

MixIOT 系统功能详细说明

Version 1.0 Release 1

目录

第一章	概述	4
1.1	文档说明	4
1.2	适用对象	4
1.3	术语和缩略词	4
第二章	功能说明	5
2.1	对象管理-Object	5
2.1.1	简介	5
2.1.2	主要功能	5
2.1.3	映射表管理	5
2.1.4	控制码管理	10
2.1.5	对象管理	14
2.1.6	显示板管理	21
2.2	终端管理-Terminal	25
2.2.1	应用场景	25
2.2.2	菜单功能	25
2.2.3	使用流程	26
2.2.4	其他说明	29
2.3	客户管理-Customer	错误! 未定义书签。
2.4	信使服务-MixMessenger	错误! 未定义书签。
2.5	代理服务-Agent	29
2.5.1	简介	29
2.5.2	事件订阅	30
2.5.3	事件发布	33
2.5.4	取消订阅	35
2.5.5	获取订阅列表	36
2.5.6	获取订阅详情	38
2.5.7	Agent 事件	40
2.5.8	用户事件	41
2.5.9	Agent 应用	41
2.5.10	附录	43
2.6	文件管理-File	错误! 未定义书签。
2.7	离线数据-Collect	44
2.7.1	简介	44
2.7.2	Collect 项目脚本	44
2.7.3	通过 API 进行数据采集	47
2.7.4	推送	48
2.7.5	Collect 接口 action 说明	48
2.7.6	Collect event 说明	65
2.7.7	Collect 应用界面使用	65
2.8	CLI 控制台-CLI	68
2.8.1	Bock 列表	68
2.8.2	命令列表	69

2.8.3	发送命令	69
2.8.4	返回结果	70
2.8.5	命令界面	71
2.9	门户中心	72
2.9.1	登录方法	72
2.9.2	添加分组	73
2.10	门户设置	74
2.10.1	主页设置	74
2.10.2	登录页设置	75
2.11	应用中心	75
2.11.1	应用管理	75
2.11.2	Block 管理	76
2.12	用户中心	78
2.12.1	用户管理	78
2.12.2	用户组管理	81
2.13	统计报表	错误! 未定义书签。
2.14	统计计算-Statistics	82
2.14.1	脚本含义说明	82
2.14.2	支持 ACTION 列表	86
2.14.3	EVENT 列表	117
2.15	事件持续-Retainer	118
2.15.1	简介	118
2.15.2	历程应用	118
2.15.3	脚本	122
2.15.4	命令列表 (action)	123
2.15.5	示例	125
2.16	维保任务	127
2.16.1	应用场景	127
2.16.2	菜单功能	128
2.16.3	使用流程	128
2.17	数据查询和导出	错误! 未定义书签。
2.18	显示板数据服务-Dashbos	130
2.18.1	Dashbos 项目脚本	130
2.18.2	Dashbos 接口 action 说明	132
2.19	显示板工具	错误! 未定义书签。
2.20	APP	错误! 未定义书签。
2.21	第三方应用数据接口	错误! 未定义书签。

第一章 概述

1.1 文档说明

本文档用于介绍 MixIOT 系统相关功能的界面和基础使用步骤，以及控制台命令程序，用于指导用户熟悉及使用 MixIOT 物联网系统。

1.2 适用对象

本文档的目的读者是智物联物联网体系下所有开发人员及 MixIOT 物联网系统的使用人员。

1.3 术语和缩略词

序号	术语名称	其它名称	术语说明
1			
2			
3			
4			
5			
6			
7			

第二章 功能说明

2.1 对象管理-Object

2.1.1 简介

Object block 是低频操作的系统，配置一次可以持续使用。本文档主要对 Object（对象管理）的使用进行详细说明，讲解如何配置对象、映射表、控制码及显示板。

2.1.2 主要功能

Object block 菜单结构包括对象管理、映射表管理、控制码管理和显示板管理 4 个菜单：



使用过程中按如下流程配置：映射表管理、控制码管理→对象管理→显示板管理，后续内容将按照此配置流程展开介绍。

2.1.3 映射表管理

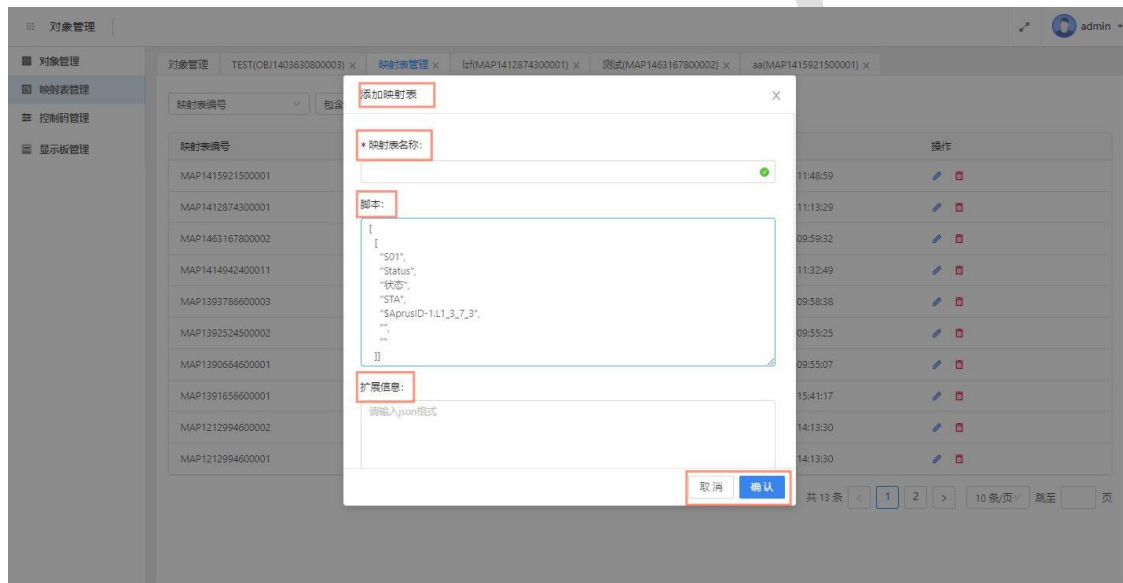
映射表是 MixIOT 的一个重要利器，一个对象的全部数据，无论是直接采集的数据，还是需要计算的数据、统计后的数据、离线的数据、甚至是外部的数据，都需要通过映射表，整

合为对象的 FV 变量，才能进行后续的使用。

2.1.3.1 配置说明

映射表添加

点击列表页的添加按钮，进入映射表添加界面，录入相应字段信息，点击确认，即完成映射表的添加。



字段说明

映射表名称：必填项，用户自定义。

描述：非必填项，用户自定义。

扩展信息：非必填项，此处用于填写扩展字段脚本，可以将关联对象中的设备信息字段进行扩展。

脚本：必填项，此处用于填写映射表脚本，是设备关联适配器上报的 key 和对象的 FV 变量之间的映射脚本。

脚本规范

映射表扩展信息：

```
{
```

```
  "title": "锅炉信息",
```

```

    "data": [
        ["燃料类型", "天然气"],
        ["锅炉重量", "2 吨"],
        ["出厂日期", "2018-03-03"]
    ]
}, {
    "title": "能效指标信息",
    "data": [
        ["热效率", 0.8],
        ["燃料热值", 7000],
        ["*吨汽燃料指标*", 0.21]
    ]
}
    
```

