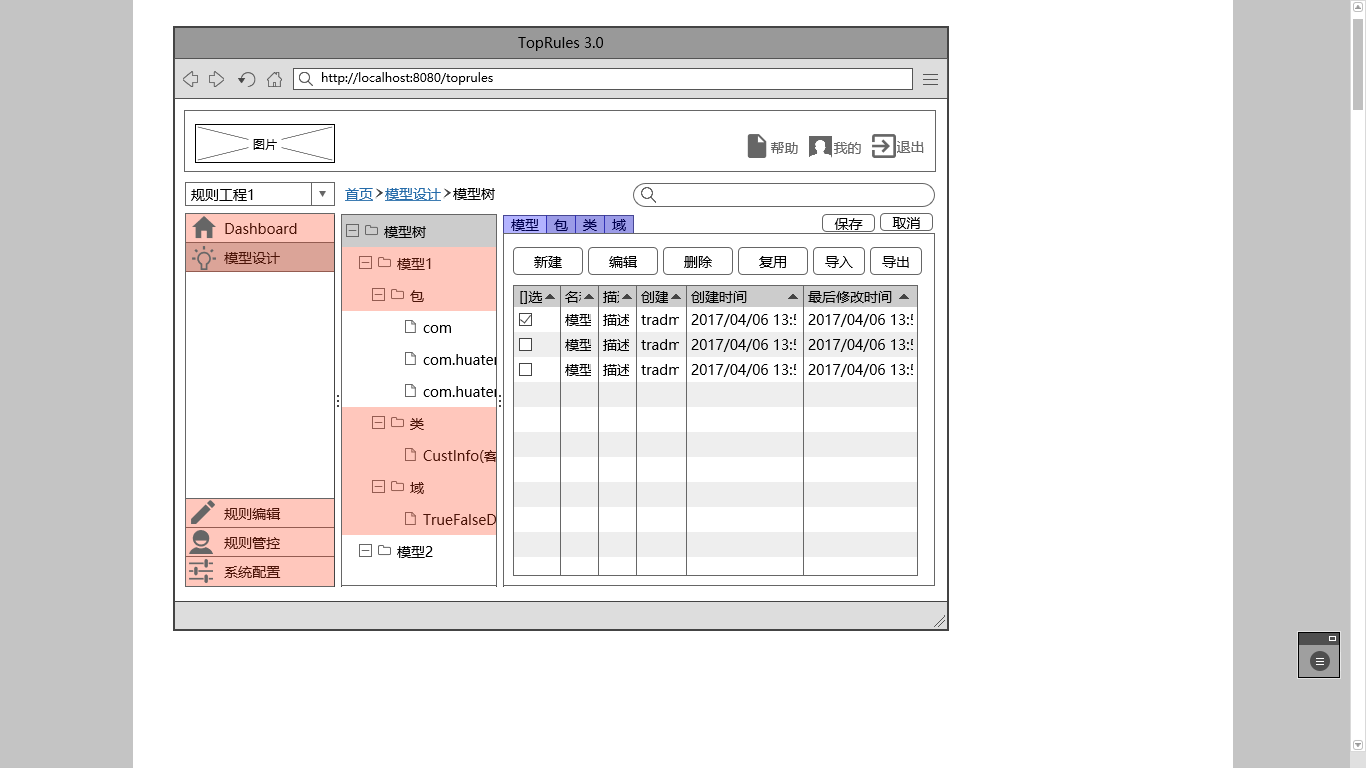
（图3.1 Dashboard页面）

# 模型设计

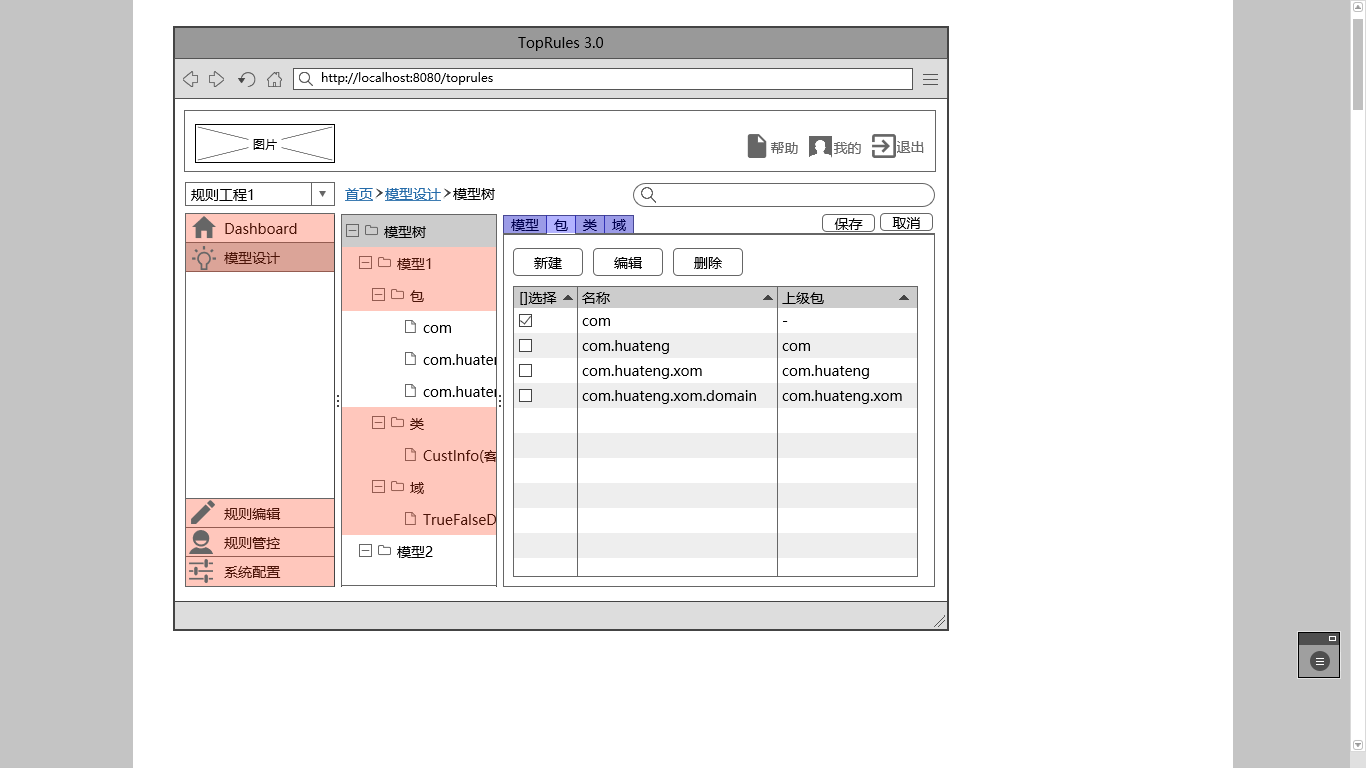
## 模型树

|  |  |
| --- | --- |
| 功能描述 | 编写业务规则之前需要建立数据模型，该功能为模型设计人员提供可视化的面向对象的数据建模页面（为了适应某些复杂的场景，也保留了现有上传JAR包、通过编码实现规则逻辑及构建动态适配器的建模方式） |
| 入口路径 | 从左侧导航菜单栏点击“模型设计”菜单默认进入模型树 |
| 关键逻辑 | 右侧导航条旁边的搜索框，支持按名称、描述等字段模糊检索模型、包、类或域；  中间模型树区域中列出当前规则工程中所有的模型及其包、类、域等子节点；  右侧主工作区域，分别列表显示了当前规则工程中的所有的模型、包、类、域的具体信息，并能通过表格上面的功能按钮进行基本的维护操作；  模型Tab页中“新建”按钮弹出新建模型对话框，用来输入模型名称和描述，不必上传JAR包。“导入”按钮用来上传JAR包，通过JAR包来导入指定模型内容，“导出”按钮用来将当前选择的模型导出为JAR包；  类Tab页中的“查看”和“编辑”按钮在新窗口中打开选中的单条类的详细信息进行查看或编辑，域Tab页类似；  新建模型后，需要同步更新模型树。 |

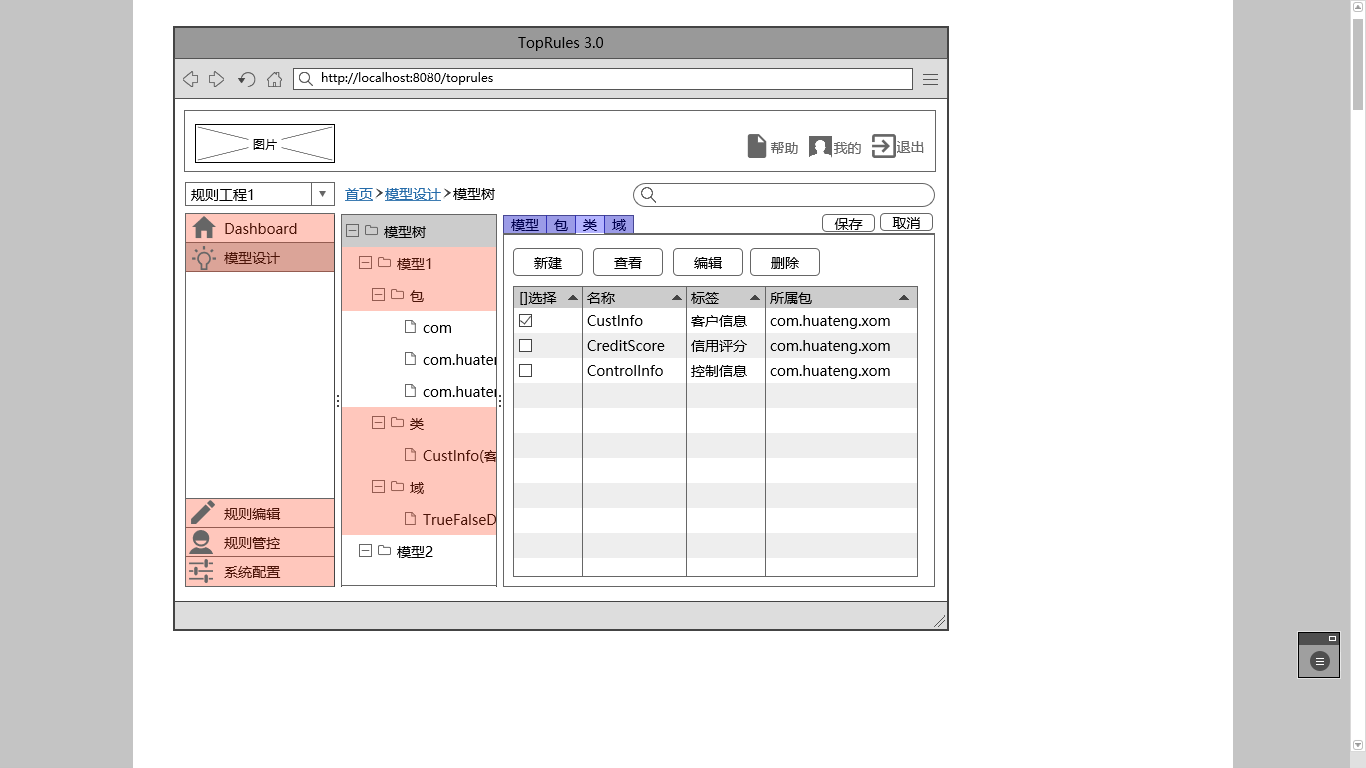
（表3.2.1 模型树说明）



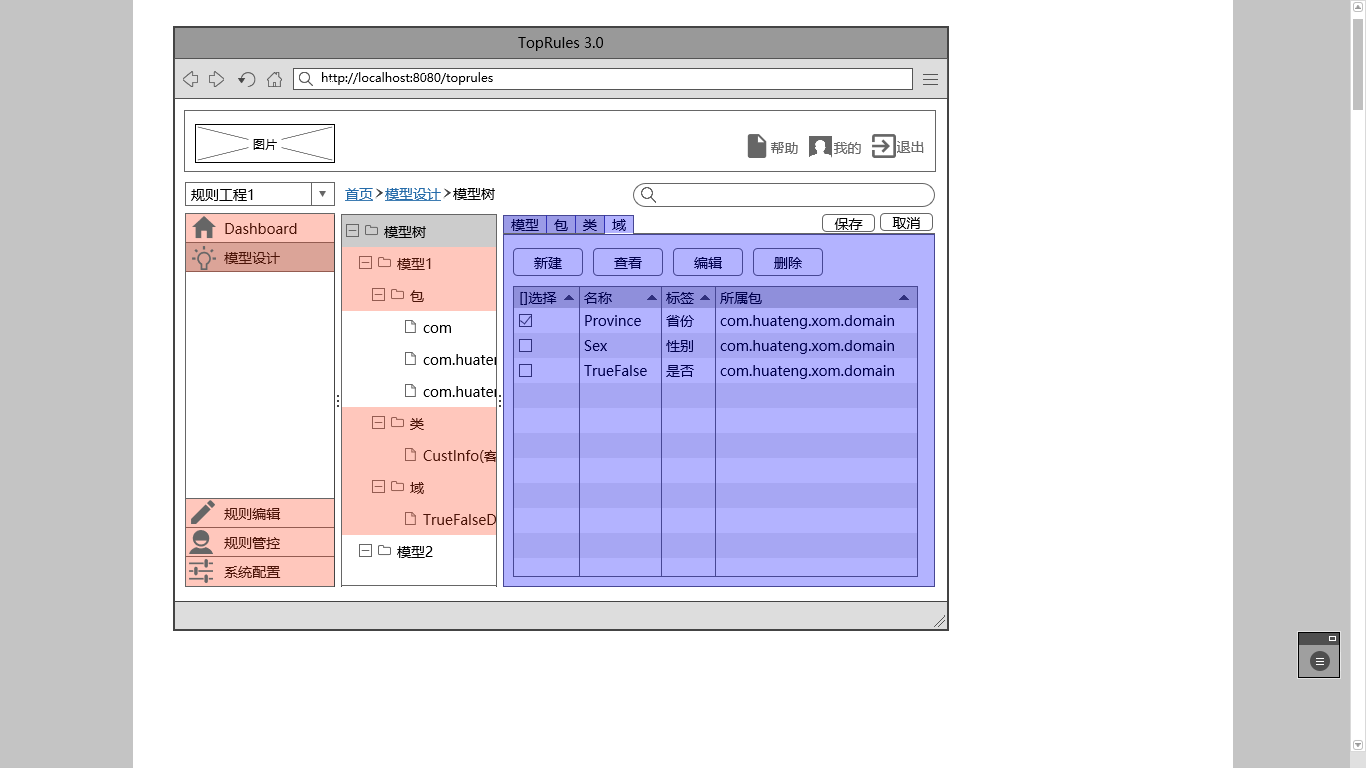
（图3.2.1.1 模型设计-模型树-模型页面）



（图3.2.1.2 模型设计-模型树-包页面）



（图3.2.1.3 模型设计-模型树-类页面）

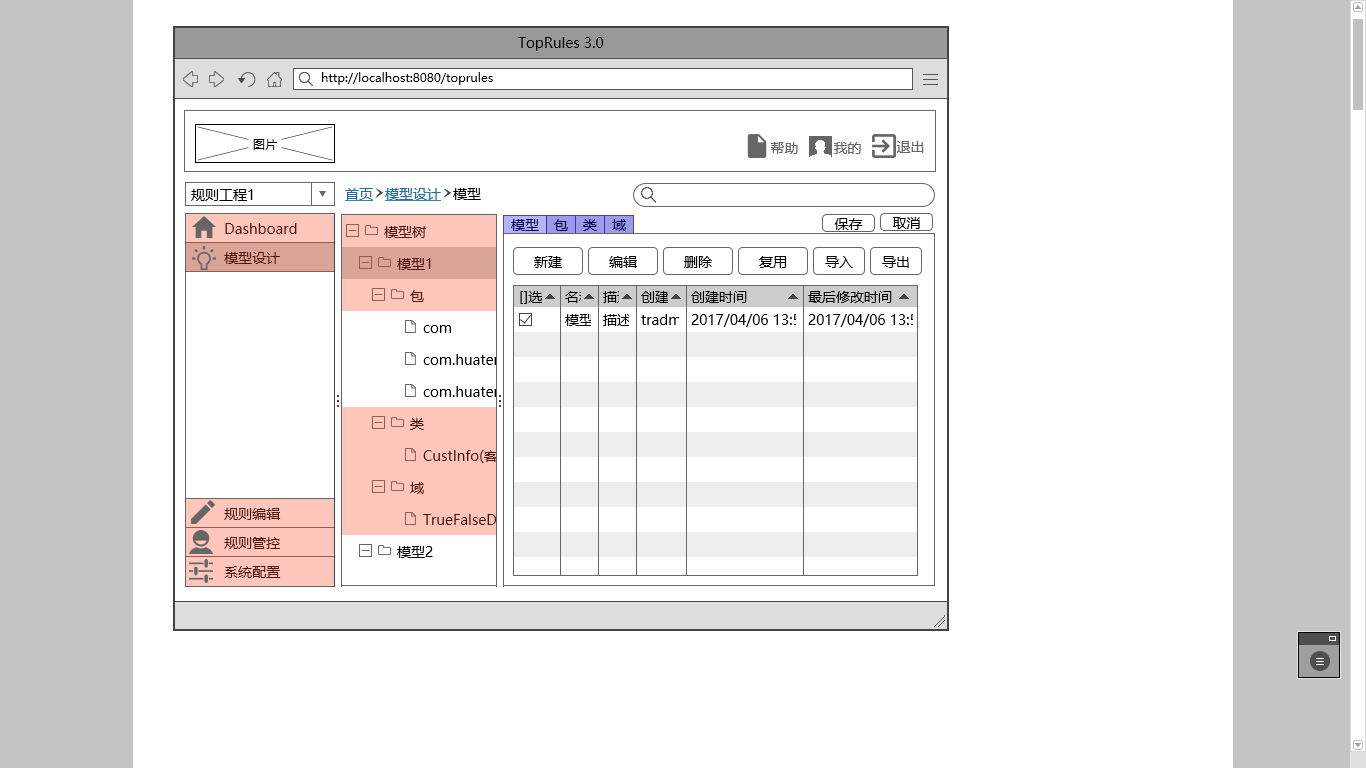


（图3.2.1.4 模型设计-模型树-域页面）

## 模型

|  |  |
| --- | --- |
| 功能描述 | 编写业务规则之前需要建立数据模型，该功能为模型设计人员提供可视化的面向对象的数据建模页面（为了适应某些复杂的场景，也保留了现有上传JAR包、通过编码实现规则逻辑及构建动态适配器的建模方式） |
| 入口路径 | 从左侧导航菜单栏点击“模型设计”打开模型树，选择指定模型，如“模型1” |
| 关键逻辑 | 1. 主工作区域的表格中只列出当前选中的模型的相关信息，包括该模型基本信息，模型下的包、类、域信息； 2. 新建模型后，需要同步更新模型树。 |

（表3.2.2 模型说明）

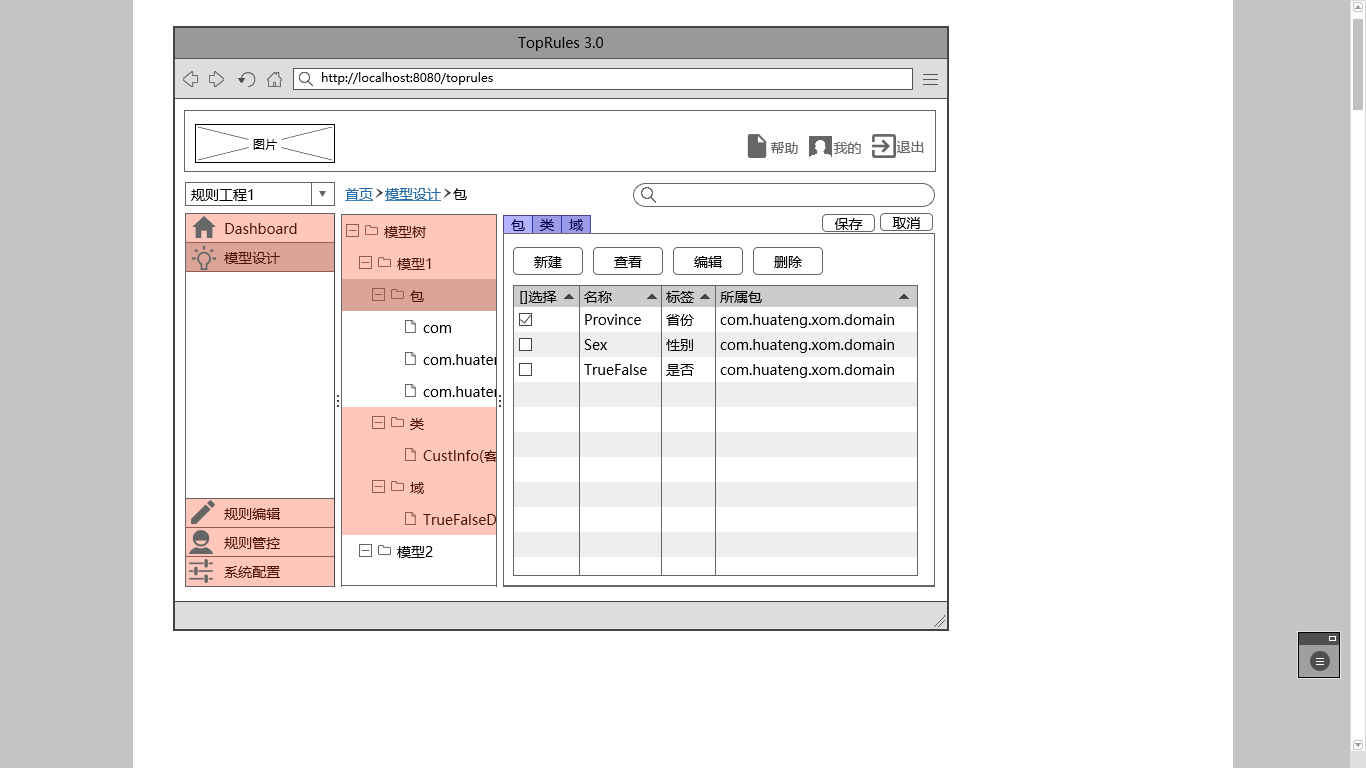


（图3.2.2.1 模型设计-模型页面）

## 包

|  |  |
| --- | --- |
| 功能描述 | 编写业务规则之前需要建立数据模型，该功能为模型设计人员提供可视化的面向对象的数据建模页面（为了适应某些复杂的场景，也保留了现有上传JAR包、通过编码实现规则逻辑及构建动态适配器的建模方式） |
| 入口路径 | 从左侧导航菜单栏点击“模型设计”打开模型树，选择指定模型，如“模型1”,再选中“包”节点 |
| 关键逻辑 | 1. 主工作区域的表格中列出当前选中的模型下的所有包、类、域信息； 2. 左侧模型树中选中某个包，右侧表格显示当前包的相关信息，包括该包的基本信息，包下的类、域信息； 3. 新建包后，需要同步更新模型树。 |

（表3.2.3 包说明）



（图3.2.3.1 模型设计-包页面）

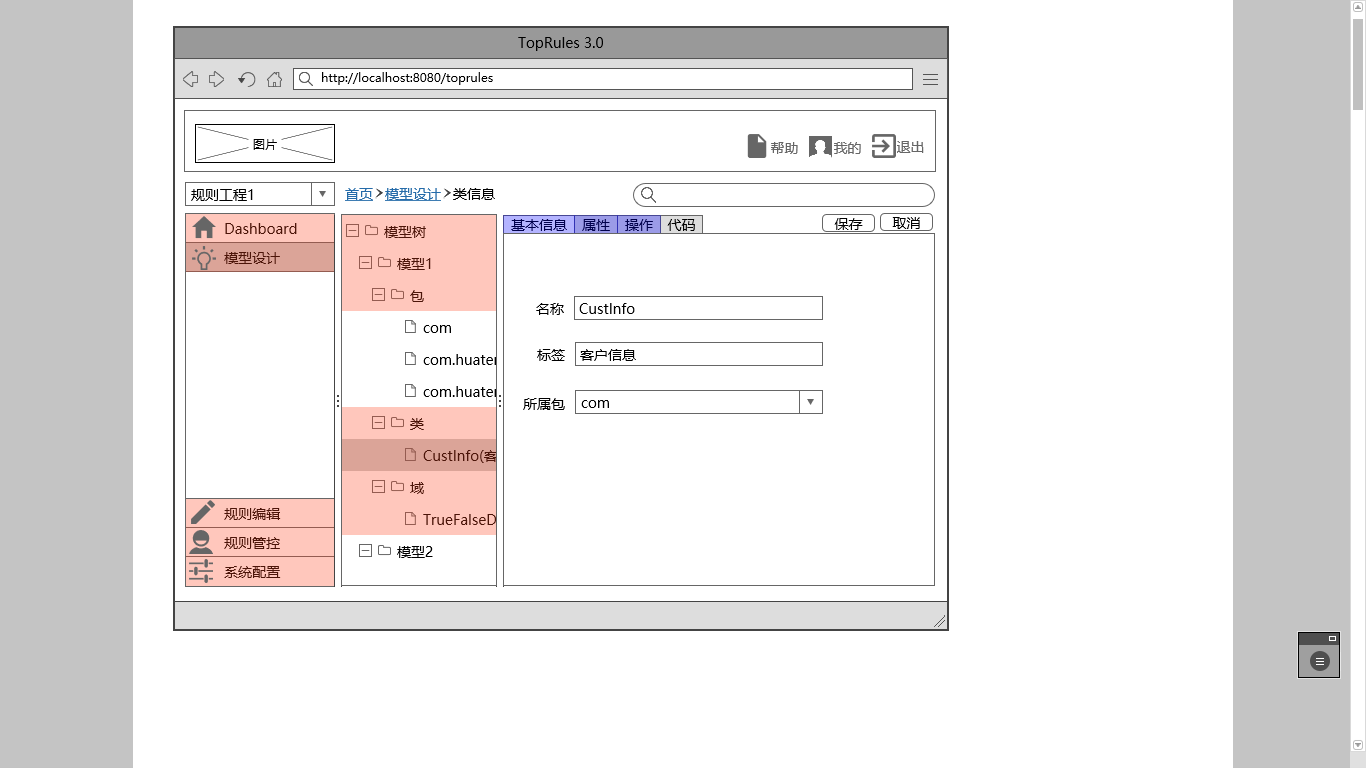
## 类

|  |  |
| --- | --- |
| 功能描述 | 编写业务规则之前需要建立数据模型，该功能为模型设计人员提供可视化的面向对象的数据建模页面（为了适应某些复杂的场景，也保留了现有上传JAR包、通过编码实现规则逻辑及构建动态适配器的建模方式） |
| 入口路径 | 从左侧导航菜单栏点击“模型设计”打开模型树，选择指定模型，如“模型1”,再选中“类”节点 |
| 关键逻辑 | 1、主工作区域的表格中列出当前选中的模型下的所有类信息；  2、类的所属包支持按树形结构或下拉框列出所有包，包名为完整包名，即名称中能体现上、下级依赖关系；  3、新建类后，需要同步更新模型树；  4、从模型树中选择指定类，右侧工作区域显示该类的详细信息，包括基本信息、属性、操作和对应的Java代码；  5、类的“属性”和“操作”Tab中支持批量新建、编辑和删除，点击保存或取消完成操作，删除操作时需要有确认；点击“代码”Tab能自动生成该类的Java代码，技术开发人员也可以直接编辑代码，保存代码时需要进行正确性校验，并能将更新的信息同步到“属性”和“操作”Tab |