Title: 为段落标题，名称可自定义，即代码中“锅炉信息”、“能效指标信息”可更改。

Data: 为段落中字段内容，字段名和字段值可自定义。

映射表脚本:

```

[
    ["v1", "", "STA 变量-相对适配器", "STA", "$AprusID-1.L1_3_7_2", "", ""],
    ["v2", "", "STA 变量-相对适配器", "STA", "$AprusID-1.L1_3_7_2", "v1+10", ""],
    ["v3", "", "STA 变量-固定适配器", "STA", "AX20210229000249.L1_3_7_2", "v2+10", ""],
    ["v4", "", "离线推送", "STA", "$CollectID-1.Tem", "Tem+20", ""],
    ["v5", "", "离线推送", "STA", "COL12236500002.Tem", "Tem+5", ""],
    ["v6", "", "函数计算", "STA", "$AprusID-1.L1_3_7_2", "min(v1, v2)", ""],
    ["v7", "", "比较运算", "STA", "$AprusID-1.L1_3_7_2", "v1>v2?v1:v2", ""],
    ["1001", "start event", "开始事件", "EVNT", "$AprusID-1.L1_3_7_2", "", "v1==0"],
    ["1002", "stop event", "停止事件", "EVNT", "$AprusID-1.L1_3_7_2", "", "v1==1"]
]
    
```

脚本： [FV/Code, LabelEn, LabelLocal, Category, Datasource, LogicExpr1, LogicExpr2]。

第一列：FV 为 mosaic 拼接时的变量名，如 FV 为"S01"，在脚本中为字符串，Code 为事件的 code 码，当逻辑表达式 2 满足条件时，会根据 code 码来发布事件到 agent，同样 code 码也为字符串，如"1001"为开始某事件，"1002"为结束某事件。

第二列：LabelEn 为英文标签，主要对 FV/Code 用英文进行描述。

第三列：LabelLocal 为中文标签，主要对 FV/Code 用本地语言进行描述。

第四列：Category 为映射类型，主要分 STA, SET, EVNT 三类映射类型，其中 STA/SET 为正常处理数据，进行 mosaic 拼接，EVNT 为事件类，在满足逻辑表达式 2 的情况下，生成事件。

第五列：Datasource 为数据来源，一般的格式为 AprusID.AprusKey，如\$AprusID-1.L1_2_3_4 需要在脚本解析时，进行宏变量替换，当然 AprusID 也可以写成固定的，还有其他格式为 CollectID.CollectKey，如\$CollectID-1.Tem 需要在脚本解析时，进行宏变量替换，当然 CollectID 也可以写成固定的。

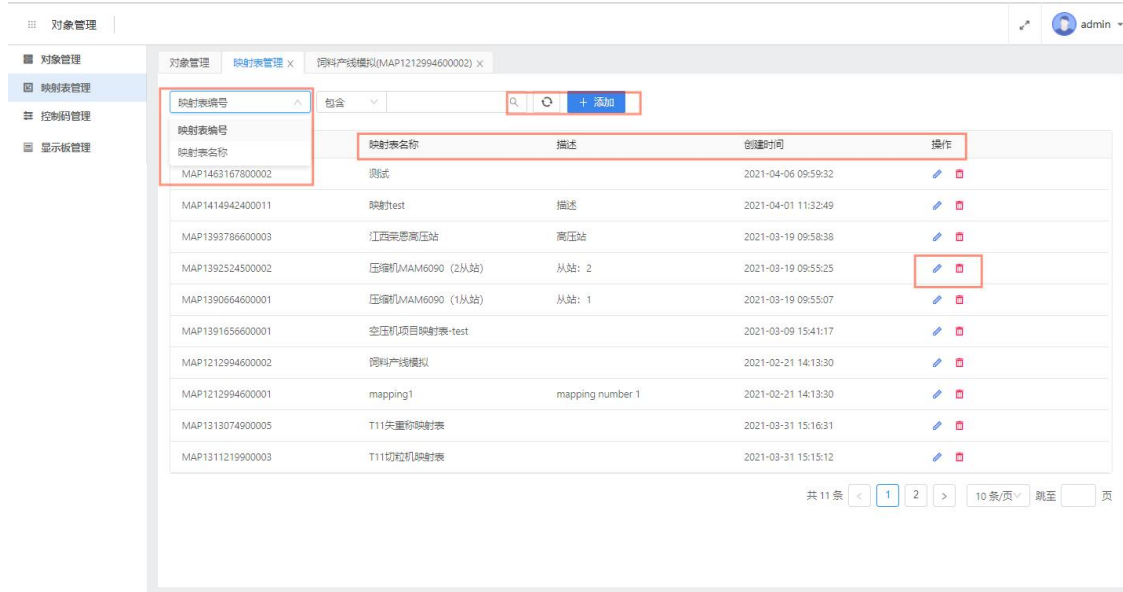
第六列：LogicExpr1 为逻辑表达式 1，主要用于值计算。

第七列：LogicExpr2 为逻辑表达式 2，主要用于是否生成事件。

2.1.3.2 界面说明

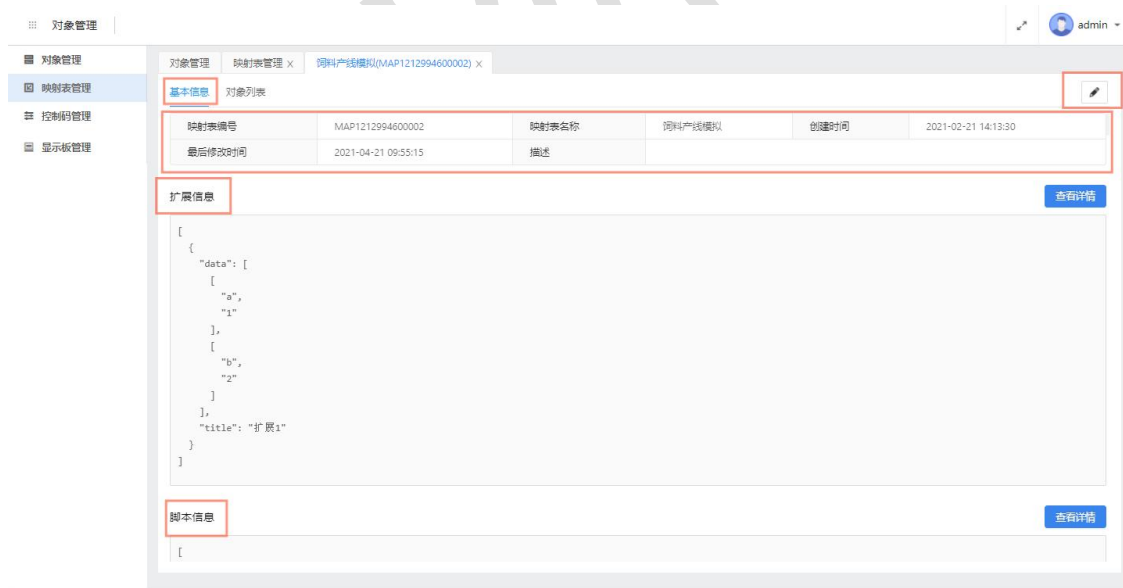
映射表列表页

展示配置的所有映射表列表，在本页面可以简单了解所有映射表的编号、名称及创建时间等基本信息，支持按映射表编号和映射表名称进行搜索及刷新，同时提供了映射表的添加、编辑和删除功能。