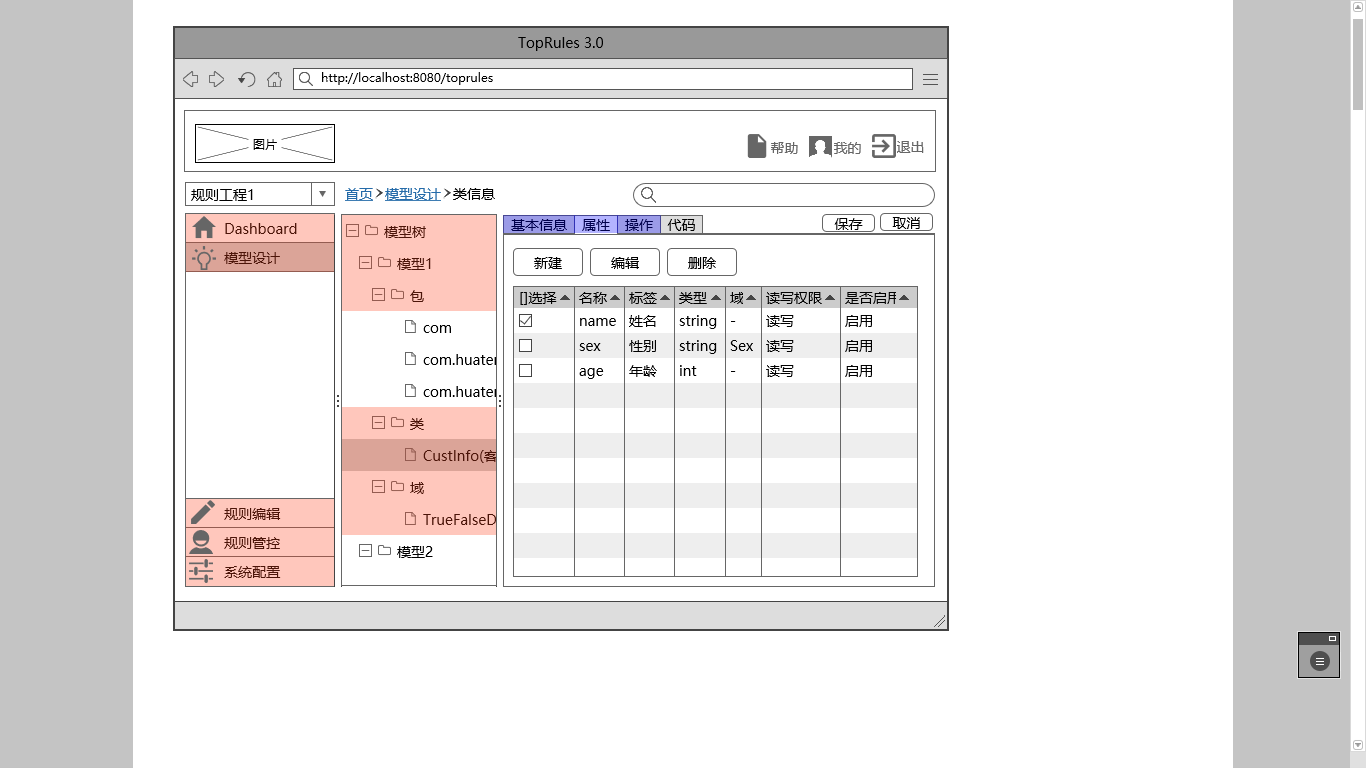
（表3.2.4 类说明）



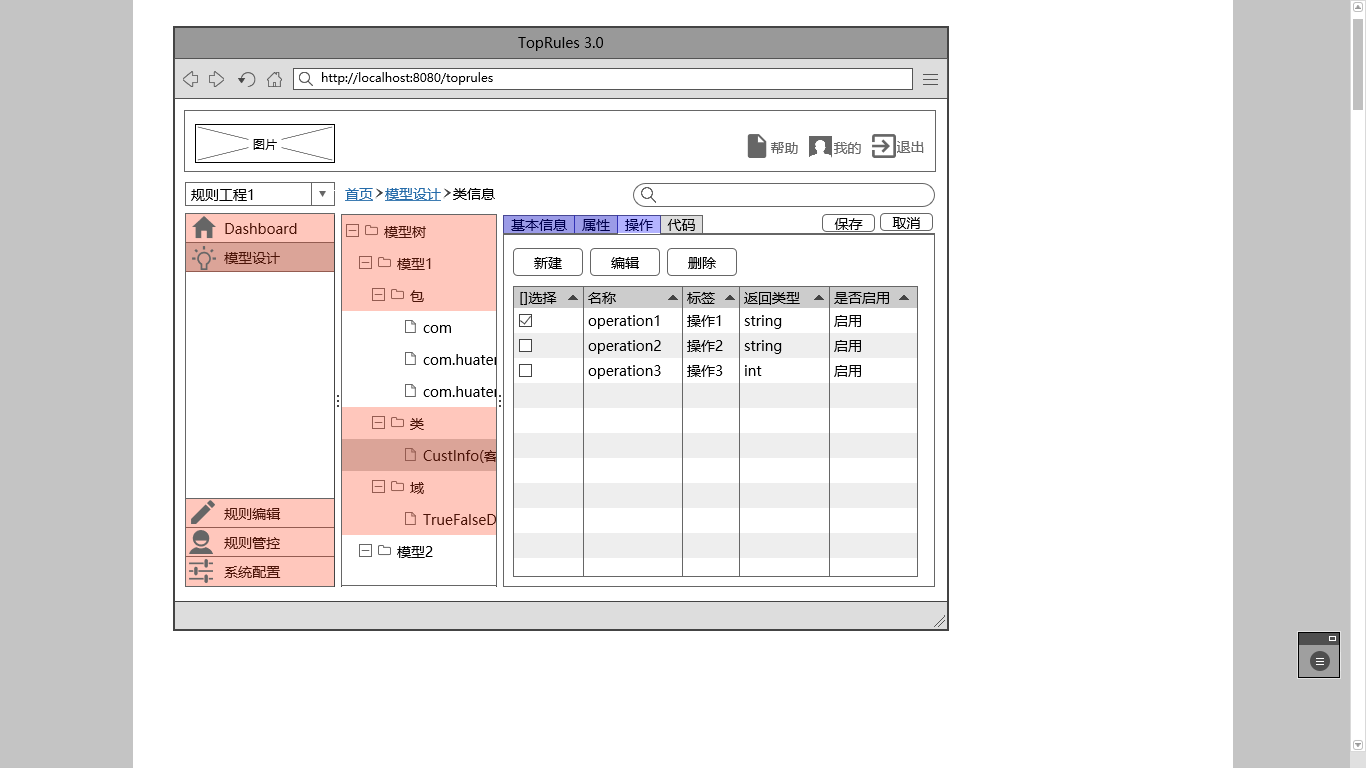
（图3.2.4.1 模型设计-类页面）



（图3.2.4.2 模型设计-类信息-基本信息页面）



（图3.2.4.3 模型设计-类信息-属性页面）

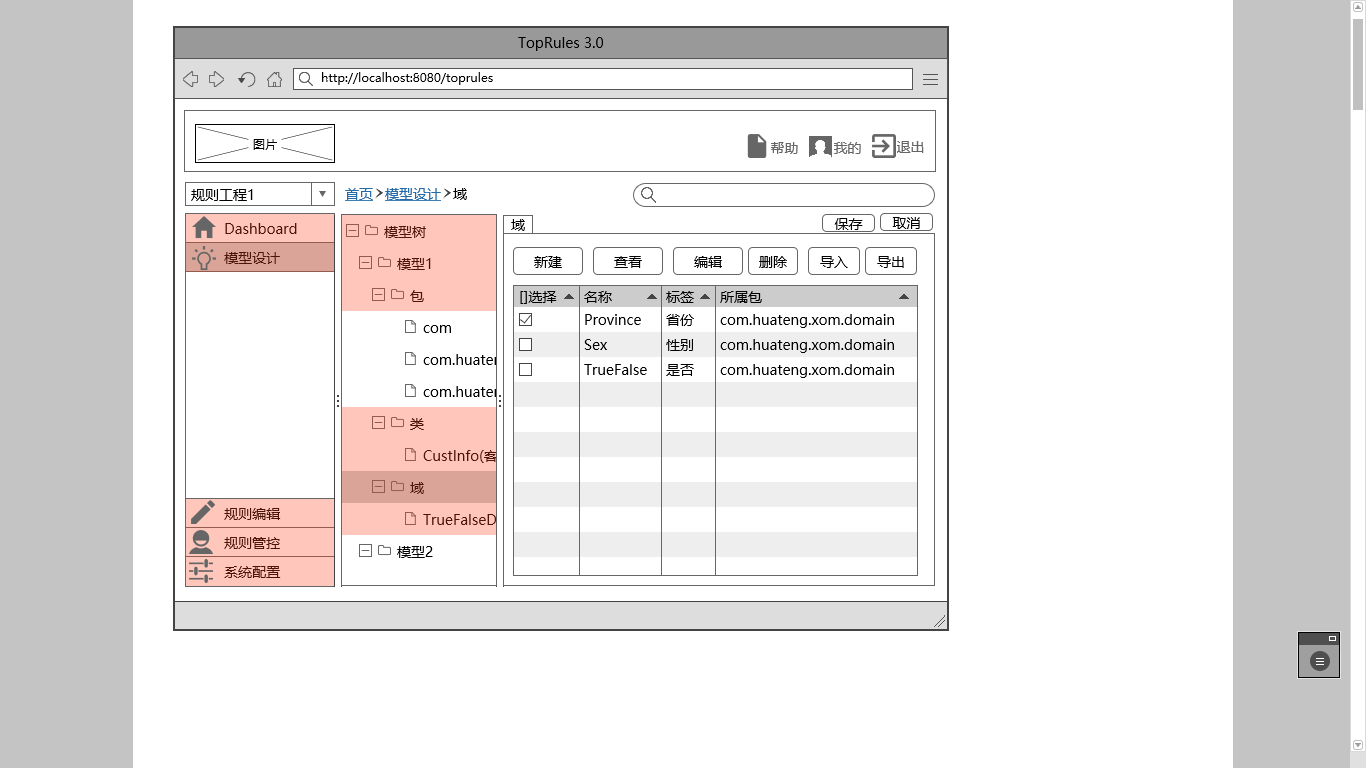


（图3.2.4.4 模型设计-类信息-操作页面）

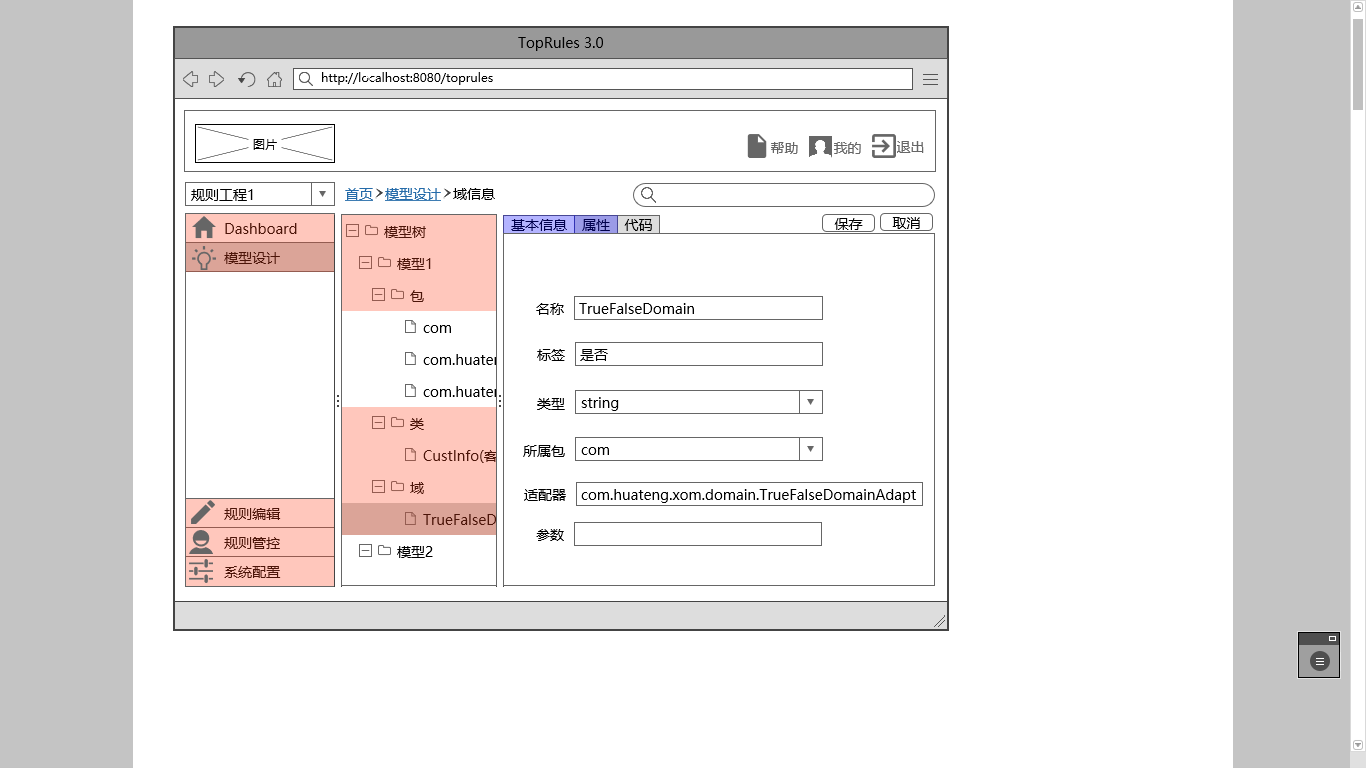
## 域

|  |  |
| --- | --- |
| 功能描述 | 编写业务规则之前需要建立数据模型，该功能为模型设计人员提供可视化的面向对象的数据建模页面（为了适应某些复杂的场景，也保留了现有上传JAR包、通过编码实现规则逻辑及构建动态适配器的建模方式） |
| 入口路径 | 从左侧导航菜单栏点击“模型设计”打开模型树，选择指定模型，如“模型1”,再选中“域”节点 |
| 关键逻辑 | 1. 主工作区域的表格中列出当前选中的模型下的所有域信息； 2. 支持批量新建、编辑或删除域，也支持从Excel导入、导出域及其属性； 3. 在模型树中选中指定域后，右侧主工作区域显示该域的详细信息，域的“基本信息”Tab中，适配器和参数字段只适用于动态域，静态域时为空，当上传的JAR包中包含动态适配器时，上传后应自动更新域的适配器、参数以及对应的域属性值；域的“属性”Tab支持批量新建、编辑或删除操作；点击域的“代码”Tab，自动显示对应的Java代码，技术开发人员也可以直接编辑“代码”Tab，保存代码时应进行正确性校验，并将更新的信息同步到域的其他Tab中。 |

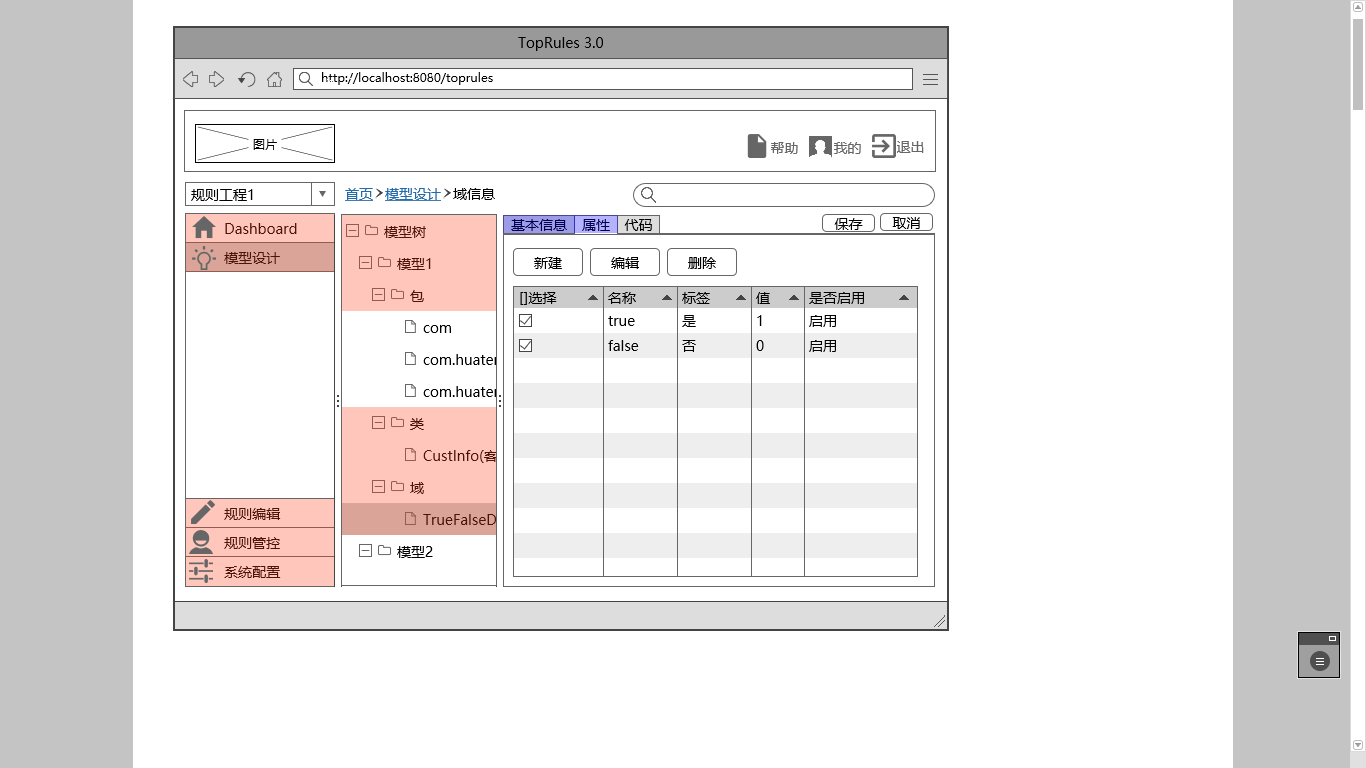
（表3.2.5 域说明）



（图3.2.5.1 模型设计-域页面）



（图3.2.5.2 模型设计-域信息-基本信息页面）



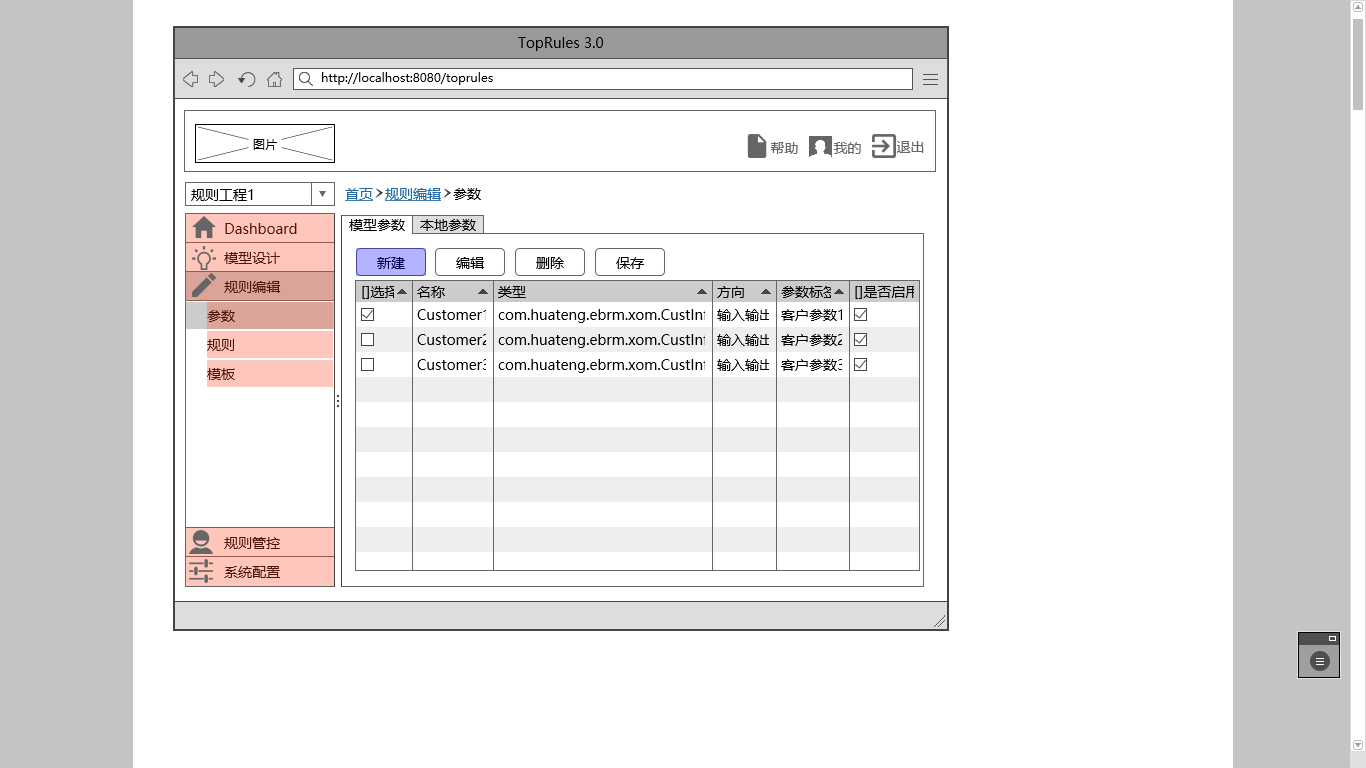
（图3.2.5.3 模型设计-域信息-属性页面）

# 规则编辑

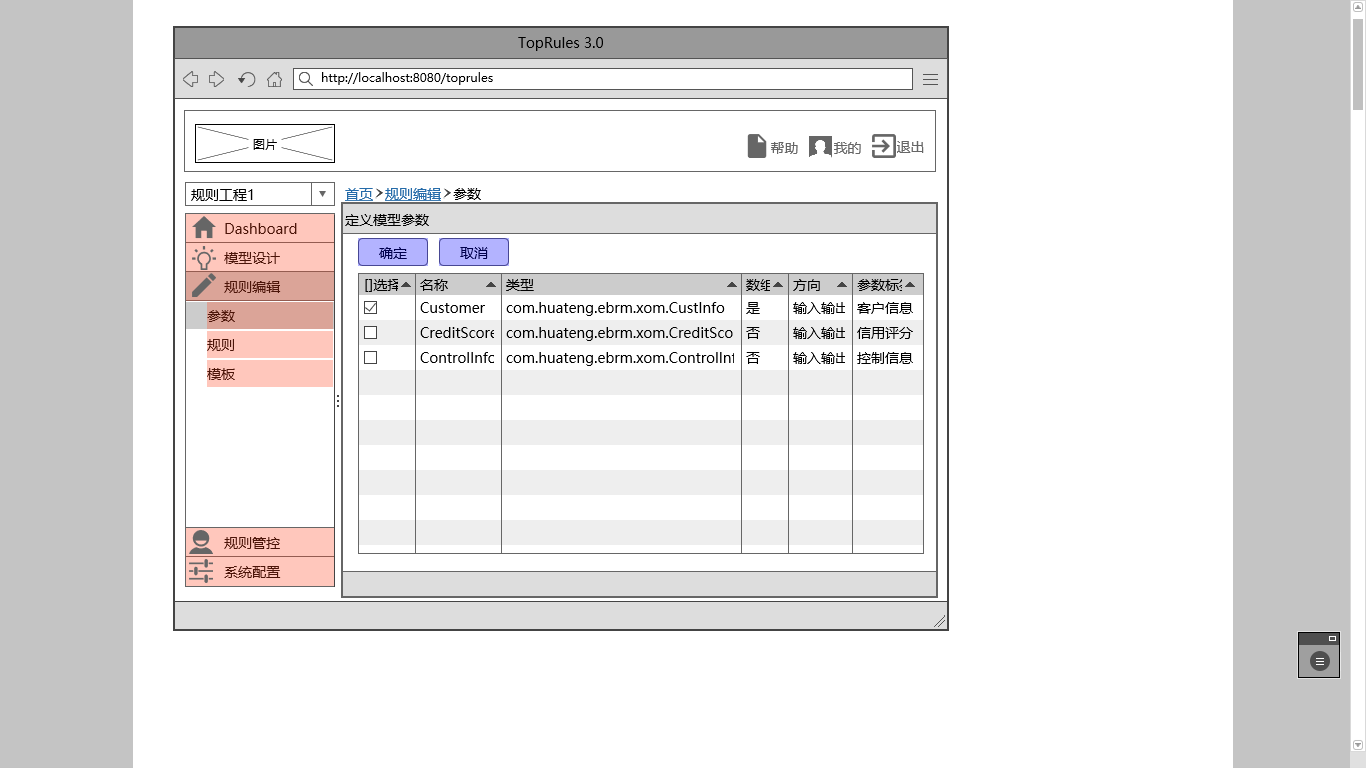
## 参数

|  |  |
| --- | --- |
| 功能描述 | 参数是业务人员编写规则时用到的业务语义，业务人员在编写规则前，需要事先定义规则需要用到的参数，包括模型参数和本地参数，其中模型参数来自Java模型，是技术语义到业务语义的映射。 |
| 入口路径 | 从左侧导航菜单栏点击“规则编辑”默认打开参数页面 |
| 关键逻辑 | 主工作区域的表格中默认列出当前定义的所有模型参数或本地参数，支持批量编辑或删除操作；  点击“新建”按钮，自动加载当前规则工程中所有模型类的信息，用户可直接在表格内批量修改一条或多条参数的“数组”、“方向”或“参数标识”信息。 |