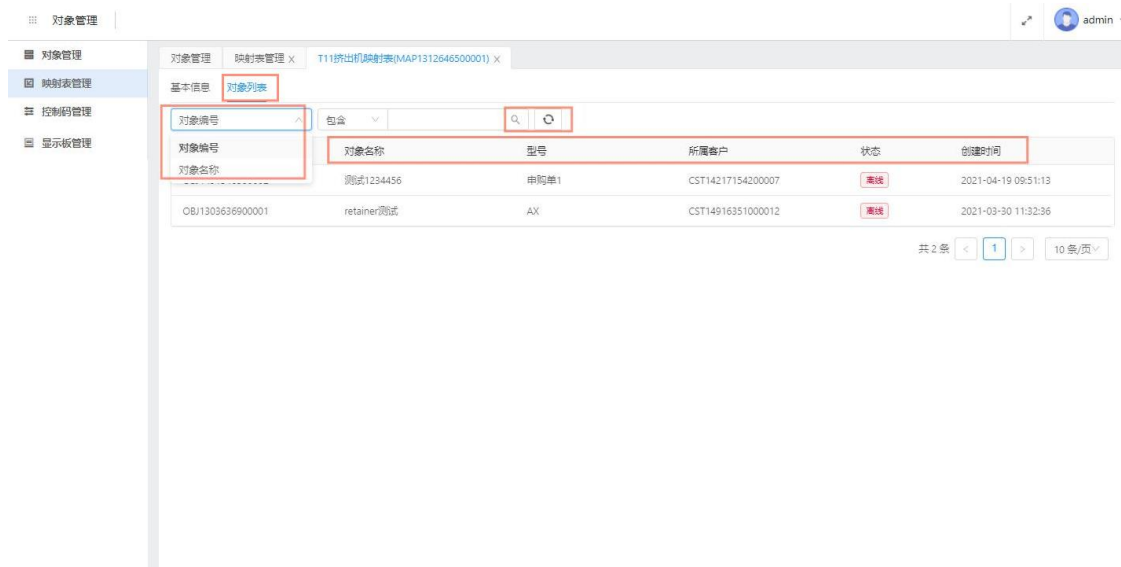
映射表详情页-基本信息

展示所选映射表的详细信息，包括映射表编号、名称、创建时间、最后修改时间、扩展信息脚本、映射表脚本，同时提供映射表的编辑功能。



映射表详情页-对象列表

展示绑定该映射表的对象的基本信息，包括对象编号、对象名称、对象型号、所属客户、对象状态及创建时间。支持按对象编号和对象名称进行搜索和刷新。



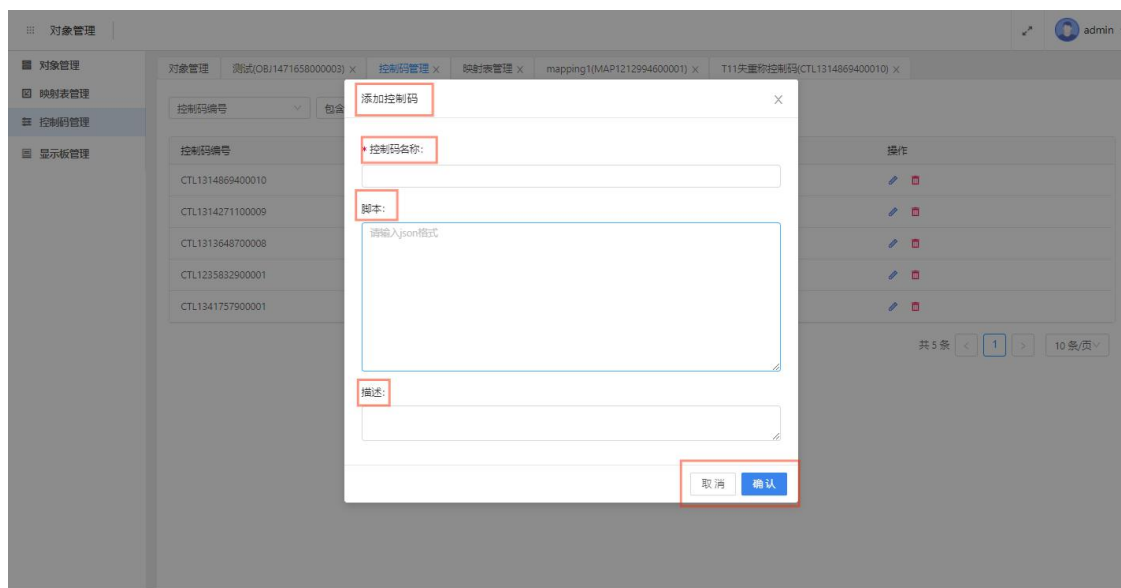
2.1.4 控制码管理

控制码描述对象的控制指令信息，将该指令信息提供发送到相应的接口，可进行对象的反向控制。如调节发动机的有功功率、调节锅炉的目标压力

2.1.4.1 配置说明

控制码添加

点击列表页的添加按钮，进入控制码添加界面，录入相应字段信息，点击确认，即完成控制码的添加。



字段说明

控制码名称：必填项，用户自定义。

描述：非必填项，用户自定义。

脚本：必填项，此处用于填写控制码脚本。

脚本规范

控制码脚本：

```
[
  ["Control_01", "Set Temperature", "设置温度", "OBJ", "$AprusID-1.L1_2_123", ""],
  ["Control_02", "Device Boot", "设备开机", "OBJ", "$AprusID-1.L1_2_234", "1"],
  ["Control_03", "Device Shutdown", "设备关机", "OBJ", "$AprusID-1.L1_2_345", "0"],
  ["Upgrade", "Aprus Upgrade", "适配器升级", "APR", "$AprusID-1", ""],
  ["Reboot", "Aprus Reboot", "适配器重启", "APR", "$AprusID-1", ""]
]
```

脚本：[Command, Label_En, Label_Cn, Category, Key, Value]。

第一列：控制指令（Command），分为自定义指令 和 保留指令。其中，自定义指令可随意指定，如：Control_03，保留指令是针对适配器的预定义指令，见下表：

类型	说明	可选控制值
----	----	-------

类型	说明	可选控制值
Upgrade	适配器升级	LUA, REMOSU, MCU, FTPLUA, FTPMCU

Reboot 适配器重启

第二列：英文标签（Label_En）：英文标签 即 英文说明。

第三列：中文标签（Label_Cn）：中文标签 即 中文说明。

第四列：控制类型（Category）：

控制类型	说明
------	----

OBJ	对象控制
-----	------

APR	适配器控制
-----	-------

其中，控制类型为 APR 时，控制指令是预定义指令 Upgrade、Reboot。

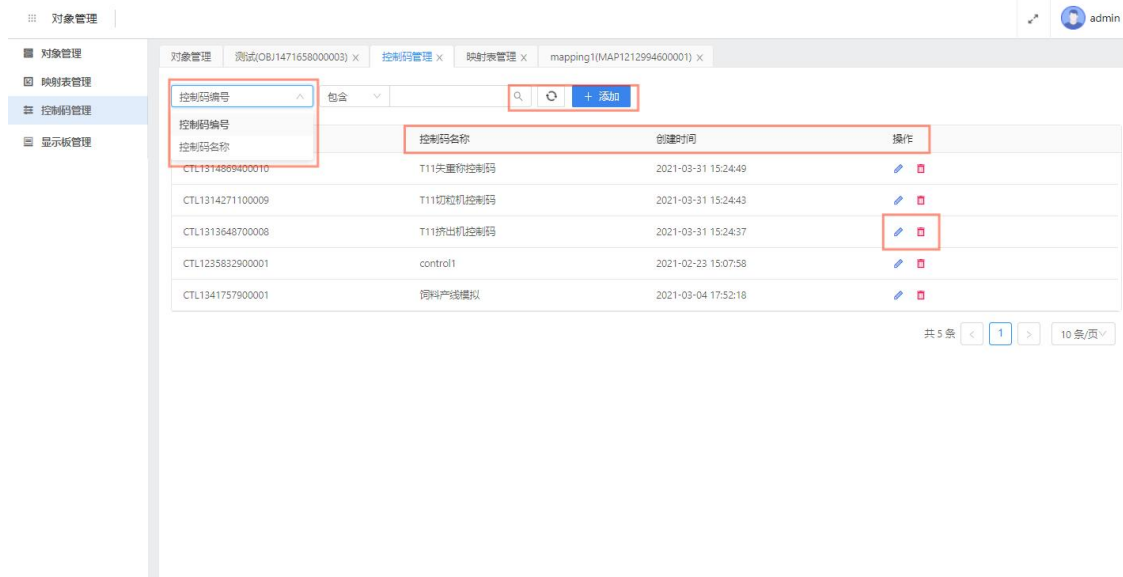
第五列：控制点位（Key）：\$AprusID-1.L1_2_123，表示第一个适配器的 L1_2_123 的点位。

第六列：控制值（Value）：控制值为空时，表示用户自定义填写；控制值不为空时，则发送脚本中的控制值。

2.1.4.2 界面说明

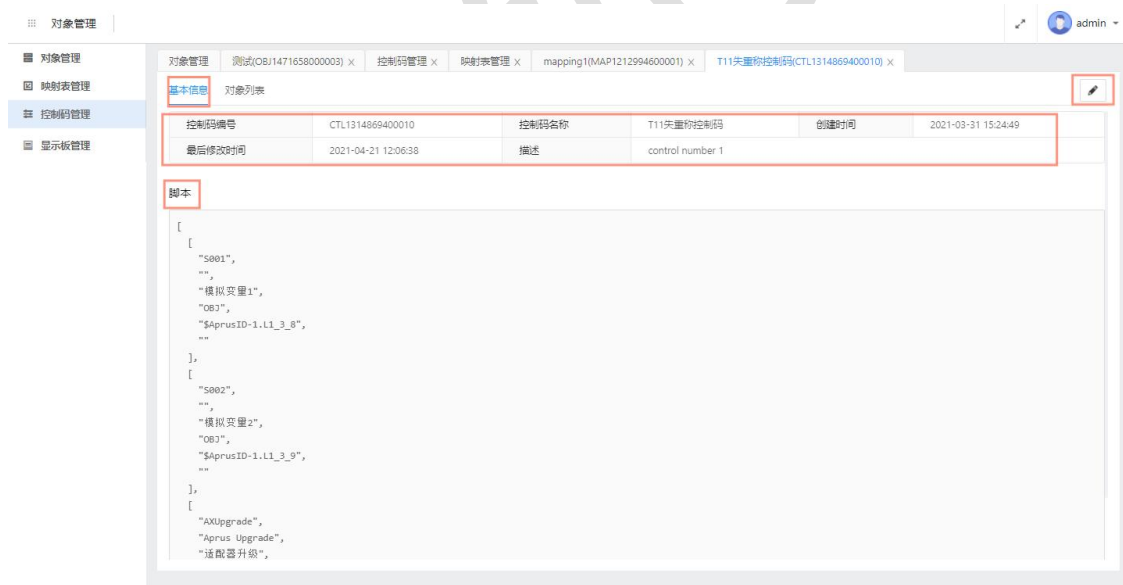
控制码列表页

展示配置的所有控制码列表，在本页面可以简单了解所有控制码的编号、名称及创建时间等基本信息，支持按控制码编号和控制码名称进行搜索及刷新，同时提供了控制码的添加、编辑和删除功能。



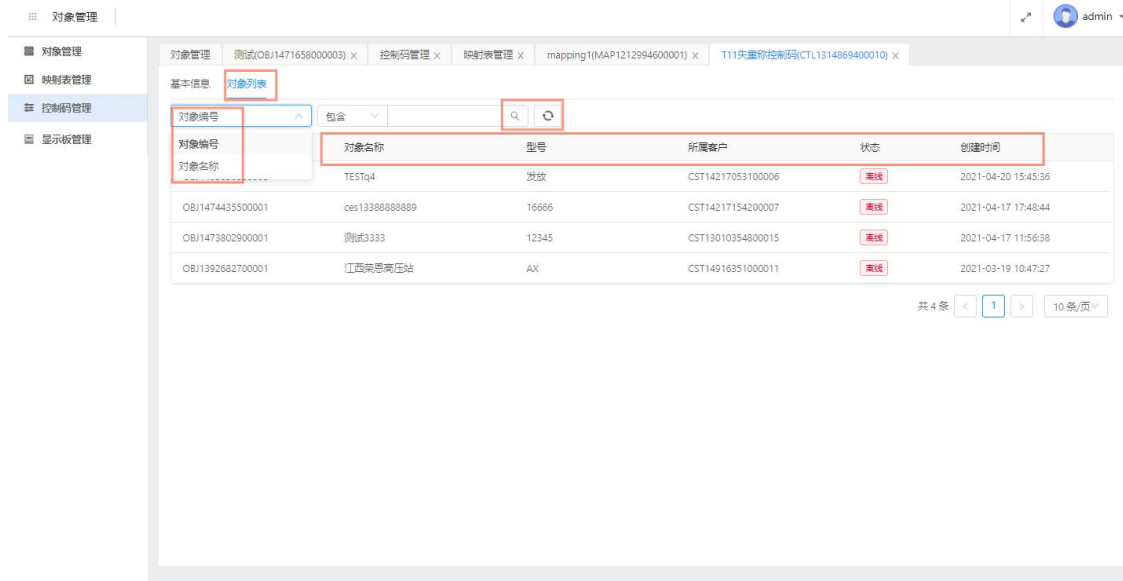
控制码详情页-基本信息

展示所选控制码的详细信息，包括控制码编号、名称、创建时间、最后修改时间、控制码脚本，同时提供控制码的编辑功能。



控制码详情页-对象列表

展示绑定该控制码的对象的基本信息，包括对象编号、对象名称、对象型号、所属客户、对象状态及创建时间。支持按对象编号和对象名称进行搜索和刷新。



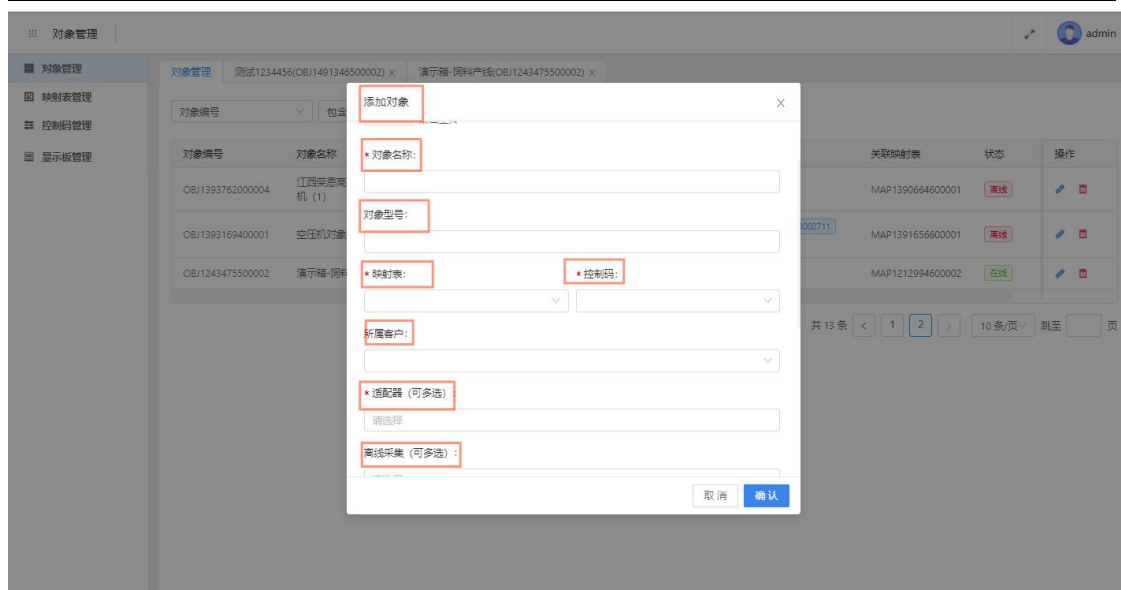
2.1.5 对象管理

将相关的所有配置如映射表、控制码、离线数据、适配器、客户等进行关联合一，以达到在系统上对对象各类数据的综合应用。

2.1.5.1 配置说明

对象添加

点击列表页的添加按钮，进入对象添加界面，录入相应字段信息，点击确认，即完成对象的添加。



字段说明

对象名称：必填项，用户自定义

上传图片：非必填项，用户可自由选择

对象型号：非必填项，用户自定义

映射表：必填项，单选，下拉选择映射表

控制码：必填项，单选，下拉选择控制码

所属客户：对象归属的客户，必填项，单选，下拉选择客户

适配器：设备数据对应的终端，必填项，支持多选，下拉选择适配器

离线采集：用于采集适配器不能直接获取的数据，如设备正在生产的产品的名称，非必填项，支持多选，下拉选择离线数据项目。注意，所选的离线数据项目必须是开启状态，同时将离线数据项目按相应脚本规范关联到该对象对应的映射表后，采集的离线数据才能在历史数据中进行查询展示。