（表3.3.1 参数说明）



（图3.3.1.1 规则编辑-参数-模型参数页面）

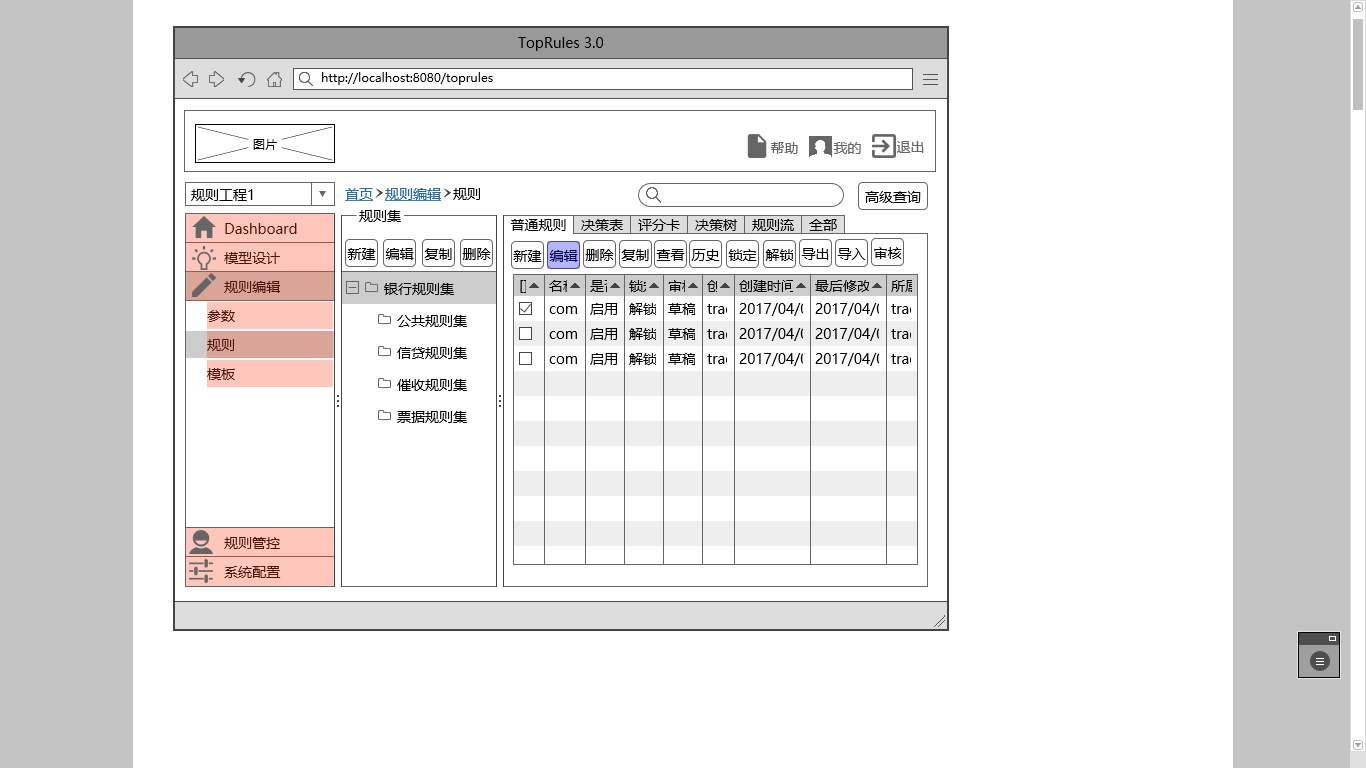


（图3.3.1.2 规则编辑-参数-模型参数-新建页面）

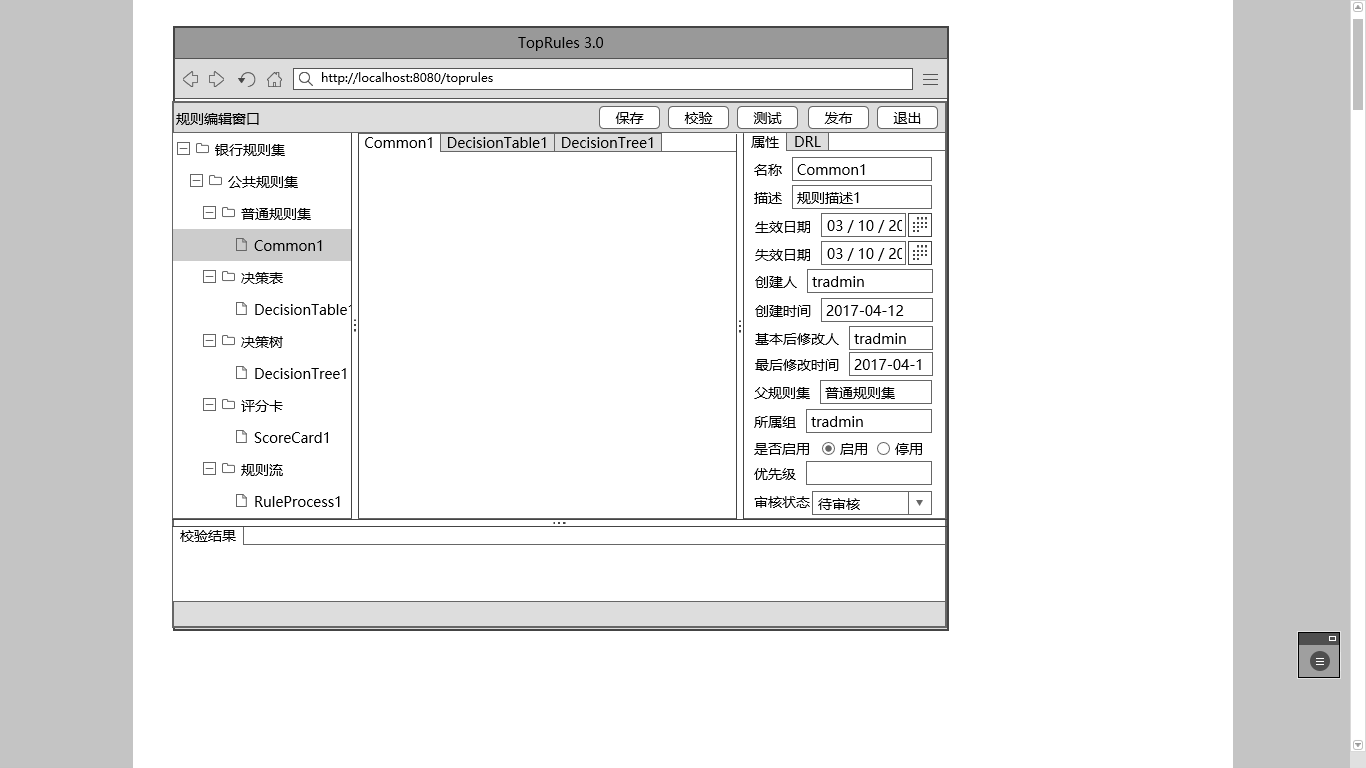
## 规则

|  |  |
| --- | --- |
| 功能描述 | 提供业务规则的可视化编写、校验、审核、测试、发布、导入、导出等操作。 |
| 入口路径 | 从左侧导航菜单栏点击“规则编辑”，选择“规则”菜单 |
| 关键逻辑 | 1. 进入“规则”菜单后，默认显示当前规则工程中所有的规则集（即现有的规则包，后台也沿用现有的规则包），规则集支持嵌套； 2. 从规则集树中选择一个规则集，右侧主工作区域分Tab显示该规则集下的所有普通规则、决策表、评分卡、决策树或规则流，除“编辑”功能外，其他按钮的功能与现有flex版本一致；“全部”Tab中列出所有规则，不作分类，该Tab下的“新建”可不显示或者点击时需要选择规则类型； 3. 规则编辑窗口中，左边的树形菜单显示当前规则所在的规则集及其他规则；点击“校验”按钮，在底部区域显示规则的校验成功或失败提示信息，点击“测试”或“发布”按钮打开当前规则集的测试或发布页面，参见3.4.2发布管理，3.4.3规则测试；点击“退出”返回到规则列表页面。 |

（表3.3.2 规则说明）



（图3.3.2.1 规则编辑-规则页面）

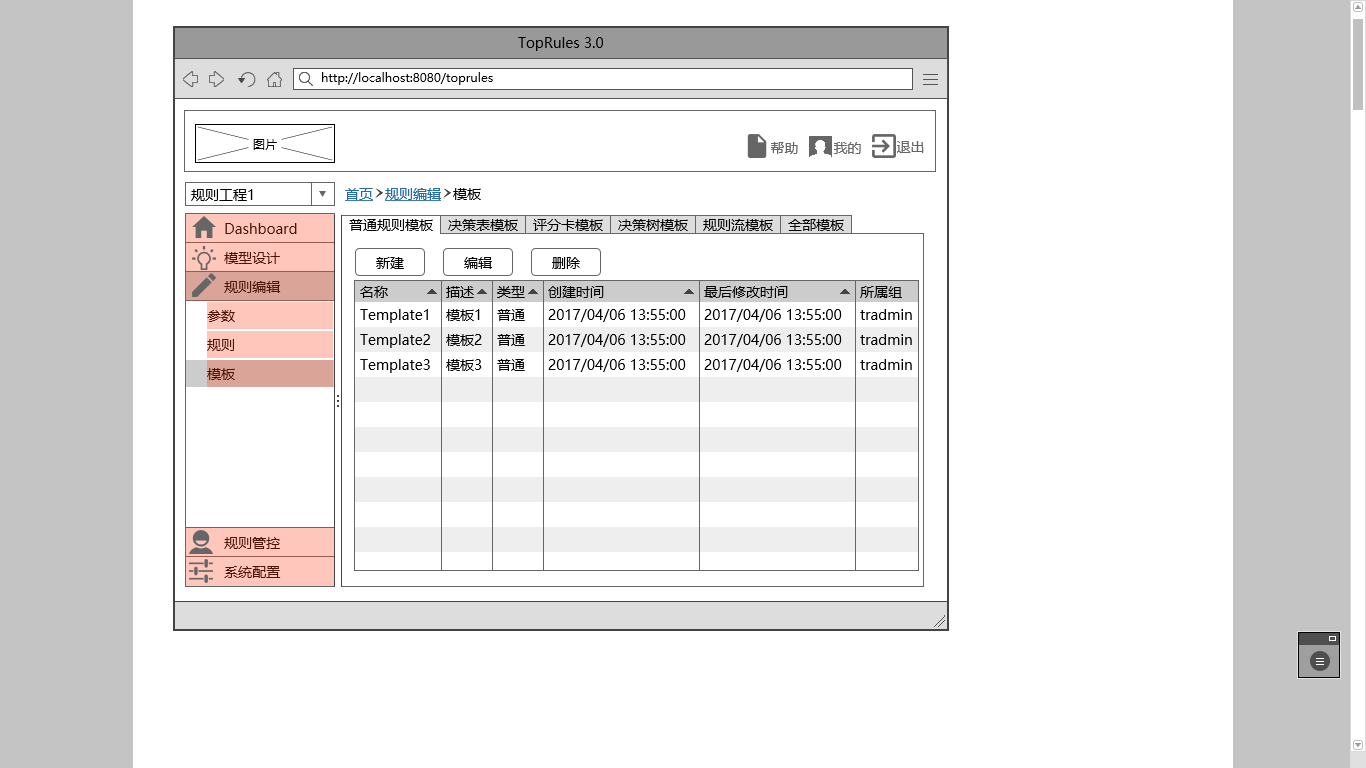


（图3.3.2.2 规则编辑-规则-编辑页面）

## 模板

|  |  |
| --- | --- |
| 功能描述 | 模板为快速创建同一类业务规则提供了通用的实现。 |
| 入口路径 | 从左侧导航菜单栏点击“规则编辑”，选择“模板”菜单 |
| 关键逻辑 | 无 |

（表3.3.3 模板说明）



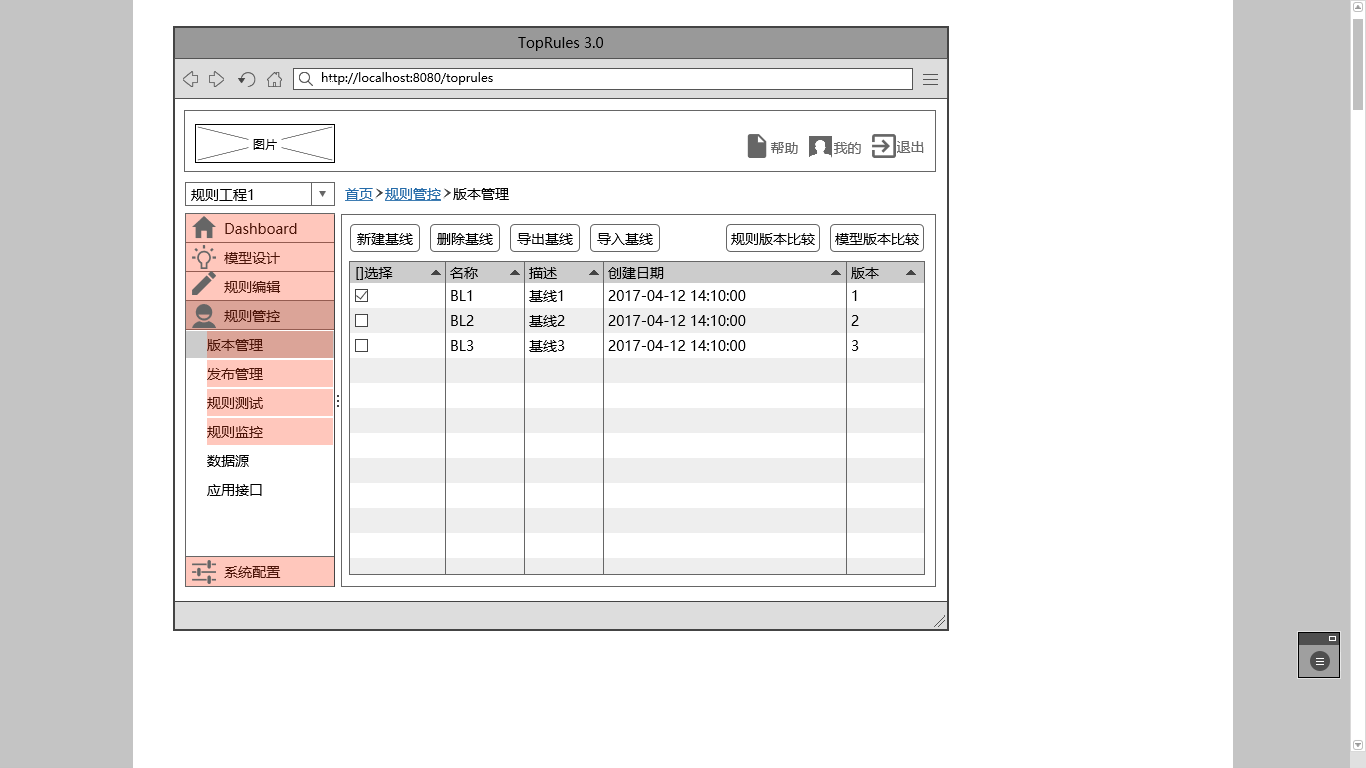
（图3.3.3.1 规则编辑-模板页面）

# 规则管控

## 版本管理

|  |  |
| --- | --- |
| 功能描述 | 对当前规则工程基于基线进行版本管理，支持新建、删除、导出、导出基线及规则和模型的版本比较。 |
| 入口路径 | 从左侧导航菜单栏点击“规则管控”，默认显示“版本管理”菜单 |
| 关键逻辑 | 1. 基线的版本号只保留整数，基线下的规则集版本号由“大版本号.小版本号”组成，其中大版本号与基线的版本号保持一致，小版本号控制规则集的版本变更，每次增加1。 |

（表3.4.1 版本管理说明）

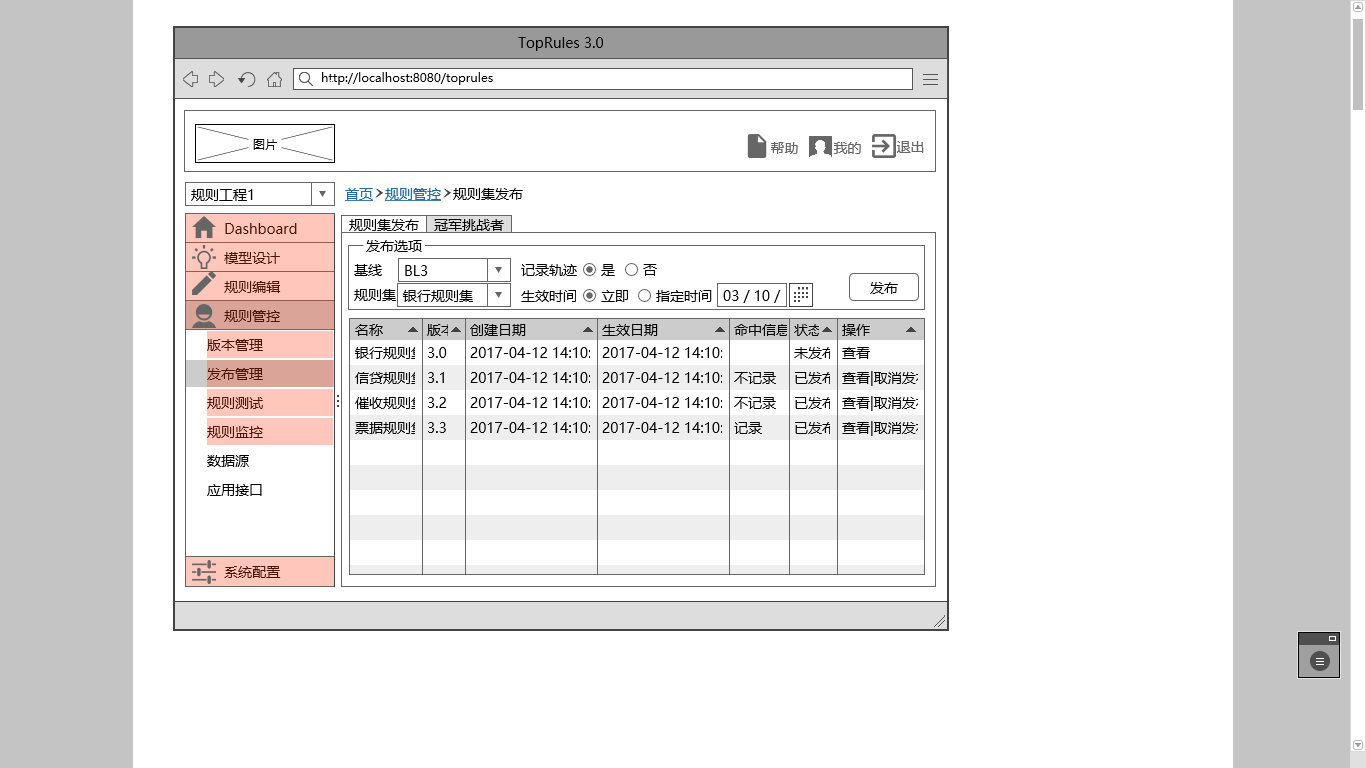


（图3.4.1.1 规则管控-版本管理页面）

## 发布管理

|  |  |
| --- | --- |
| 功能描述 | 规则集必须发布到规则执行引擎才能被外部应用调用，规则集是发布的最小单位，支持规则集发布、查看和取消发布。 |
| 入口路径 | 从左侧导航菜单栏点击“规则管控”，打开“发布管理”菜单 |
| 关键逻辑 | 1. 规则集版本号由“大版本号.小版本号”组成，其中大版本号与基线的版本号保持一致，小版本号控制规则集的版本变更，每次增加1；规则集发布后根据已有的版本号自动生成规则集新的版本号，默认从X.0开始； 2. 点击“操作”栏中的“查看”，可查看当前规则集中的规则列表 |