子对象：非必填项，支持多选，下拉选择子对象，注意对象之间不能相互作为彼此的子对象，如 a 对象和 b 对象，若已选择 b 为 a 的子对象，则不能再选择 a 作为 b 的子对象。

序列号：非必填项，用户自定义

省份：非必填项

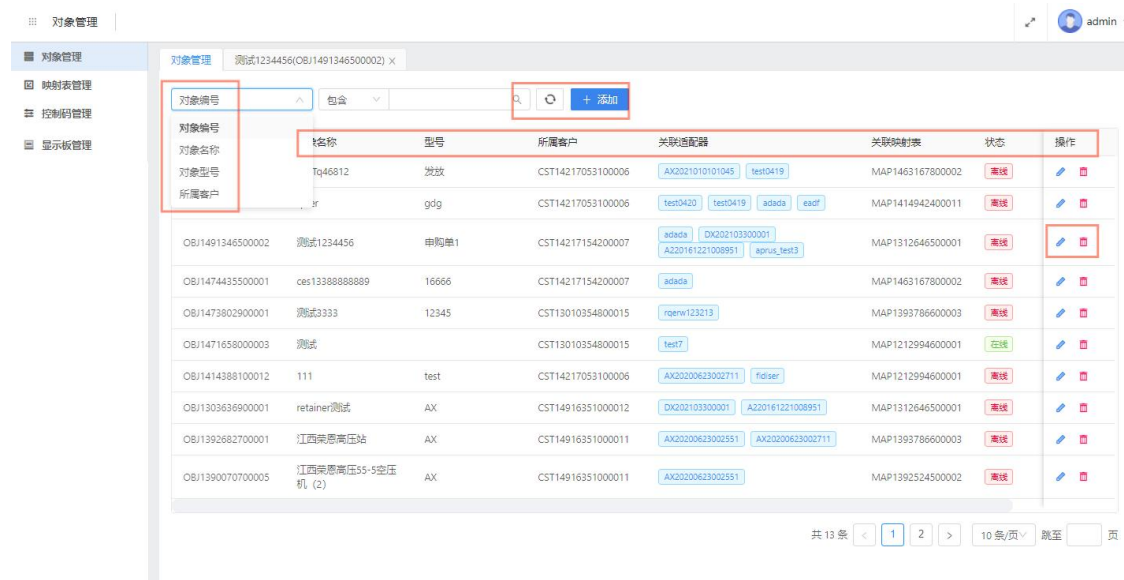
城市：非必填项

描述：非必填项

2.1.5.2 界面说明

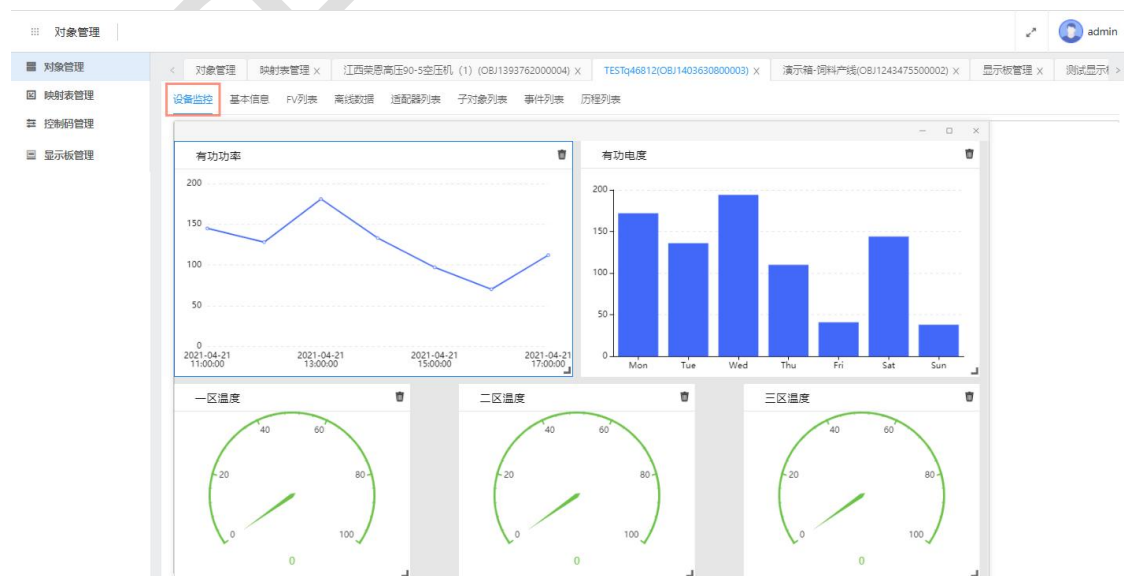
对象列表页

展示配置的所有对象列表，在本页面可以简单了解所有对象的编号、名称、型号、所属客户、关联适配器、关联映射表、对象状态及创建时间等基本信息，支持按对象编号、对象名称、对象型号和所属客户进行搜索及刷新，同时提供了对象的添加、编辑和删除功能。