（表3.4.2 发布管理说明）

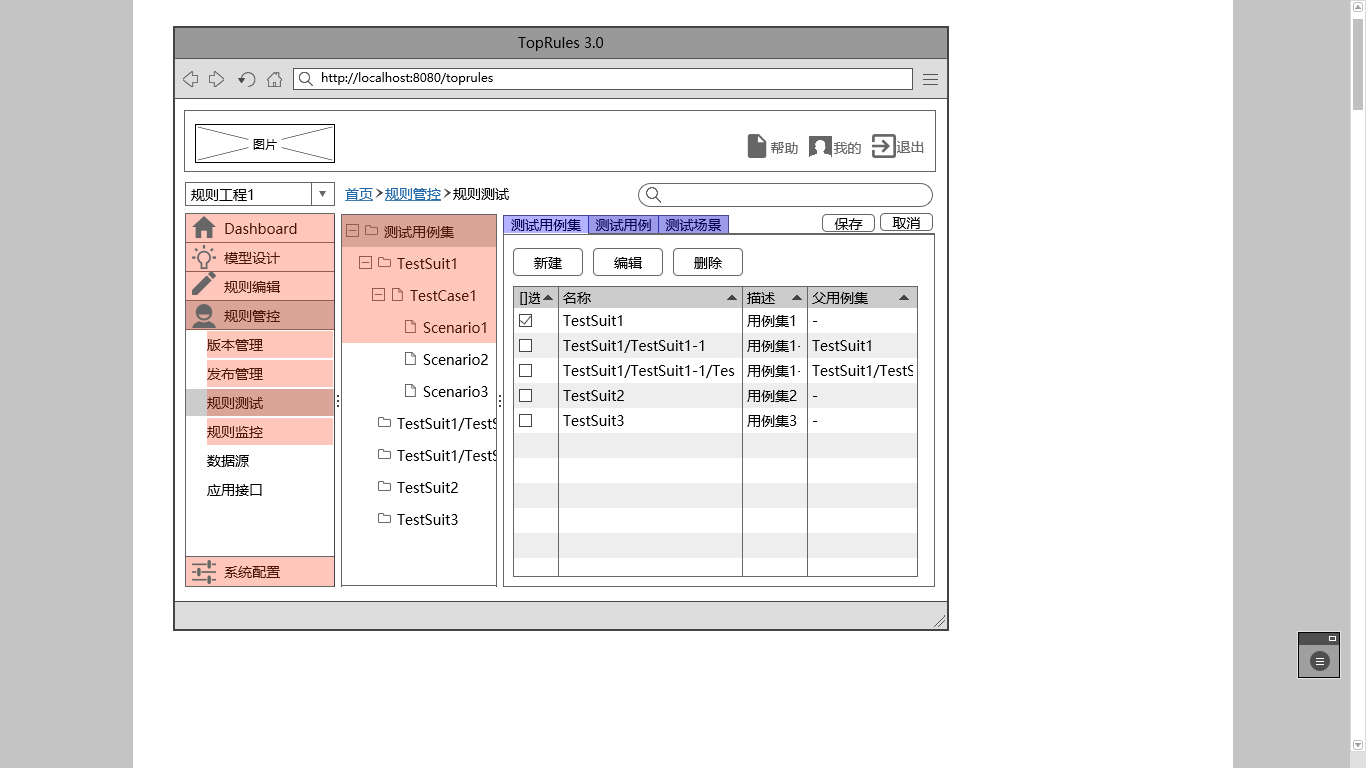


（图3.4.2.1 规则管控-发布管理-规则集发布页面）

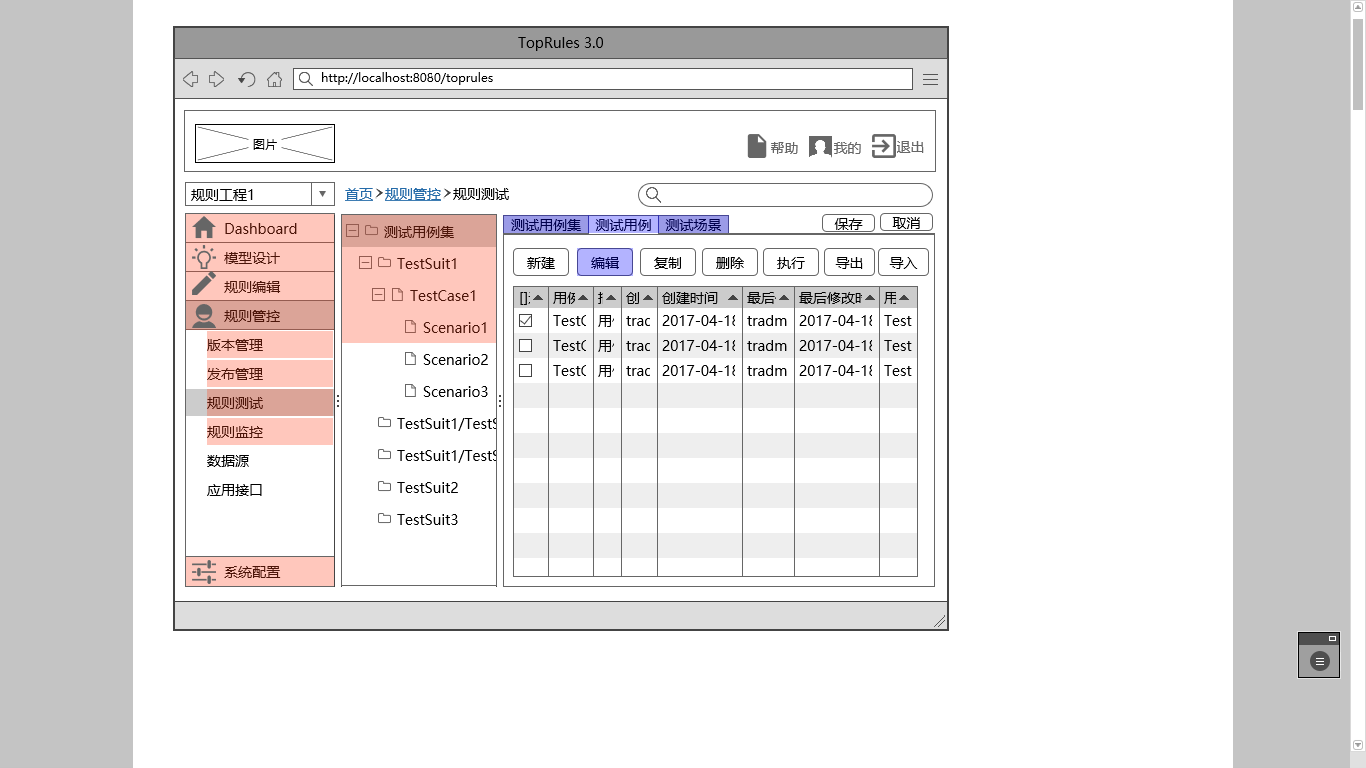
## 规则测试

|  |  |
| --- | --- |
| 功能描述 | 对编写的规则进行业务逻辑验证，可以测试单个规则或者整个规则集。 |
| 入口路径 | 从左侧导航菜单栏点击“规则管控”，打开“规则测试”菜单 |
| 关键逻辑 | 1. 测试用例集是一个或多个测试用例的集合，测试用例集可以嵌套，其名称中应体现上级测试用例，用“/”分隔，类似包结构；测试场景是测试用例的一个实例，一个测试用例包含有多个测试场景； 2. 选择测试用例集树，右侧的主工作区域显示当前规则工程中的所有测试用例集、测试用例和测试场景； 3. 选择一个测试用例集，如TestSuit1，右侧的主工作区域显示该测试用例集及其包含的测试用例和测试场景；选择一个测试用例或测试场景，类似； 4. “测试用例”Tab中，点击“编辑”按钮打开用例编辑窗口，用例编辑中可以选择一个或多个规则，也可以规则集，用例编辑窗口中可直接对当前用例进行测试，也可以在用例列表中选择多个用例测试，测试用例可以进行复制、导出、导入操作。 |