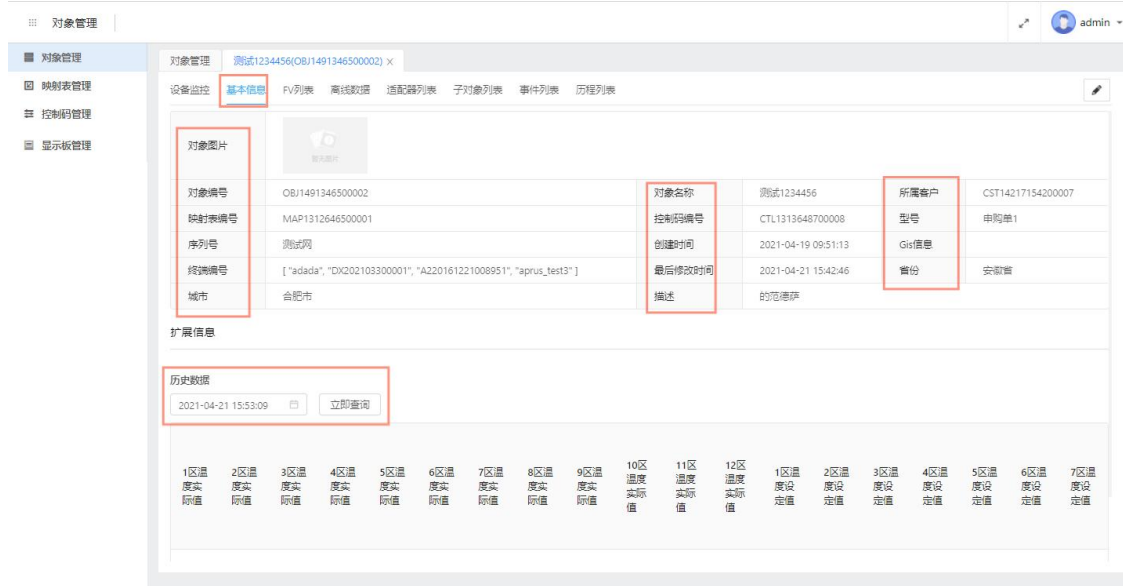
对象详情页-设备监控

展示所选设备的实时运行状态。



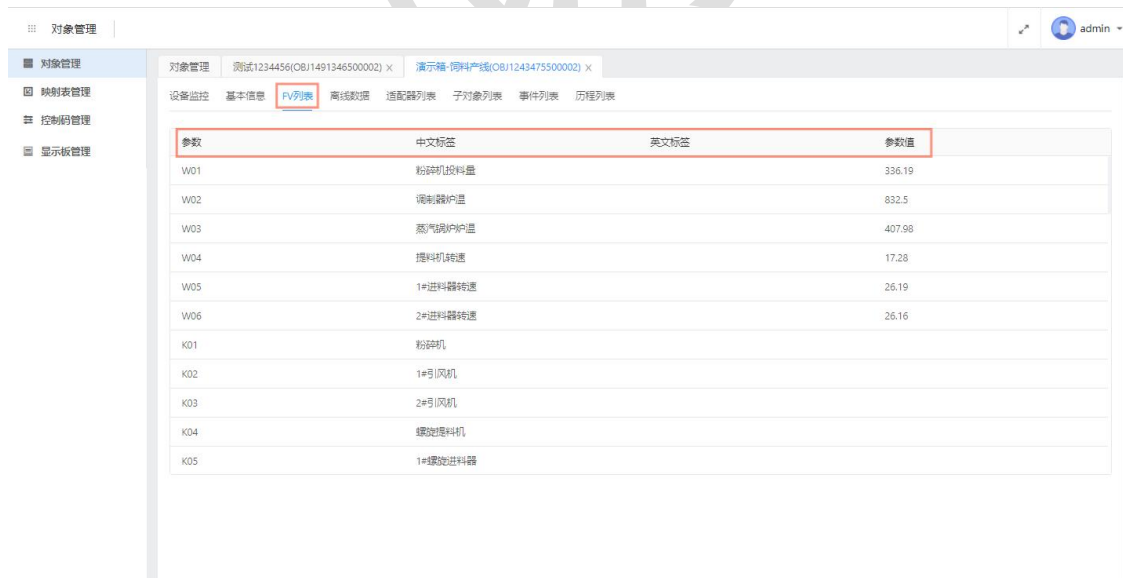
对象详情页-基本信息

展示所选对象的详细信息，包括对象图片、编号、名称、所属客户、映射表编号、控制码编号、型号、序列号、终端编号、省市、GIS 信息、创建时间、最后修改时间、所选映射表的扩展信息，同时提供对象的编辑功能及对象任一历史时刻的数据查询功能。



对象详情页-FV 列表

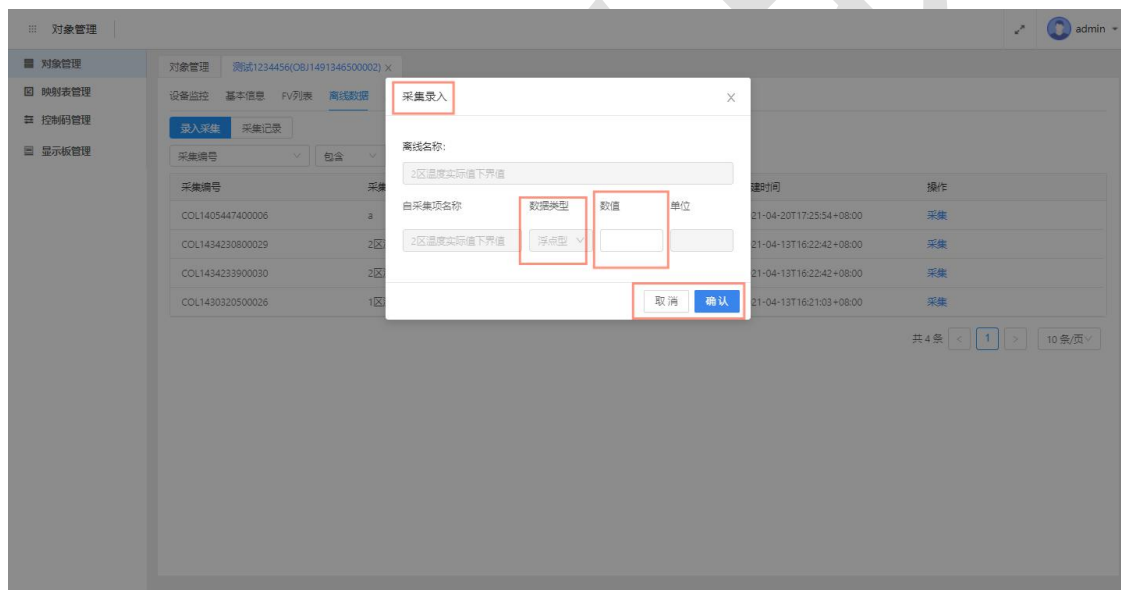
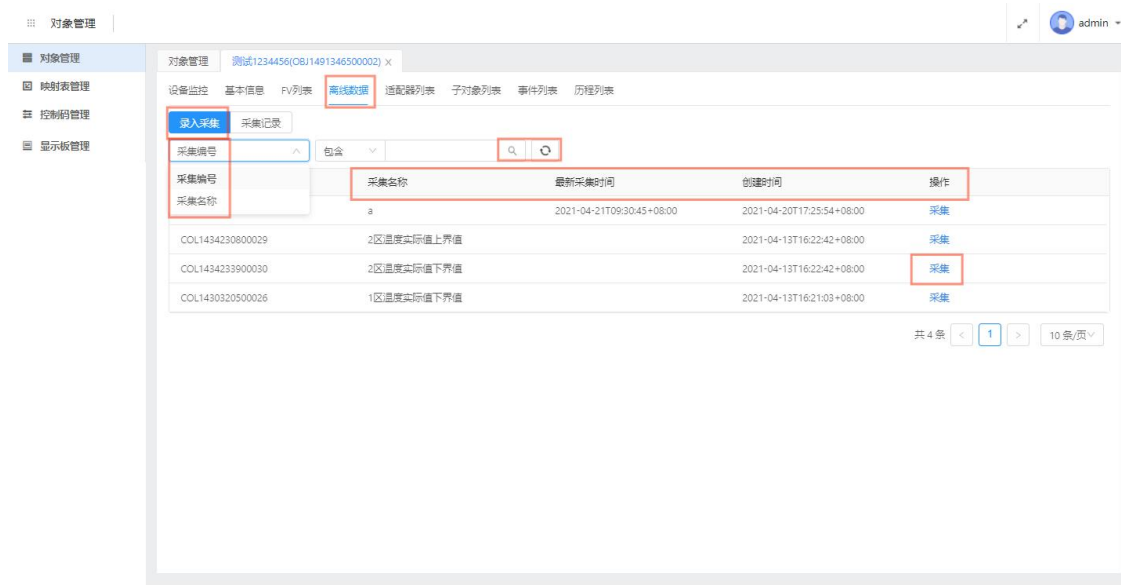
展示所选对象关联的映射表参数及各参数的实时数据。



对象详情页-离线数据

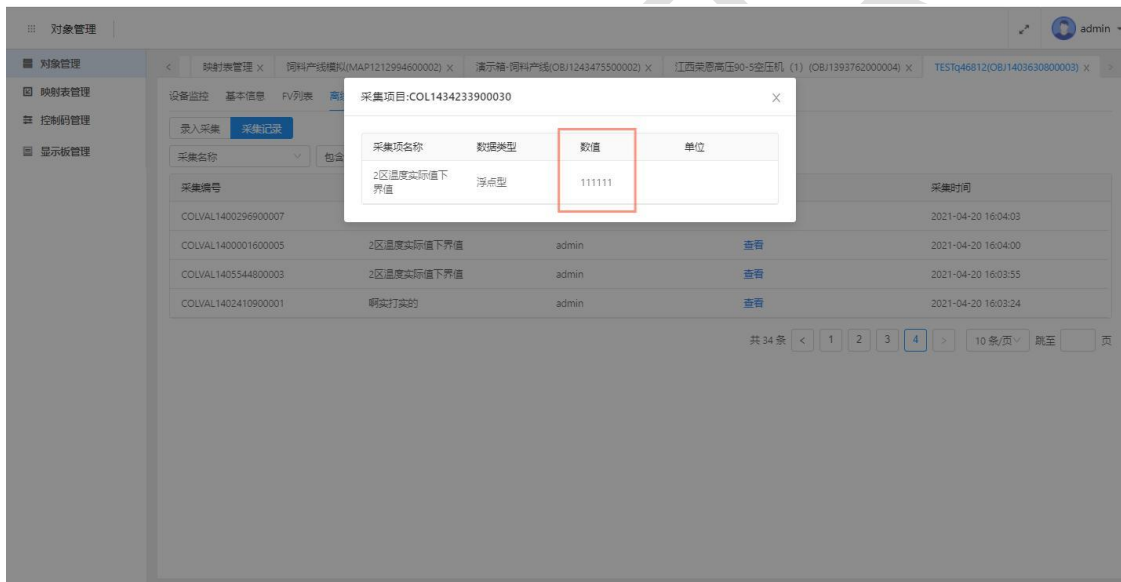
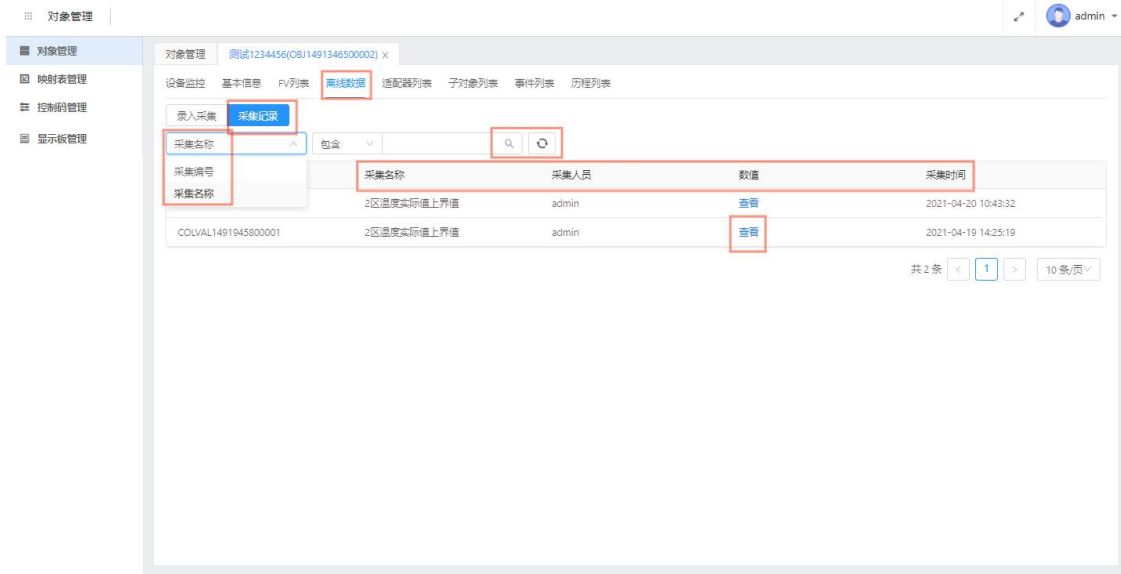
展示所选对象关联的离线数据（采集）项目，支持按采集编号和采集名称搜索及刷新，点击采集，弹出离线数据采集录入界面，可在该界面进行离线数据录入，注意录入的数值与数据

类型应匹配。



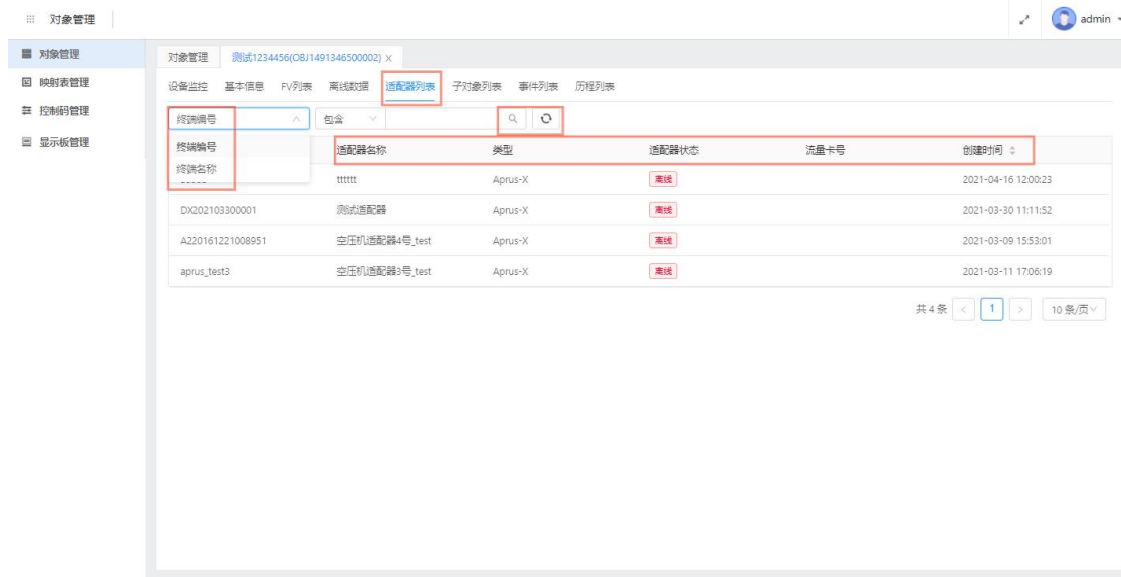
对象详情页-离线数据-采集记录

展示所选对象关联的离线数据项目的采集记录，支持按采集编号和采集名称搜索及刷新，点击查看，可展示采集录入的具体数值。



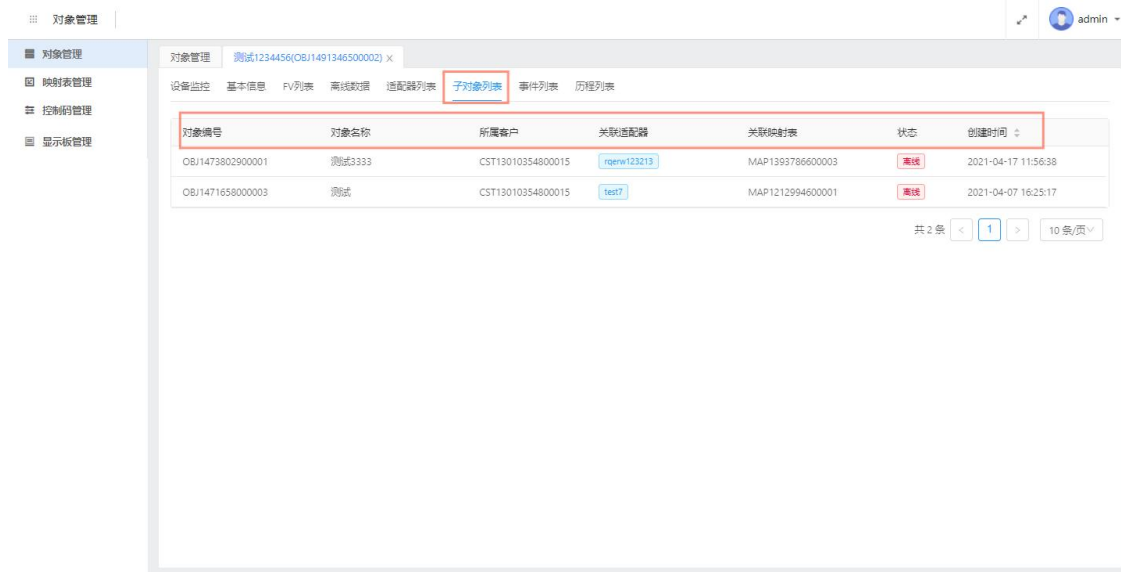
对象详情页-适配器列表

展示所选对象关联的适配器列表，可查看适配器编号、名称、类型、适配器状态、流量卡号，支持按终端编号和终端名称对适配器搜索及刷新。



对象详情页-子对象列表

展示所选对象关联的子对象列表，可查看子对象的编号、名称、所属客户、关联适配器、子对象状态。



对象详情页-事件列表

展示所选对象配置的事件的历史记录，可查看事件编号、名称、事件编码、发生时间及对应的事件值。支持按事件编号及事件名称搜索及刷新，同时可选取开始时间和结束时间，展示某一时间区间的事件记录。