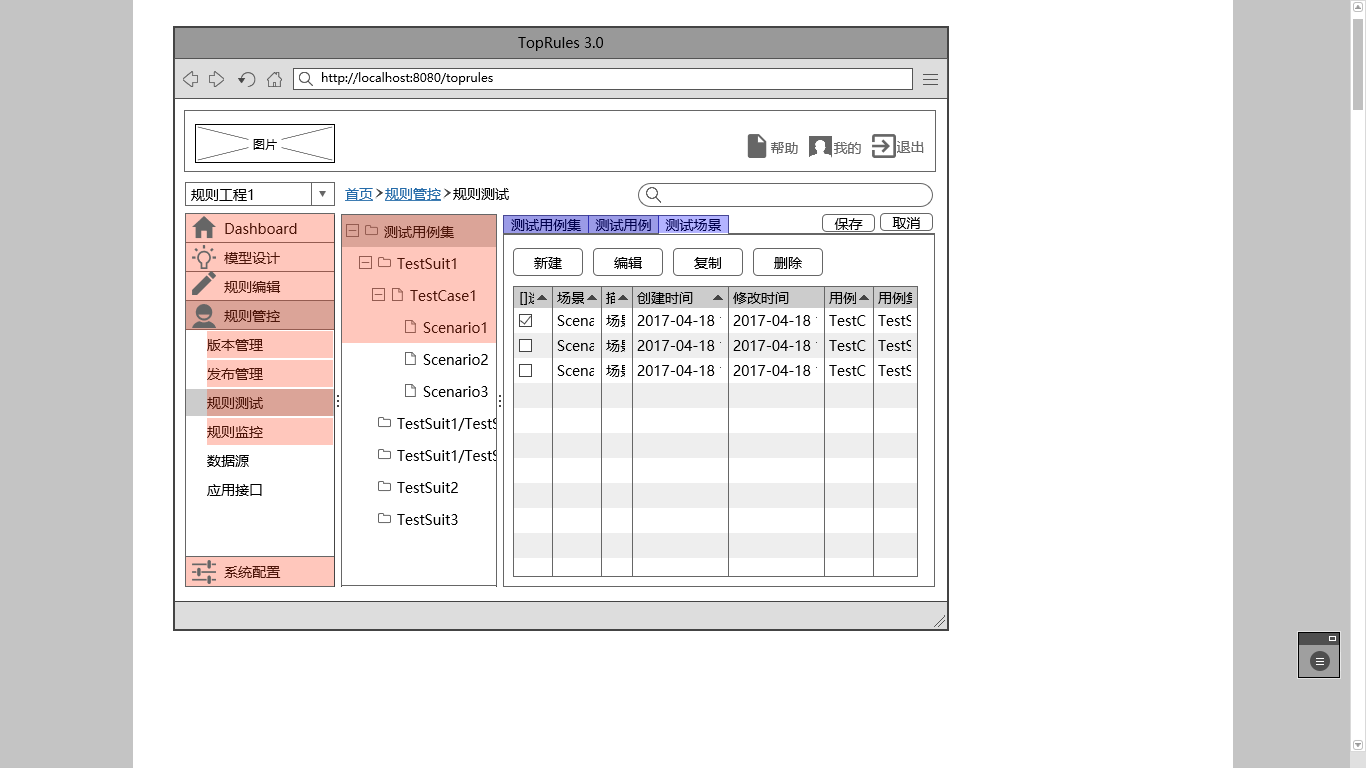
（表3.4.3 规则测试说明）



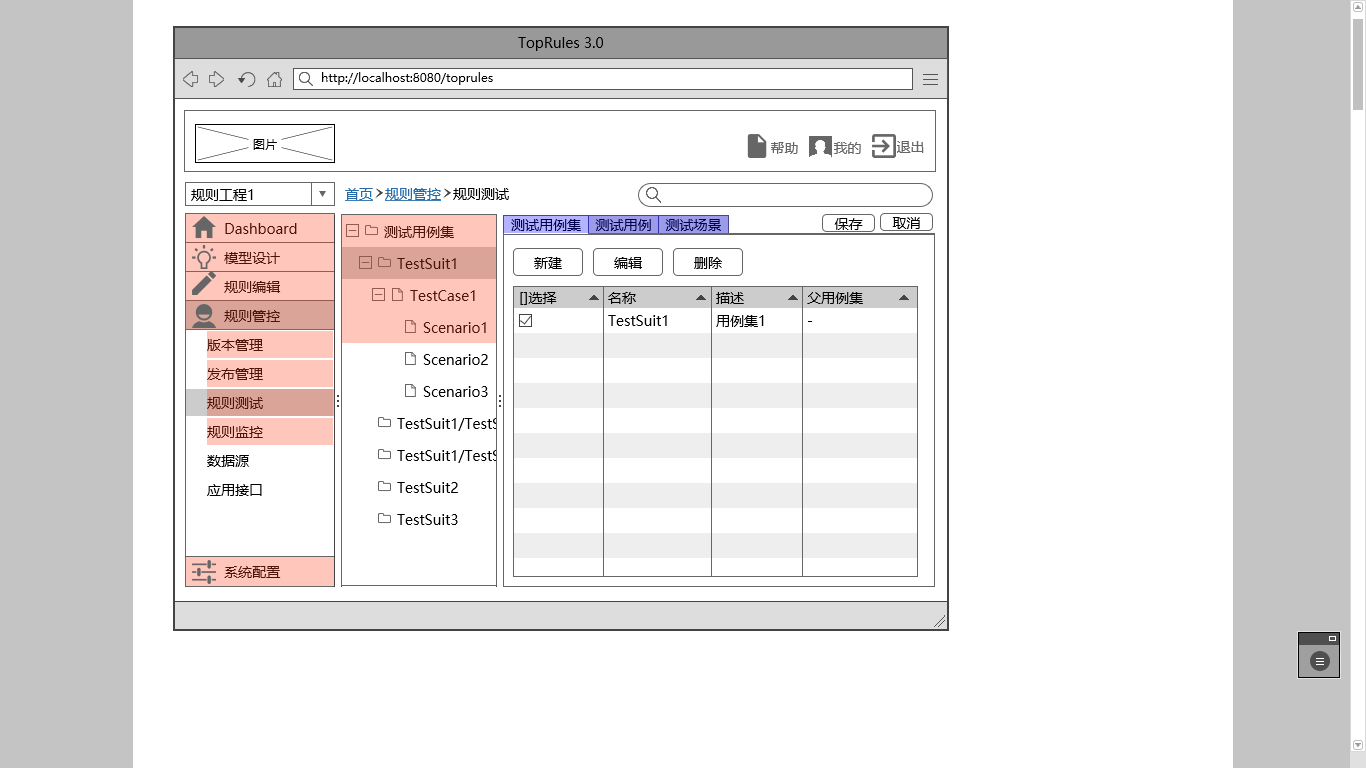
（图3.4.3.1 规则管控-规则测试-测试用例集页面）



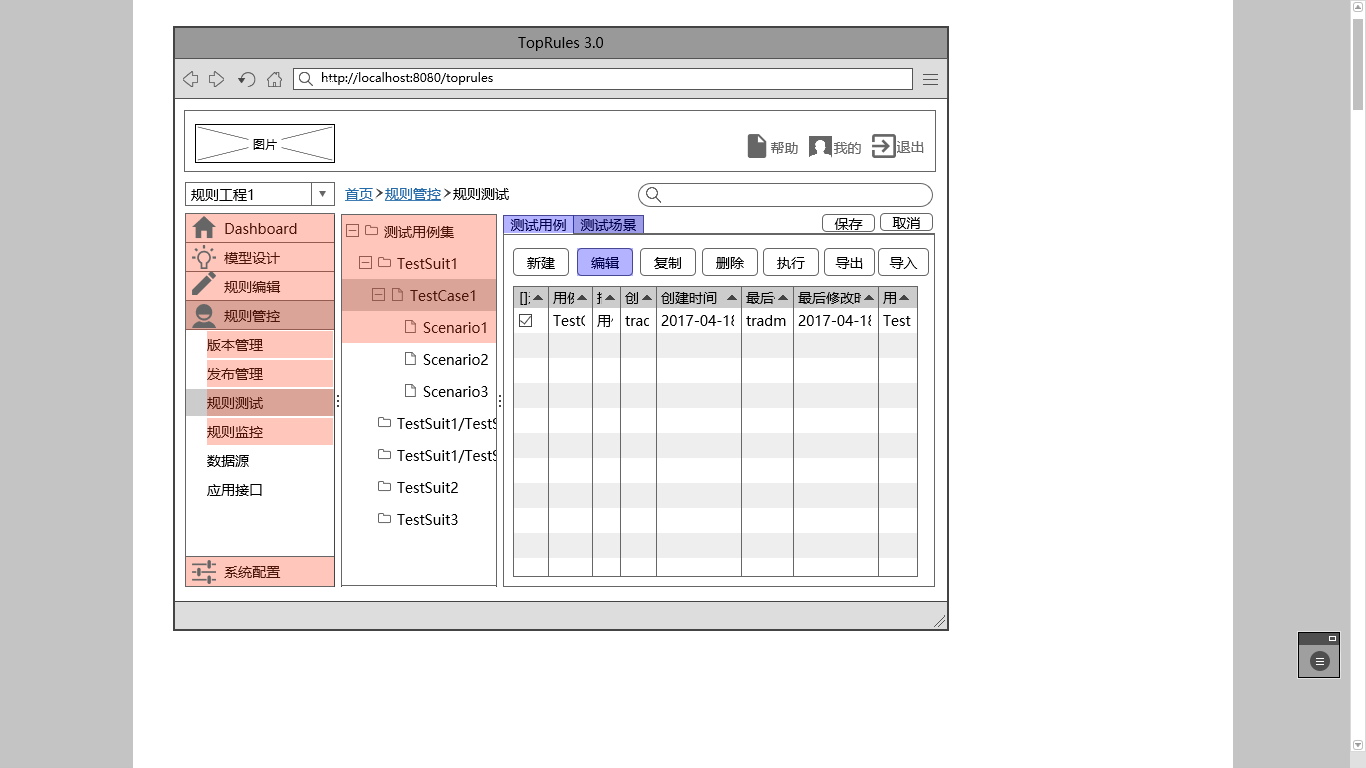
（图3.4.3.2 规则管控-规则测试-测试用例集-测试用例页面）



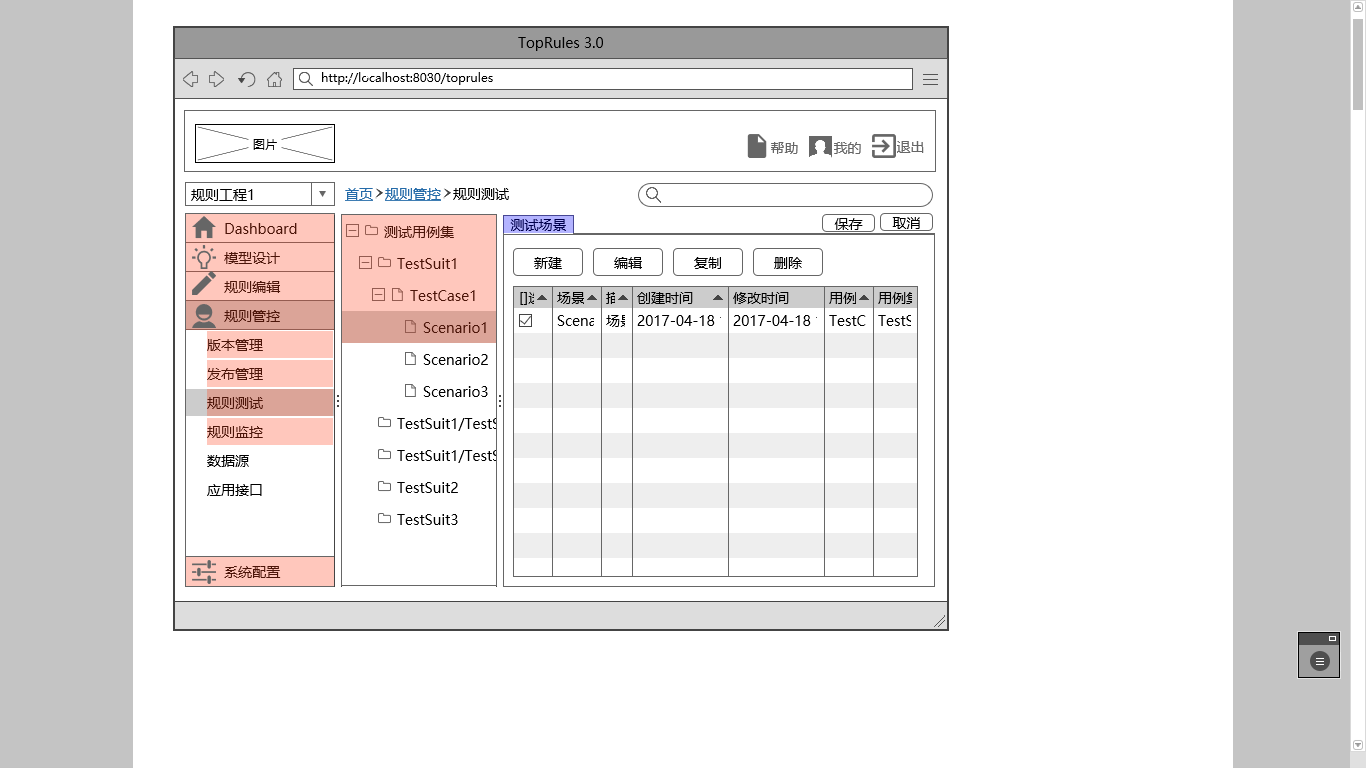
（图3.4.3.3 规则管控-规则测试-测试用例集-测试场景页面）



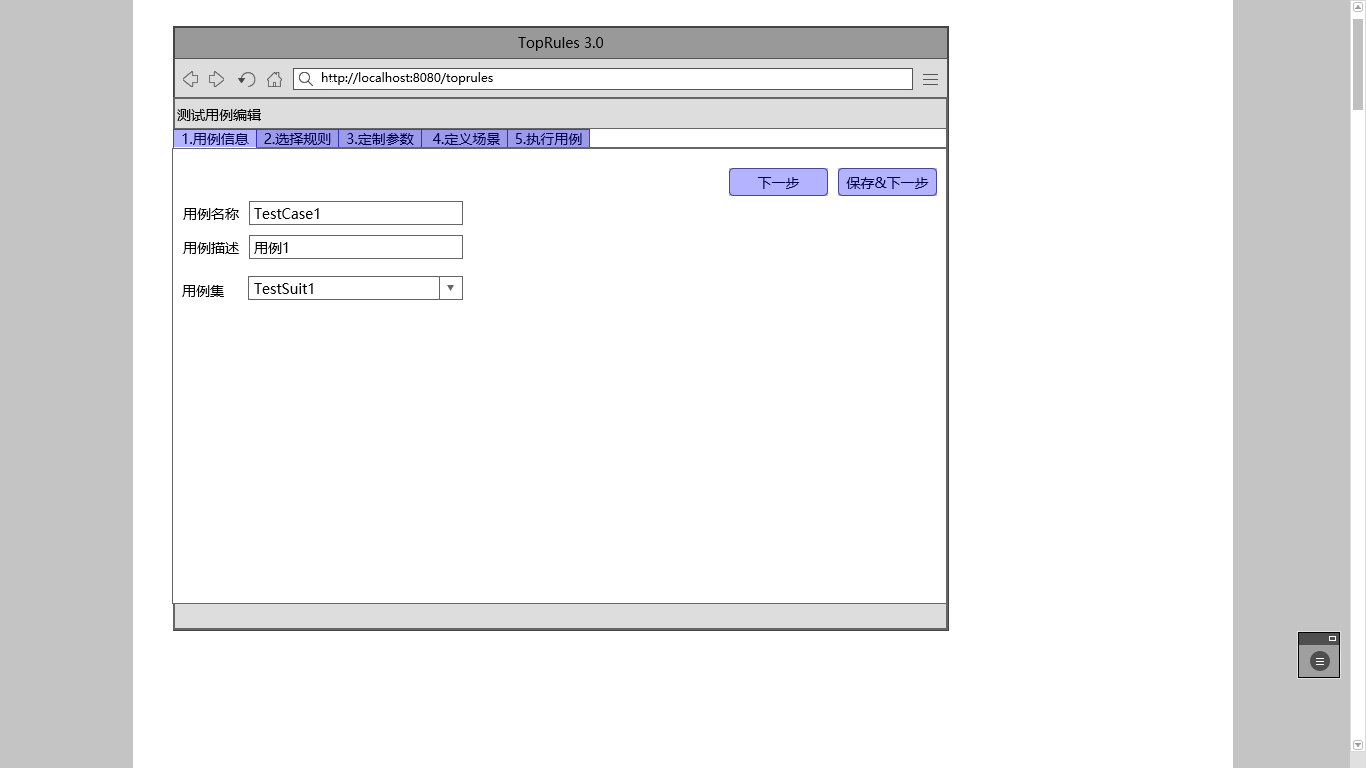
（图3.4.3.4 规则管控-规则测试-测试用例集-TestSuit1页面）



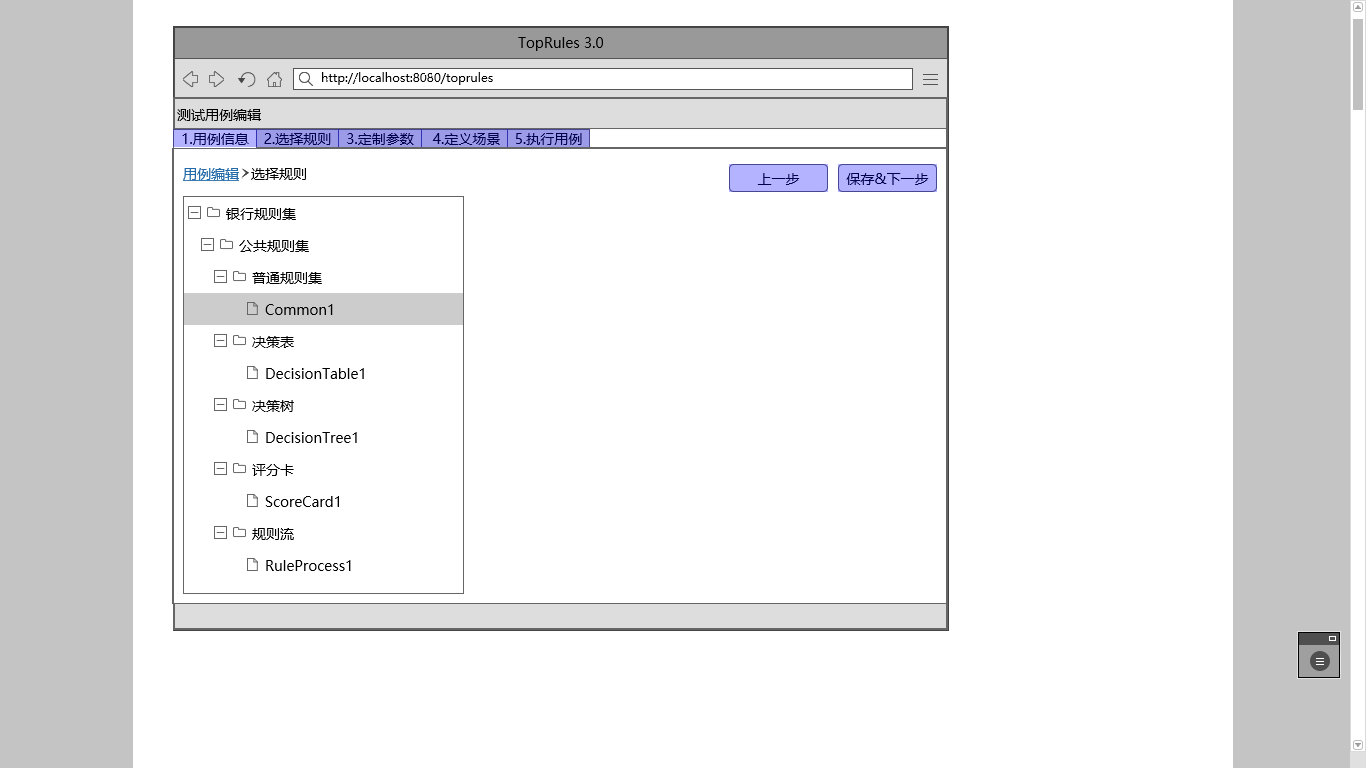
（图3.4.3.5 规则管控-规则测试-测试用例集-TestSuit1-TestCase1页面）



（图3.4.3.6 规则管控-规则测试-测试用例集-TestSuit1-TestCase1-Scenario1页面）



（图3.4.3.7 规则管控-规则测试-测试用例集-测试用例-编辑-用例信息页面）



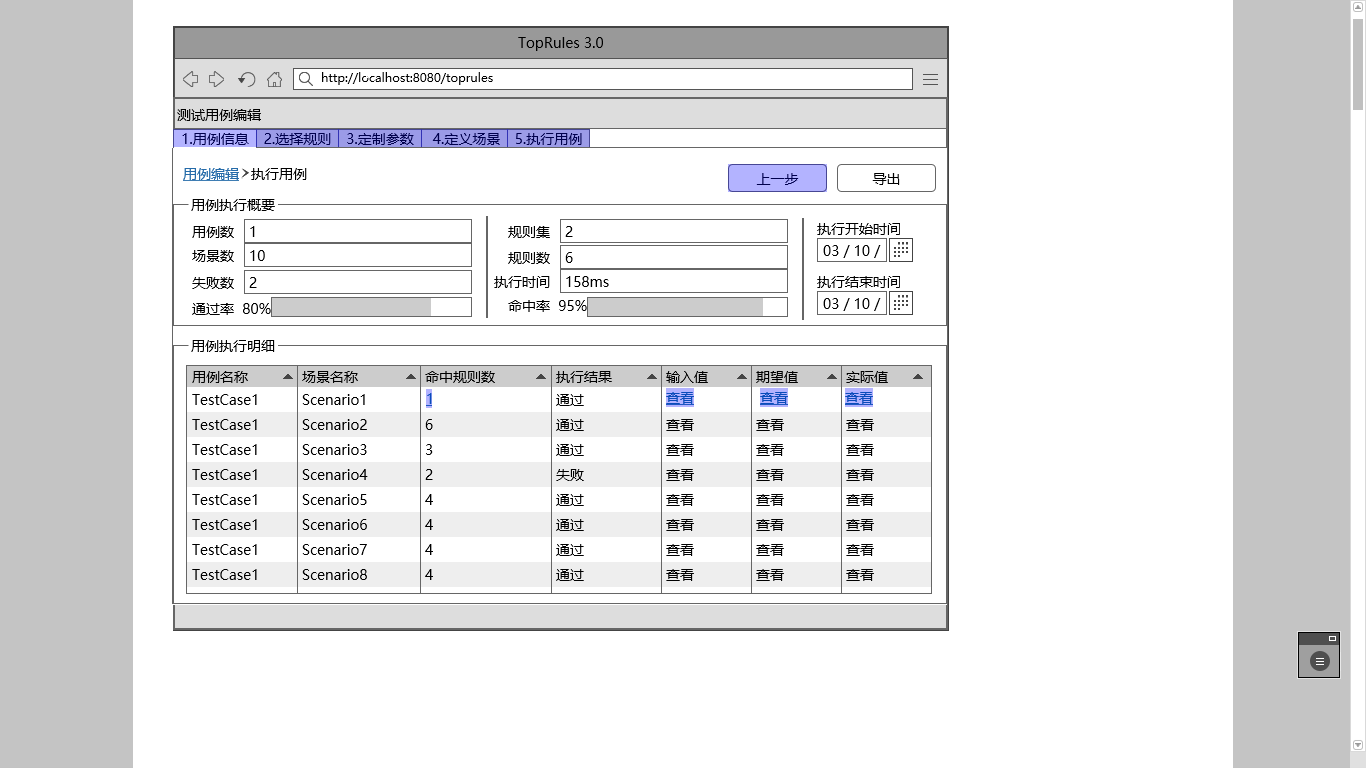
（图3.4.3.8 规则管控-规则测试-测试用例集-测试用例-编辑-选择规则页面）



（图3.4.3.9 规则管控-规则测试-测试用例集-测试用例-编辑-定制参数页面）



（图3.4.3.10 规则管控-规则测试-测试用例集-测试用例-编辑-定义场景页面）

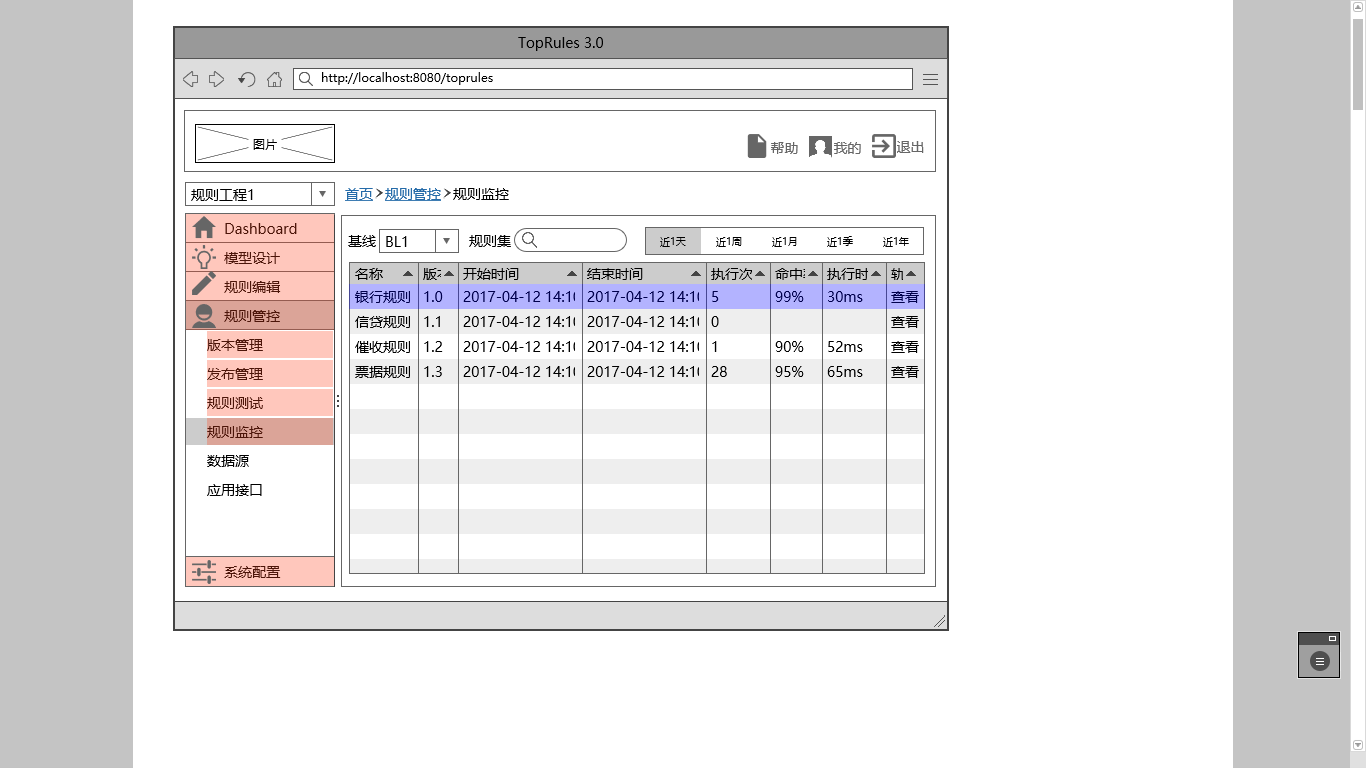


（图3.4.3.11 规则管控-规则测试-测试用例集-测试用例-编辑-执行用例页面）

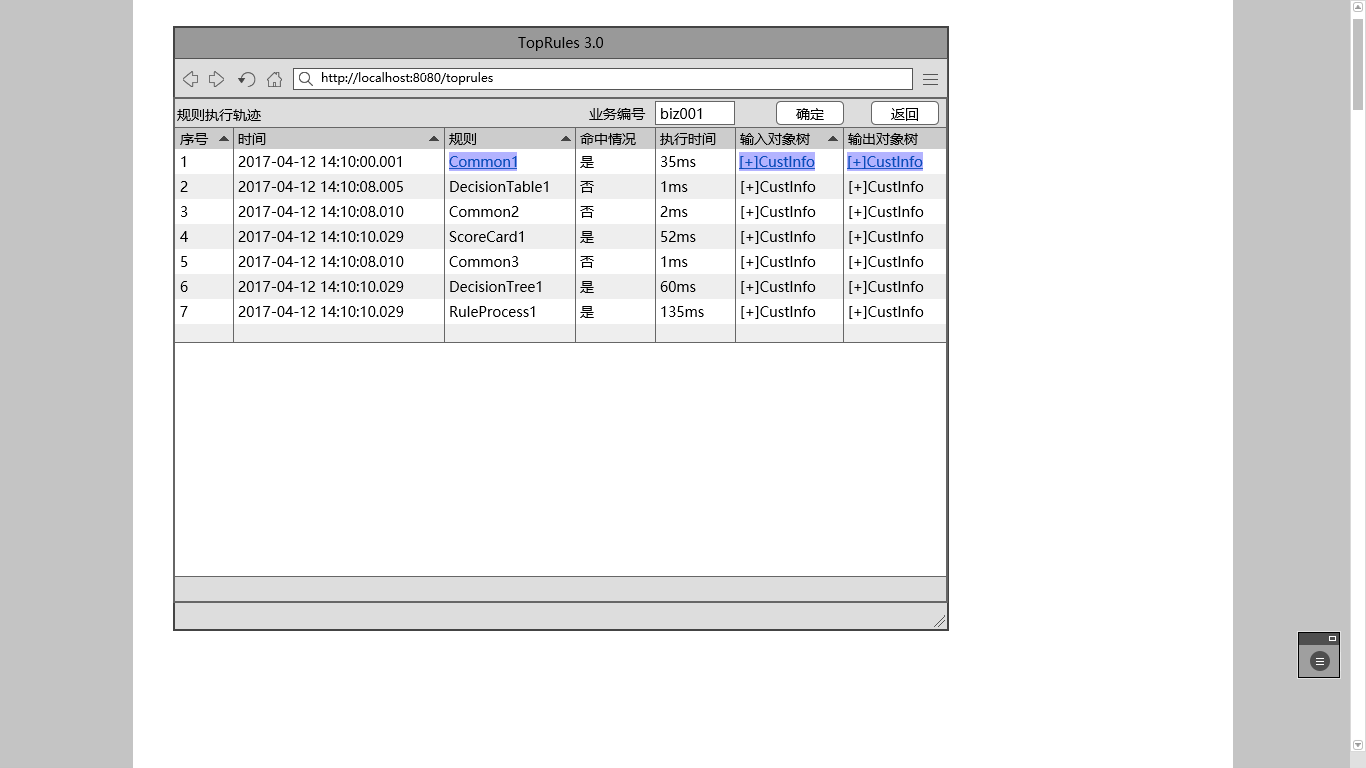
## 规则监控

|  |  |
| --- | --- |
| 功能描述 | 支持监控规则集的运行情况和执行轨迹。 |
| 入口路径 | 从左侧导航菜单栏点击“规则管控”，打开“规则监控”菜单 |
| 关键逻辑 | 1. 规则集的执行轨迹按业务编号来查看； 2. 规则执行轨迹中点击“规则”、“输入对象树”、“输出对象树”，可以在当前行展开详细信息，再次点击可以收缩关闭，其中，输入对象树和输出对象树都可以包含多个对象及其嵌套关系。 |

（表3.4.4 规则监控说明）



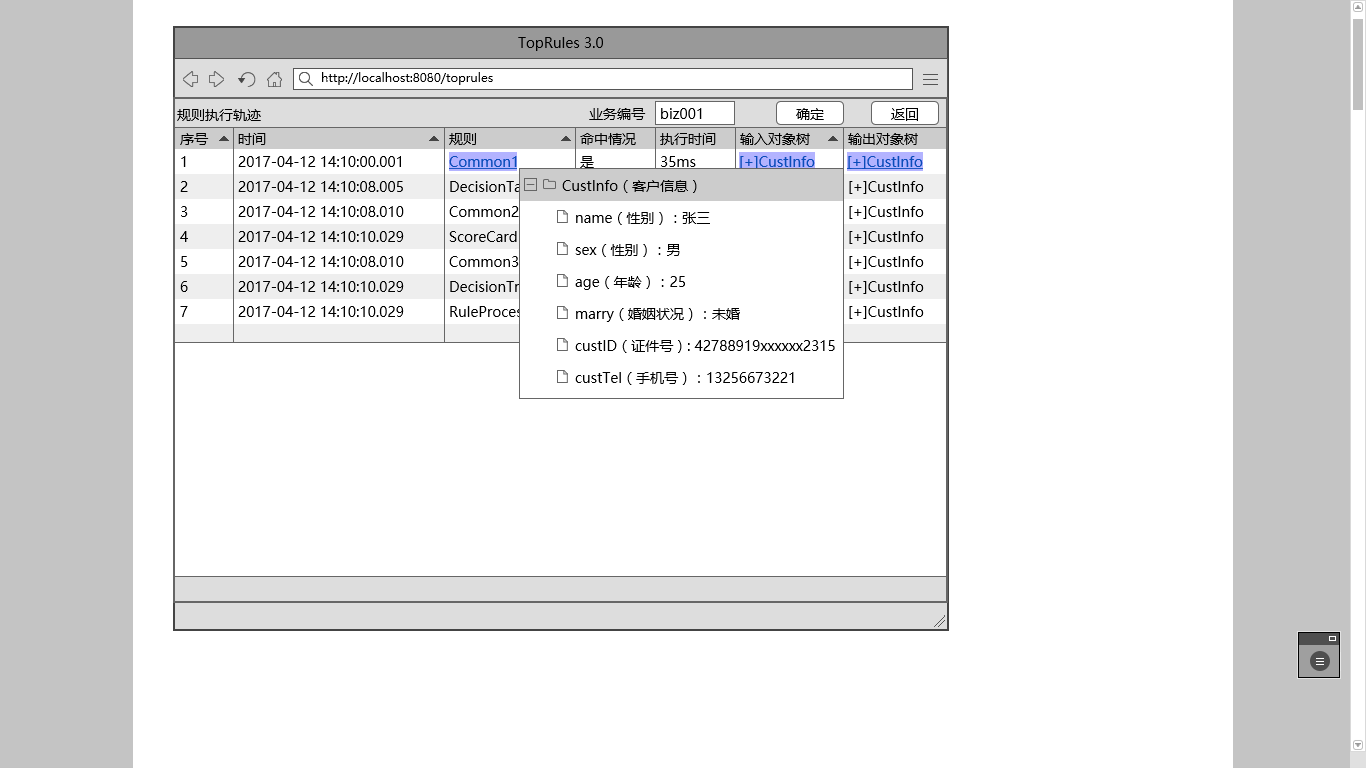
（图3.4.4.1 规则管控-规则监控页面）



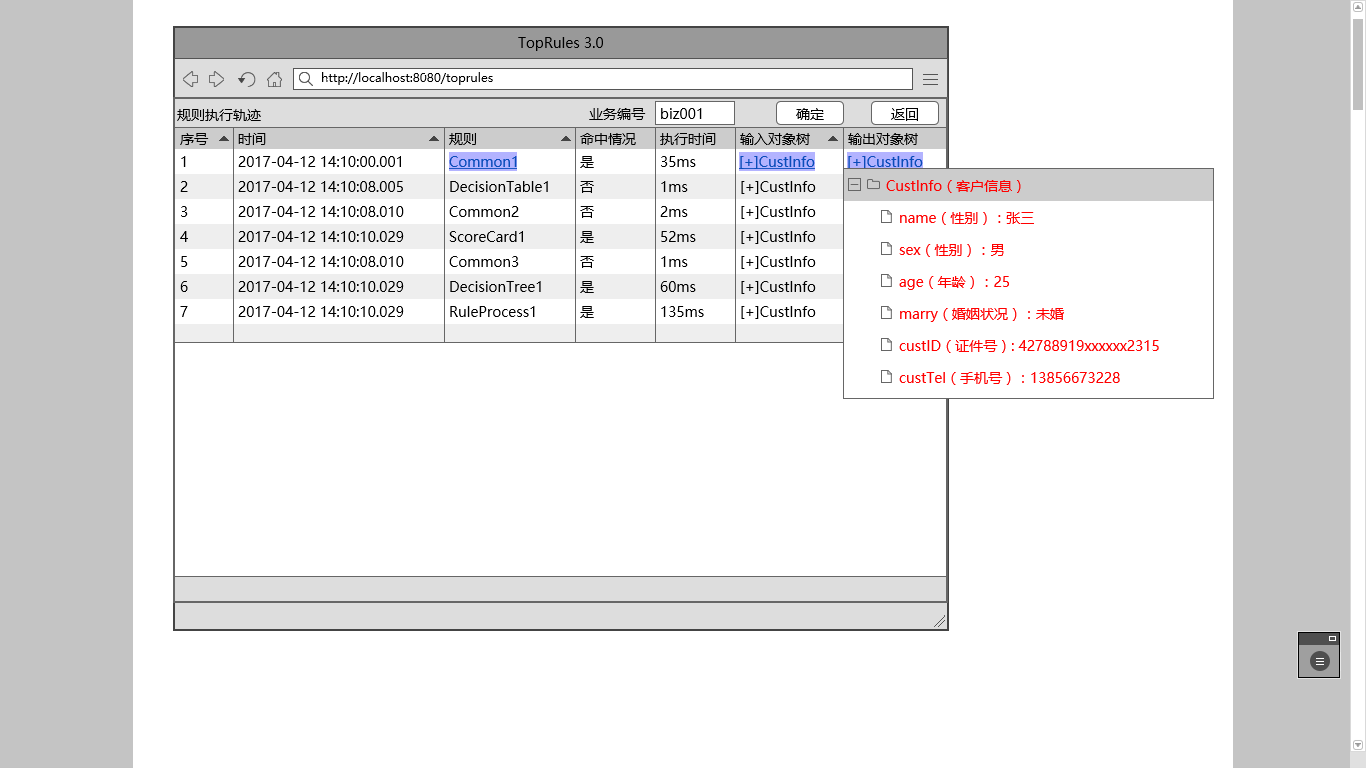
（图3.4.4.2 规则管控-规则监控-规则执行轨迹页面）



（图3.4.4.3 规则管控-规则监控-规则执行轨迹1页面）



（图3.4.4.4 规则管控-规则监控-规则执行轨迹2页面）

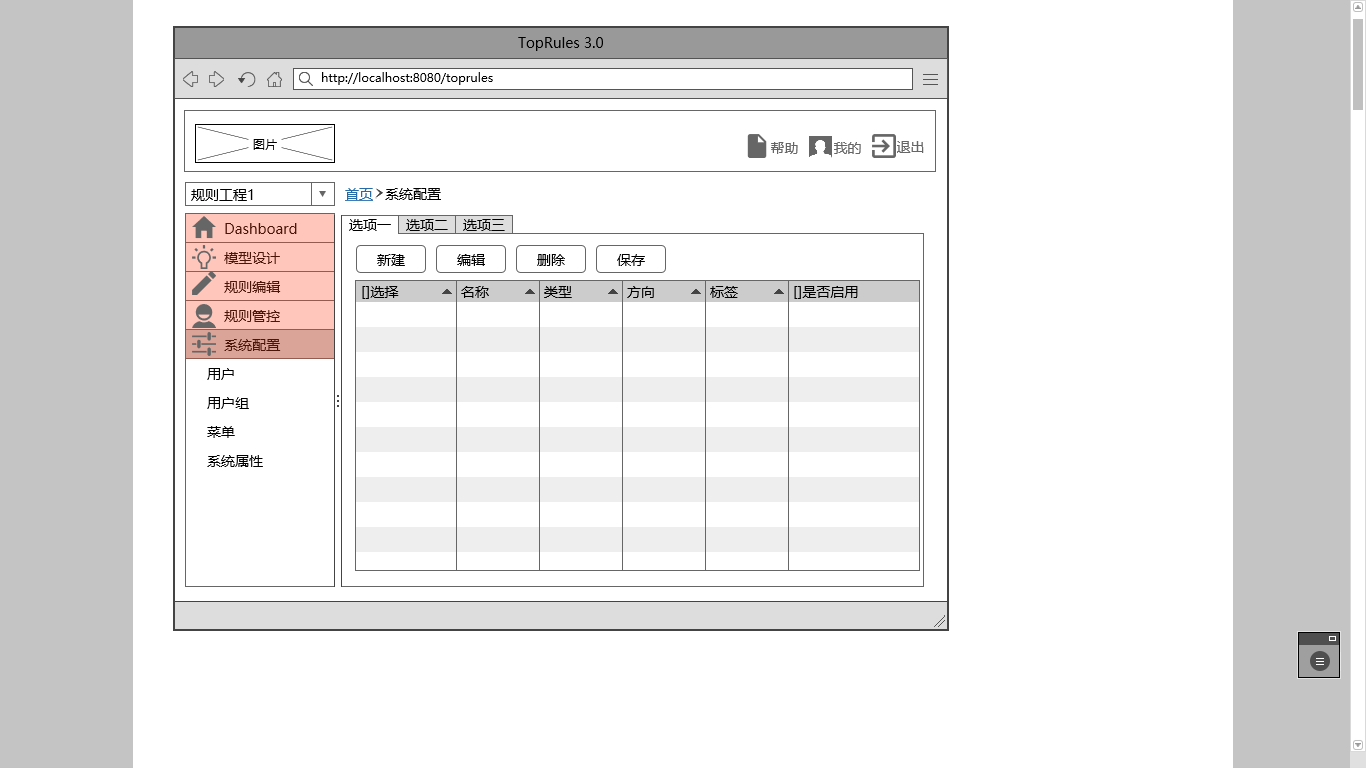


（图3.4.4.5 规则管控-规则监控-规则执行轨迹3页面）

# 系统配置

|  |  |
| --- | --- |
| 功能描述 | 包括用户、用户组、菜单及权限配置，以及系统属性设置 |
| 入口路径 | 从左侧导航菜单栏点击“系统配置”，再选择菜单进入 |
| 关键逻辑 | 1. 页面设计及处理逻辑参看现有flex版本，逻辑不变。 |

（表3.5 系统配置说明）



（图3.5.1 系统配置页面）

# 附录

## 时间格式字符串

日期和时间格式由日期和时间模式 字符串指定。在日期和时间模式字符串中，未加引号的字母 'A' 到 'Z' 和 'a' 到 'z' 被解释为模式字母，用来表示日期或时间字符串元素。文本可以使用单引号 (') 引起来，以免进行解释。"''" 表示单引号。所有其他字符均不解释；只是在格式化时将它们简单复制到输出字符串，或者在解析时与输入字符串进行匹配。

定义了以下模式字母（所有其他字符 'A' 到 'Z' 和 'a' 到 'z' 都被保留）：

|  |  |  |  |
| --- | --- | --- | --- |
| **字母** | **日期或时间元素** | **表示** | **示例** |
| G | Era 标志符 | [Text](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#text) | AD |
| y | 年 | [Year](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#year) | 1996; 96 |
| M | 年中的月份 | [Month](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#month) | July; Jul; 07 |
| w | 年中的周数 | [Number](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#number) | 27 |
| W | 月份中的周数 | [Number](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#number) | 2 |
| D | 年中的天数 | [Number](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#number) | 189 |
| d | 月份中的天数 | [Number](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#number) | 10 |
| F | 月份中的星期 | [Number](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#number) | 2 |
| E | 星期中的天数 | [Text](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#text) | Tuesday; Tue |
| a | Am/pm 标记 | [Text](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#text) | PM |
| H | 一天中的小时数（0-23） | [Number](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#number) | 0 |
| k | 一天中的小时数（1-24） | [Number](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#number) | 24 |
| K | am/pm 中的小时数（0-11） | [Number](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#number) | 0 |
| h | am/pm 中的小时数（1-12） | [Number](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#number) | 12 |
| m | 小时中的分钟数 | [Number](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#number) | 30 |
| s | 分钟中的秒数 | [Number](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#number) | 55 |
| S | 毫秒数 | [Number](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#number) | 978 |
| z | 时区 | [General time zone](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#timezone) | Pacific Standard Time; PST; GMT-08:00 |
| Z | 时区 | [RFC 822 time zone](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#rfc822timezone) | -0800 |

模式字母通常是重复的，其数量确定其精确表示：

* **Text:** 对于格式化来说，如果模式字母的数量大于等于 4，则使用完全形式；否则，在可用的情况下使用短形式或缩写形式。对于解析来说，两种形式都是可接受的，与模式字母的数量无关。
* **Number:** 对于格式化来说，模式字母的数量是最小的数位，如果数位不够，则用 0 填充以达到此数量。对于解析来说，模式字母的数量被忽略，除非必须分开两个相邻字段。
* **Year:** 如果格式器的 [Calendar](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/DateFormat.html#getCalendar()) 是格里高利历，则应用以下规则。
  + 对于格式化来说，如果模式字母的数量为 2，则年份截取为 2 位数,否则将年份解释为 [number](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#number)。
  + 对于解析来说，如果模式字母的数量大于 2，则年份照字面意义进行解释，而不管数位是多少。因此使用模式 "MM/dd/yyyy"，将 "01/11/12" 解析为公元 12 年 1 月 11 日。
  + 在解析缩写年份模式（"y" 或 "yy"）时，SimpleDateFormat 必须相对于某个世纪来解释缩写的年份。这通过将日期调整为 SimpleDateFormat 实例创建之前的 80 年和之后 20 年范围内来完成。例如，在 "MM/dd/yy" 模式下，如果 SimpleDateFormat 实例是在 1997 年 1 月 1 日创建的，则字符串 "01/11/12" 将被解释为 2012 年 1 月 11 日，而字符串 "05/04/64" 将被解释为 1964 年 5 月 4 日。在解析时，只有恰好由两位数字组成的字符串（如 [Character.isDigit(char)](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/lang/Character.html#isDigit(char)) 所定义的）被解析为默认的世纪。其他任何数字字符串将照字面意义进行解释，例如单数字字符串，3 个或更多数字组成的字符串，或者不都是数字的两位数字字符串（例如"-1"）。因此，在相同的模式下， "01/02/3" 或 "01/02/003" 解释为公元 3 年 1 月 2 日。同样，"01/02/-3" 解析为公元前 4 年 1 月 2 日。

否则，则应用日历系统特定的形式。对于格式化和解析，如果模式字母的数量为 4 或大于 4，则使用日历特定的 [long form](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/Calendar.html#LONG)。否则，则使用日历特定的 [short or abbreviated form](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/Calendar.html#SHORT)。

* **Month:** 如果模式字母的数量为 3 或大于 3，则将月份解释为 [text](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#text)；否则解释为 [number](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#number)。
* **General time zone:** 如果时区有名称，则将它们解释为 [text](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#text)。对于表示 GMT 偏移值的时区，使用以下语法：
* *GMTOffsetTimeZone:*
* GMT *Sign* *Hours* : *Minutes*
* *Sign:* one of
* + -
* *Hours:*
* *Digit*
* *Digit* *Digit*
* *Minutes:*
* *Digit* *Digit*
* *Digit:* one of

0 1 2 3 4 5 6 7 8 9

*Hours* 必须在 0 到 23 之间，*Minutes* 必须在 00 到 59 之间。格式是与语言环境无关的，并且数字必须取自 Unicode 标准的 Basic Latin 块。

对于解析来说，[RFC 822 time zones](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#rfc822timezone) 也是可接受的。

* **RFC 822 time zone:** 对于格式化来说，使用 RFC 822 4-digit 时区格式：
* *RFC822TimeZone:*
* *Sign* *TwoDigitHours* *Minutes*
* *TwoDigitHours:*

*Digit Digit*

*TwoDigitHours* 必须在 00 和 23 之间。其他定义请参阅 [general time zones](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#timezone)。

对于解析来说，[general time zones](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/text/SimpleDateFormat.html#timezone) 也是可接受的。

SimpleDateFormat 还支持本地化日期和时间模式 字符串。在这些字符串中，以上所述的模式字母可以用其他与语言环境有关的模式字母来替换。SimpleDateFormat 不处理除模式字母之外的文本本地化；而由类的客户端来处理。

**示例**

以下示例显示了如何在美国语言环境中解释日期和时间模式。给定的日期和时间为美国太平洋时区的本地时间 2001-07-04 12:08:56。

|  |  |
| --- | --- |
| **日期和时间模式** | **结果** |
| "yyyy.MM.dd G 'at' HH:mm:ss z" | 2001.07.04 AD at 12:08:56 PDT |
| "EEE, MMM d, ''yy" | Wed, Jul 4, '01 |
| "h:mm a" | 12:08 PM |
| "hh 'o''clock' a, zzzz" | 12 o'clock PM, Pacific Daylight Time |
| "K:mm a, z" | 0:08 PM, PDT |
| "yyyyy.MMMMM.dd GGG hh:mm aaa" | 02001.July.04 AD 12:08 PM |
| "EEE, d MMM yyyy HH:mm:ss Z" | Wed, 4 Jul 2001 12:08:56 -0700 |
| "yyMMddHHmmssZ" | 010704120856-0700 |
| "yyyy-MM-dd'T'HH:mm:ss.SSSZ" | 2001-07-04T12:08:56.235-0700 |

## 正则表达式

### 正则表达式的构造摘要

|  |  |
| --- | --- |
| **构造** | **匹配** |
|  |  |
| **字符** | |
| *x* | 字符 *x* |
| \\ | 反斜线字符 |
| \0*n* | 带有八进制值 0 的字符 *n* (0 <= *n* <= 7) |
| \0*nn* | 带有八进制值 0 的字符 *nn* (0 <= *n* <= 7) |
| \0*mnn* | 带有八进制值 0 的字符 *mnn*（0 <= *m* <= 3、0 <= *n* <= 7） |
| \x*hh* | 带有十六进制值 0x 的字符 *hh* |
| \u*hhhh* | 带有十六进制值 0x 的字符 *hhhh* |
| \t | 制表符 ('\u0009') |
| \n | 新行（换行）符 ('\u000A') |
| \r | 回车符 ('\u000D') |
| \f | 换页符 ('\u000C') |
| \a | 报警 (bell) 符 ('\u0007') |
| \e | 转义符 ('\u001B') |
| \c*x* | 对应于 *x* 的控制符 |
|  |  |
| **字符类** | |
| [abc] | a、b 或 c（简单类） |
| [^abc] | 任何字符，除了 a、b 或 c（否定） |
| [a-zA-Z] | a 到 z 或 A 到 Z，两头的字母包括在内（范围） |
| [a-d[m-p]] | a 到 d 或 m 到 p：[a-dm-p]（并集） |
| [a-z&&[def]] | d、e 或 f（交集） |
| [a-z&&[^bc]] | a 到 z，除了 b 和 c：[ad-z]（减去） |
| [a-z&&[^m-p]] | a 到 z，而非 m 到 p：[a-lq-z]（减去） |
|  |  |
| **预定义字符类** | |
| . | 任何字符（与[行结束符](mk:@MSITStore:D:\\帮助文档\\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html" \l "lt)可能匹配也可能不匹配） |
| \d | 数字：[0-9] |
| \D | 非数字： [^0-9] |
| \s | 空白字符：[ \t\n\x0B\f\r] |
| \S | 非空白字符：[^\s] |
| \w | 单词字符：[a-zA-Z\_0-9] |
| \W | 非单词字符：[^\w] |
|  |  |
| **POSIX 字符类（仅 US-ASCII）** | |
| \p{Lower} | 小写字母字符：[a-z] |
| \p{Upper} | 大写字母字符：[A-Z] |
| \p{ASCII} | 所有 ASCII：[\x00-\x7F] |
| \p{Alpha} | 字母字符：[\p{Lower}\p{Upper}] |
| \p{Digit} | 十进制数字：[0-9] |
| \p{Alnum} | 字母数字字符：[\p{Alpha}\p{Digit}] |
| \p{Punct} | 标点符号：!"#$%&'()\*+,-./:;<=>?@[\]^\_`{|}~ |
| \p{Graph} | 可见字符：[\p{Alnum}\p{Punct}] |
| \p{Print} | 可打印字符：[\p{Graph}\x20] |
| \p{Blank} | 空格或制表符：[ \t] |
| \p{Cntrl} | 控制字符：[\x00-\x1F\x7F] |
| \p{XDigit} | 十六进制数字：[0-9a-fA-F] |
| \p{Space} | 空白字符：[ \t\n\x0B\f\r] |
|  |  |
| **java.lang.Character 类（简单的** [**java 字符类型**](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#jcc)**）** | |
| \p{javaLowerCase} | 等效于 java.lang.Character.isLowerCase() |
| \p{javaUpperCase} | 等效于 java.lang.Character.isUpperCase() |
| \p{javaWhitespace} | 等效于 java.lang.Character.isWhitespace() |
| \p{javaMirrored} | 等效于 java.lang.Character.isMirrored() |
|  |  |
| **Unicode 块和类别的类** | |
| \p{InGreek} | Greek 块（简单[块](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#ubc)）中的字符 |
| \p{Lu} | 大写字母（简单[类别](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#ubc)） |
| \p{Sc} | 货币符号 |
| \P{InGreek} | 所有字符，Greek 块中的除外（否定） |
| [\p{L}&&[^\p{Lu}]] | 所有字母，大写字母除外（减去） |
|  |  |
| **边界匹配器** | |
| ^ | 行的开头 |
| $ | 行的结尾 |
| \b | 单词边界 |
| \B | 非单词边界 |
| \A | 输入的开头 |
| \G | 上一个匹配的结尾 |
| \Z | 输入的结尾，仅用于最后的[结束符](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#lt)（如果有的话） |
| \z | 输入的结尾 |
|  |  |
| **Greedy 数量词** | |
| *X*? | *X*，一次或一次也没有 |
| *X*\* | *X*，零次或多次 |
| *X*+ | *X*，一次或多次 |
| *X*{*n*} | *X*，恰好 *n* 次 |
| *X*{*n*,} | *X*，至少 *n* 次 |
| *X*{*n*,*m*} | *X*，至少 *n* 次，但是不超过 *m* 次 |
|  |  |
| **Reluctant 数量词** | |
| *X*?? | *X*，一次或一次也没有 |
| *X*\*? | *X*，零次或多次 |
| *X*+? | *X*，一次或多次 |
| *X*{*n*}? | *X*，恰好 *n* 次 |
| *X*{*n*,}? | *X*，至少 *n* 次 |
| *X*{*n*,*m*}? | *X*，至少 *n* 次，但是不超过 *m* 次 |
|  |  |
| **Possessive 数量词** | |
| *X*?+ | *X*，一次或一次也没有 |
| *X*\*+ | *X*，零次或多次 |
| *X*++ | *X*，一次或多次 |
| *X*{*n*}+ | *X*，恰好 *n* 次 |
| *X*{*n*,}+ | *X*，至少 *n* 次 |
| *X*{*n*,*m*}+ | *X*，至少 *n* 次，但是不超过 *m* 次 |
|  |  |
| **Logical 运算符** | |
| *XY* | *X* 后跟 *Y* |
| *X*|*Y* | *X* 或 *Y* |
| (*X*) | X，作为[捕获组](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#cg) |
|  |  |
| **Back 引用** | |
| \*n* | 任何匹配的 *n*th [捕获组](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#cg) |
|  |  |
| **引用** | |
| \ | Nothing，但是引用以下字符 |
| \Q | Nothing，但是引用所有字符，直到 \E |
| \E | Nothing，但是结束从 \Q 开始的引用 |
|  |  |
| **特殊构造（非捕获）** | |
| (?:*X*) | *X*，作为非捕获组 |
| (?idmsux-idmsux) | Nothing，但是将匹配标志[i](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#CASE_INSENSITIVE) [d](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#UNIX_LINES) [m](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#MULTILINE) [s](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#DOTALL) [u](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#UNICODE_CASE) [x](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#COMMENTS) on - off |
| (?idmsux-idmsux:*X*) | *X*，作为带有给定标志 [i](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#CASE_INSENSITIVE) [d](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#UNIX_LINES) [m](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#MULTILINE) [s](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#DOTALL) [u](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#UNICODE_CASE) [x](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#COMMENTS) on - off的[非捕获组](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#cg) |
| (?=*X*) | *X*，通过零宽度的正 lookahead |
| (?!*X*) | *X*，通过零宽度的负 lookahead |
| (?<=*X*) | *X*，通过零宽度的正 lookbehind |
| (?<!*X*) | *X*，通过零宽度的负 lookbehind |
| (?>*X*) | *X*，作为独立的非捕获组 |

### 反斜线、转义和引用

反斜线字符 ('\') 用于引用转义构造，如上表所定义的，同时还用于引用其他将被解释为非转义构造的字符。因此，表达式 \\ 与单个反斜线匹配，而 \{ 与左括号匹配。

在不表示转义构造的任何字母字符前使用反斜线都是错误的；它们是为将来扩展正则表达式语言保留的。可以在非字母字符前使用反斜线，不管该字符是否非转义构造的一部分。

根据 [Java Language Specification](http://java.sun.com/docs/books/jls) 的要求，Java 源代码的字符串中的反斜线被解释为 [Unicode 转义](http://java.sun.com/docs/books/jls/third_edition/html/lexical.doc.html#100850)或其他[字符转义](http://java.sun.com/docs/books/jls/third_edition/html/lexical.doc.html#101089)。因此必须在字符串字面值中使用两个反斜线，表示正则表达式受到保护，不被 Java 字节码编译器解释。例如，当解释为正则表达式时，字符串字面值 "\b" 与单个退格字符匹配，而 "\\b" 与单词边界匹配。字符串字面值 "\(hello\)" 是非法的，将导致编译时错误；要与字符串 (hello) 匹配，必须使用字符串字面值 "\\(hello\\)"。

### 字符类

字符类可以出现在其他字符类中，并且可以包含并集运算符（隐式）和交集运算符 (&&)。并集运算符表示至少包含其某个操作数类中所有字符的类。交集运算符表示包含同时位于其两个操作数类中所有字符的类。

字符类运算符的优先级如下所示，按从最高到最低的顺序排列：

|  |  |  |
| --- | --- | --- |
| **1** | 字面值转义 | \x |
| **2** | 分组 | [...] |
| **3** | 范围 | a-z |
| **4** | 并集 | [a-e][i-u] |
| **5** | 交集 | [a-z&&[aeiou]] |

注意，元字符的不同集合实际上位于字符类的内部，而非字符类的外部。例如，正则表达式 . 在字符类内部就失去了其特殊意义，而表达式 - 变成了形成元字符的范围。

### 行结束符

*行结束符* 是一个或两个字符的序列，标记输入字符序列的行结尾。以下代码被识别为行结束符：

* 新行（换行）符 ('\n')、
* 后面紧跟新行符的回车符 ("\r\n")、
* 单独的回车符 ('\r')、
* 下一行字符 ('\u0085')、
* 行分隔符 ('\u2028') 或
* 段落分隔符 ('\u2029)。

如果激活 [UNIX\_LINES](mk:@MSITStore:D:\\帮助文档\\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html" \l "UNIX_LINES) 模式，则新行符是唯一识别的行结束符。

如果未指定 [DOTALL](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#DOTALL) 标志，则正则表达式 . 可以与任何字符（行结束符除外）匹配。

默认情况下，正则表达式 ^ 和 $ 忽略行结束符，仅分别与整个输入序列的开头和结尾匹配。如果激活 [MULTILINE](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#MULTILINE) 模式，则 ^ 在输入的开头和行结束符之后（输入的结尾）才发生匹配。处于 [MULTILINE](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html#MULTILINE) 模式中时，$ 仅在行结束符之前或输入序列的结尾处匹配。

### 组和捕获

捕获组可以通过从左到右计算其开括号来编号。例如，在表达式 ((A)(B(C))) 中，存在四个这样的组：

|  |  |
| --- | --- |
| **1** | ((A)(B(C))) |
| **2** | \A |
| **3** | (B(C)) |
| **4** | (C) |

组零始终代表整个表达式。

之所以这样命名捕获组是因为在匹配中，保存了与这些组匹配的输入序列的每个子序列。捕获的子序列稍后可以通过 Back 引用在表达式中使用，也可以在匹配操作完成后从匹配器获取。

与组关联的捕获输入始终是与组最近匹配的子序列。如果由于量化的缘故再次计算了组，则在第二次计算失败时将保留其以前捕获的值（如果有的话）例如，将字符串 "aba" 与表达式 (a(b)?)+ 相匹配，会将第二组设置为 "b"。在每个匹配的开头，所有捕获的输入都会被丢弃。

以 (?) 开头的组是纯的*非捕获* 组，它不捕获文本，也不针对组合计进行计数。

### Unicode 支持

此类符合 *[Unicode Technical Standard #18:Unicode Regular Expression Guidelines](http://www.unicode.org/reports/tr18/)* 第 1 级和 RL2.1 Canonical Equivalents。

Java 源代码中的 Unicode 转义序列（如 \u2014）是按照 Java Language Specification 的 [第 3.3 节](http://java.sun.com/docs/books/jls/third_edition/html/lexical.html#100850)中的描述处理的。这样的转义序列还可以由正则表达式解析器直接实现，以便在从文件或键盘击键读取的表达式中使用 Unicode 转义。因此，可以将不相等的字符串 "\u2014" 和 "\\u2014" 编译为相同的模式，从而与带有十六进制值 0x2014 的字符匹配。

与 Perl 中一样，Unicode 块和类别是使用 \p 和 \P 构造编写的。如果输入具有属性 *prop*，则与 \p{*prop*} 匹配，而输入具有该属性时与 \P{*prop*} 不匹配。块使用前缀 In 指定，与在 InMongolian 中一样。可以使用可选前缀 Is 指定类别：\p{L} 和 \p{IsL} 都表示 Unicode 字母的类别。块和类别在字符类的内部和外部都可以使用。

受支持的类别是由 [Character](mk:@MSITStore:D:\\帮助文档\\JDK_API_1_6_zh_CN.CHM::/java/lang/Character.html" \o "java.lang 中的类) 类指定版本中的 [*The Unicode Standard*](http://www.unicode.org/unicode/standard/standard.html) 的类别。类别名称是在 Standard 中定义的，即标准又丰富。Pattern 所支持的块名称是 [UnicodeBlock.forName](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/lang/Character.UnicodeBlock.html#forName(java.lang.String)) 所接受和定义的有效块名称。

行为类似 java.lang.Character boolean 是 *methodname* 方法（废弃的类别除外）的类别，可以通过相同的 \p{*prop*} 语法来提供，其中指定的属性具有名称 java*methodname*。

### 与 Perl 5 相比较

Pattern 引擎用有序替换项执行传统上基于 NFA 的匹配，与 Perl 5 中进行的相同。

此类不支持 Perl 构造：

* 条件构造 (?{*X*}) 和 (?(*condition*)*X*|*Y*)、
* 嵌入式代码构造 (?{*code*}) 和 (??{*code*})、
* 嵌入式注释语法 (?#comment) 和
* 预处理操作 \l \u、\L 和 \U。

此类支持但 Perl 不支持的构造：

* Possessive 数量词，它可以尽可能多地进行匹配，即使这样做导致所有匹配都成功时也如此。
* 字符类并集和交集，如[上文](mk:@MSITStore:D:\\帮助文档\\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Pattern.html" \l "cc)所述。

与 Perl 的显著不同点是：

* 在 Perl 中，\1 到 \9 始终被解释为 Back 引用；如果至少存在多个子表达式，则大于 9 的反斜线转义数按 Back 引用对待，否则在可能的情况下，它将被解释为八进制转义。在此类中，八进制转义必须始终以零开头。在此类中，\1 到 \9 始终被解释为 Back 引用，较大的数被接受为 Back 引用，如果在正则表达式中至少存在多个子表达式的话；否则，解析器将删除数字，直到该数小于等于组的现有数或者其为一个数字。
* Perl 使用 g 标志请求恢复最后匹配丢失的匹配。此功能是由 [Matcher](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Matcher.html) 类显式提供的：重复执行 [find](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/Matcher.html#find()) 方法调用可以恢复丢失的最后匹配，除非匹配器被重置。
* 在 Perl 中，位于表达式顶级的嵌入式标记对整个表达式都有影响。在此类中，嵌入式标志始终在它们出现的时候才起作用，不管它们位于顶级还是组中；在后一种情况下，与在 Perl 中类似，标志在组的结尾处还原。
* Perl 允许错误匹配构造，如在表达式 \*a 中，以及不匹配的括号，如在在表达式 abc] 中，并将其作为字面值对待。此类还接受不匹配的括号，但对 +、? 和 \* 不匹配元字符有严格限制；如果遇到它们，则抛出 [PatternSyntaxException](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/util/regex/PatternSyntaxException.html)。

有关正则表达式构造行为更准确的描述，请参见 [*Mastering Regular Expressions, 2nd Edition*](http://www.oreilly.com/catalog/regex2/)，该书由 Jeffrey E. F. Friedl、O'Reilly 和 Associates 合著，于 2002 年出版。

## 数字格式字符串

### 模式

DecimalFormat 模式具有下列语法：

*模式：*

*正数模式*

*正数模式*；*负数模式*

*正数模式：*

*前缀opt* *数字后缀opt*

*负数模式：*

*前缀opt* *数字后缀opt*

*前缀：*

除 \uFFFE、\uFFFF 和特殊字符以外的所有 Unicode 字符

*后缀：*

除 \uFFFE、\uFFFF 和特殊字符以外的所有 Unicode 字符

*数字：*

*整数指数opt*

*整数*。*小数指数opt*

*整数：*

*最小整数*

#

# *整数*

# , *整数*

*最小整数：*

0

0 *最小整数*

0 , *最小整数*

*小数：*

*最小小数opt* *可选小数opt*

*最小小数：*

0 *最小小数opt*

*可选小数：*

# *可选小数opt*

*指数：*

E *最小指数*

*最小指数：*

0 *最小指数opt*

DecimalFormat 模式包含正数和负数子模式，例如 "#,##0.00;(#,##0.00)"。每个子模式都有前缀、数字部分和后缀。负数子模式是可选的；如果存在，则将用已本地化的减号（在多数语言环境中是 '-'）作为前缀的正数子模式用作负数子模式。也就是说，单独的 "0.00" 等效于 "0.00;-0.00"。如果存在显式的负数子模式，则它仅指定负数前缀和后缀；数字位数、最小位数，其他特征都与正数模式相同。这意味着 "#,##0.0#;(#)" 的行为与 "#,##0.0#;(#,##0.0#)" 完全相同。

用于无穷大值、数字、千位分隔符、小数分隔符等的前缀、后缀和各种符号可设置为任意值，并且能在格式化期间正确显示。但是，必须注意不要让符号和字符串发生冲突，否则解析是不可靠的。例如，为了让 DecimalFormat.parse() 能够区分正数和负数，正数和负数前缀或后缀必须是不同的。（如果它们相同，则 DecimalFormat 的行为就如同未指定负数子模式一样。）另一个示例是小数分隔符和千位分隔符应该是不同的字符，否则将不可能进行解析。

分组分隔符通常用于千位，但是在某些国家/地区中它用于分隔万位。分组大小是分组字符之间的固定数字位数，例如 100,000,000 是 3，而 1,0000,0000 则是 4。如果使用具有多个分组字符的模式，则最后一个分隔符和整数结尾之间的间隔才是使用的分组大小。所以 "#,##,###,####" == "######,####" == "##,####,####"。

### 特殊模式字符

模式中的很多字符都是按字面解释的；在解析期间对其进行匹配，在格式化期间则不经改变地输出。另一方面，特殊字符代表了其他字符、字符串或字符类。如果要将其作为字面量出现在前缀或后缀中，那么除非另行说明，否则必须对其加引号。

下列字符用在非本地化的模式中。已本地化的模式使用从此格式器的 DecimalFormatSymbols 对象中获得的相应字符，这些字符已失去其特殊状态。两种例外是货币符号和引号，不将其本地化。

|  |  |  |  |
| --- | --- | --- | --- |
| **符号** | **位置** | **本地化？** | **含义** |
| 0 | 数字 | 是 | 阿拉伯数字 |
| # | 数字字 | 是 | 阿拉伯数字，如果不存在则显示为 0 |
| . | 数字 | 是 | 小数分隔符或货币小数分隔符 |
| - | 数字 | 是 | 减号 |
| , | 数字 | 是 | 分组分隔符 |
| E | 数字 | 是 | 分隔科学计数法中的尾数和指数。在前缀或后缀中无需加引号。 |
| ; | 子模式边界 | 是 | 分隔正数和负数子模式 |
| % | 前缀或后缀 | 是 | 乘以 100 并显示为百分数 |
| \u2030 | 前缀或后缀 | 是 | 乘以 1000 并显示为千分数 |
| ¤ (\u00A4) | 前缀或后缀 | 否 | 货币记号，由货币符号替换。如果两个同时出现，则用国际货币符号替换。如果出现在某个模式中，则使用货币小数分隔符，而不使用小数分隔符。 |
| ' | 前缀或后缀 | 否 | 用于在前缀或或后缀中为特殊字符加引号，例如 "'#'#" 将 123 格式化为 "#123"。要创建单引号本身，请连续使用两个单引号："# o''clock"。 |

### 科学计数法

科学计数法中的数表示为一个尾数和一个 10 的几次幂的乘积，例如可将 1234 表示为 1.234 x 10^3。尾数的范围通常是 1.0 <= x < 10.0，但并非必需如此。可指示 DecimalFormat 仅通过某个模式 来格式化和解析科学计数法表示的数；目前没有创建科学计数法格式的工厂方法。在这个模式中，指数字符后面紧跟着一个或多个数字字符即指示科学计数法。示例："0.###E0" 将数字 1234 格式化为 "1.234E3"。

* 指数字符后面的数字位数字符数给出了最小的指数位数。没有最大值。使用本地化的减号来格式化负数指数，不 使用模式中的前缀和后缀。这就允许存在诸如 "0.###E0 m/s" 等此类的模式。
* 最小和最大整数数字位数一起进行解释：
  + 如果最大整数数字位数大于其最小整数数字位数并且大于 1，则强制要求指数为最大整数数字位数的倍数，并将最小整数数字位数解释为 1。最常见的用法是生成工程计数法，其中指数是 3 的倍数，如 "##0.#####E0"。使用此模式时，数 12345 格式化为 "12.345E3"，123456 则格式化为 "123.456E3"。
  + 否则通过调整指数来得到最小整数数字位数。示例：使用 "00.###E0" 格式化 0.00123 时得到 "12.3E-4"。
* 尾数中的有效位数是最小整数 和最大小数 位数的和，不受最大整数位数的影响。例如，使用 "##0.##E0" 格式化 12345 得到 "12.3E3"。要显示所有位数，请将有效位数计数设置为零。有效位数不会影响解析。
* 指数模式可能不包含分组分隔符。

### 舍入

DecimalFormat 提供 [RoundingMode](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/math/RoundingMode.html) 中定义的舍入模式进行格式化。默认情况下，它使用 [RoundingMode.HALF\_EVEN](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/math/RoundingMode.html#HALF_EVEN)。

### 阿拉伯数字

为了进行格式化，DecimalFormat 使用 DecimalFormatSymbols 对象中所定义的、从已本地化的阿拉伯数字 0 开始的 10 个连续字符作为阿拉伯数字。为了进行解析，可识别 [Character.digit](mk:@MSITStore:D:\帮助文档\JDK_API_1_6_zh_CN.CHM::/java/lang/Character.html#digit(char, int)) 所定义的这些阿拉伯数字和所有 Unicode 十进制阿拉伯数字。

### 特殊值

NaN 被格式化为一个字符串，通常具有单个字符 \uFFFD。此字符串由 DecimalFormatSymbols 对象所确定。这是唯一不使用前缀和后缀的值。

无穷大的值被格式化为一个字符串，通常具有单个字符 \u221E，具有正数或负数前缀和后缀。无穷大值的字符串由 DecimalFormatSymbols 对象所确定。

将负零（"-0"）解析为

* 如果 isParseBigDecimal() 为 true，则为 BigDecimal(0)，
* 如果 isParseBigDecimal() 为 false 并且 isParseIntegerOnly() 为 true，则为 Long(0)，
* 如果 isParseBigDecimal() 和 isParseIntegerOnly() 均为 false，则为 Double(-0.0)。