事件编号	事件名称(中文)	事件英文名	事件编码	事件值	发生时间
	一区温度高结束	stop event	1112	832.95	2021-04-21 17:08:18
EVT1411726318650	一区温度高结束	stop event	1112	832.73	2021-04-21 17:08:17
EVT1412625918550	一区温度高结束	stop event	1112	832.95	2021-04-21 17:07:26
EVT1412525918546	一区温度高结束	stop event	1112	832.73	2021-04-21 17:07:25
EVT1411925718537	一区温度高结束	stop event	1112	832.95	2021-04-21 17:07:19
EVT1411725918530	一区温度高结束	stop event	1112	832.73	2021-04-21 17:07:17
EVT1415326418243	一区温度高结束	stop event	1112	832.95	2021-04-21 17:04:53
EVT1415226318239	一区温度高结束	stop event	1112	833.17	2021-04-21 17:04:52
EVT1413826418090	一区温度高结束	stop event	1112	832.95	2021-04-21 17:03:38
EVT1413716018084	一区温度高结束	stop event	1112	832.73	2021-04-21 17:03:37

对象详情页-历程列表

展示所选对象配置的历程的历史记录，可查看历程编号、名称、历程编码、历程开始时间、历程结束时间及历程持续时长。支持按历程编号及历程名称搜索及刷新，同时可选取开始时间和结束时间，展示某一时间区间的历程记录。

历程编号	名称(中文)	历程名称(英文)	历程状态	统一编码	开始时间	结束时间	持续时长	创建时间
	故障历程	FLT_EVENT	关闭	RV1	2021-04-09 17:45:52	2021-04-13 17:01:11	95h15m19.452401076s	2021-04-09 17:45:51
RETAINER1494455700009	故障历程	FLT_EVENT	关闭	RV1	2021-04-09 16:38:45	2021-04-09 17:45:13	1h6m28.15593266s	2021-04-09 16:38:44
RETAINER1495444900008	报警历程	ALM_EVENT	关闭	RV2	2021-04-09 16:37:54	2021-04-09 16:38:42	47.089292991s	2021-04-09 16:37:54
RETAINER1490924100007	故障历程	FLT_EVENT	关闭	RV1	2021-04-09 16:36:09	2021-04-09 16:37:03	54.117970933s	2021-04-09 16:36:09
RETAINER1492077200006	报警历程	ALM_EVENT	关闭	RV2	2021-04-09 16:34:21	2021-04-09 16:35:36	1m15.429827161s	2021-04-09 16:34:20
RETAINER1300821900025	故障历程	FLT_EVENT	关闭	RV1	2021-03-30 15:01:08	2021-03-30 15:01:09	971.766036ms	2021-03-30 15:01:08
RETAINER1303921	设备会集	start_stop	关闭	RV1	2021-03-30	2021-03-30	1.106879319s	2021-03-30

2.1.6 显示板管理

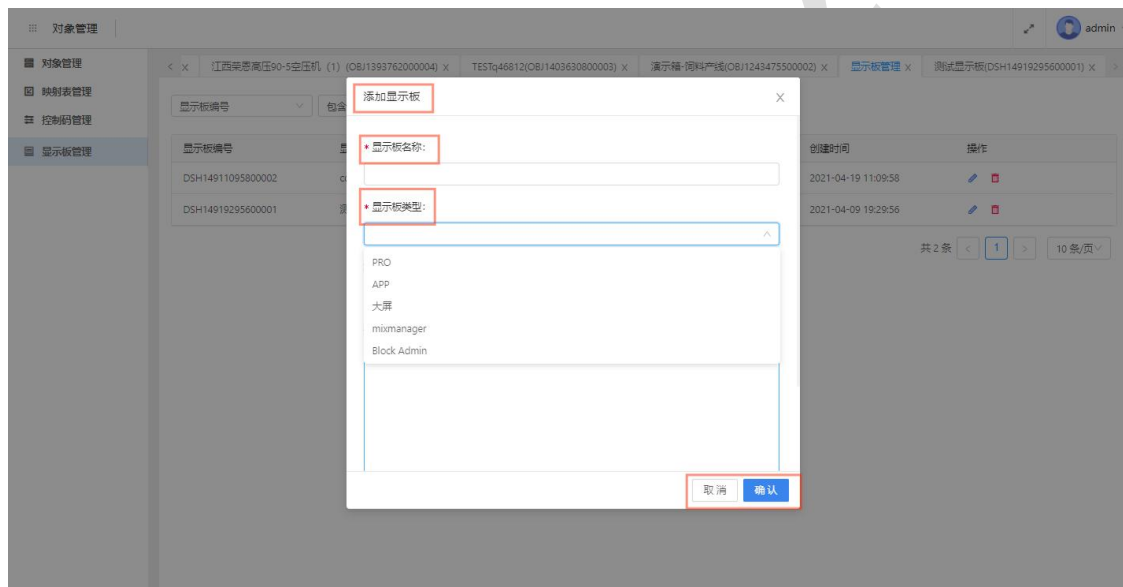
用于对象监控页面的图标配置，根据设备特性，将对象原始数据及进行统计、分析、整理后

的数据，通过合适的图标组件进行更直观的展示。

2.1.6.1 配置说明

显示板添加

点击列表页的添加按钮，进入显示板添加界面，录入相应字段信息，点击确认，即完成显示板的添加。



字段说明

显示板名称：必填项，用户自定义。

显示板类型：必填项，单选，下拉选择显示板类型，目前支持 PRO、APP、大屏、mixmanager、Block Admin。

绑定对象：非必填项，多选，下拉选择要绑定的对象。

脚本：必填项，此处用于填写显示板脚本。

脚本规范

显示板脚本：

略

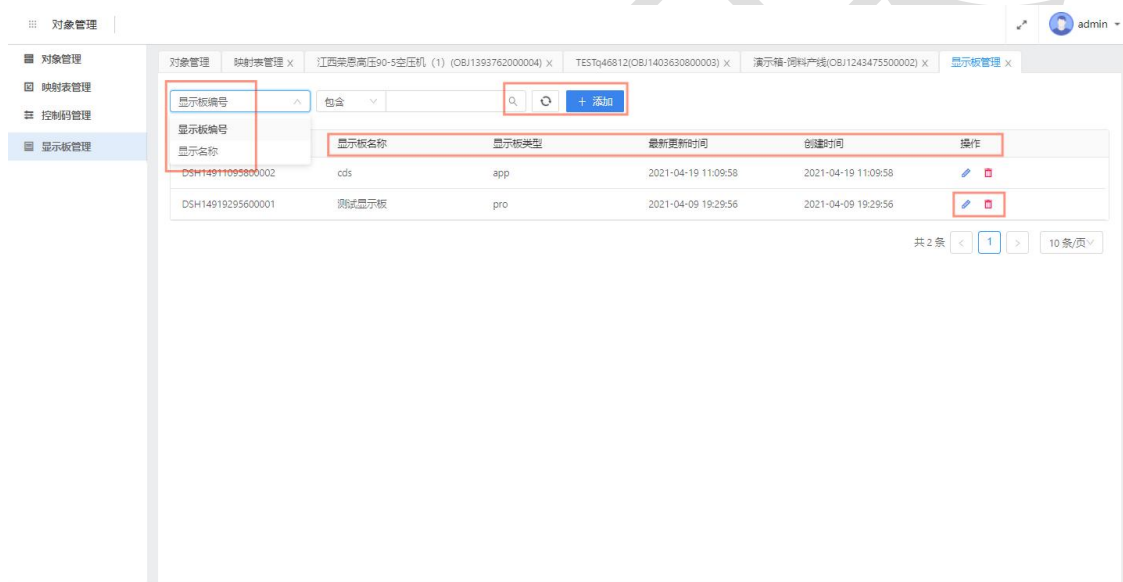
注意：显示板脚本除了手动编辑还可以通过 DD 自动生成，在 Dashbos（显示板数据服务）

中配置项目，作为显示板的数据源，在 Dashboard Designer（显示板设计）中利用组件设计显示板样式，保存后即可生成显示板脚本，将生成的脚本直接复制到上述显示板添加界面的脚本处即可。其中，Dashbos 和 Dashboard Designer 的使用可参照相应的使用介绍文档。

2.1.6.2 界面说明

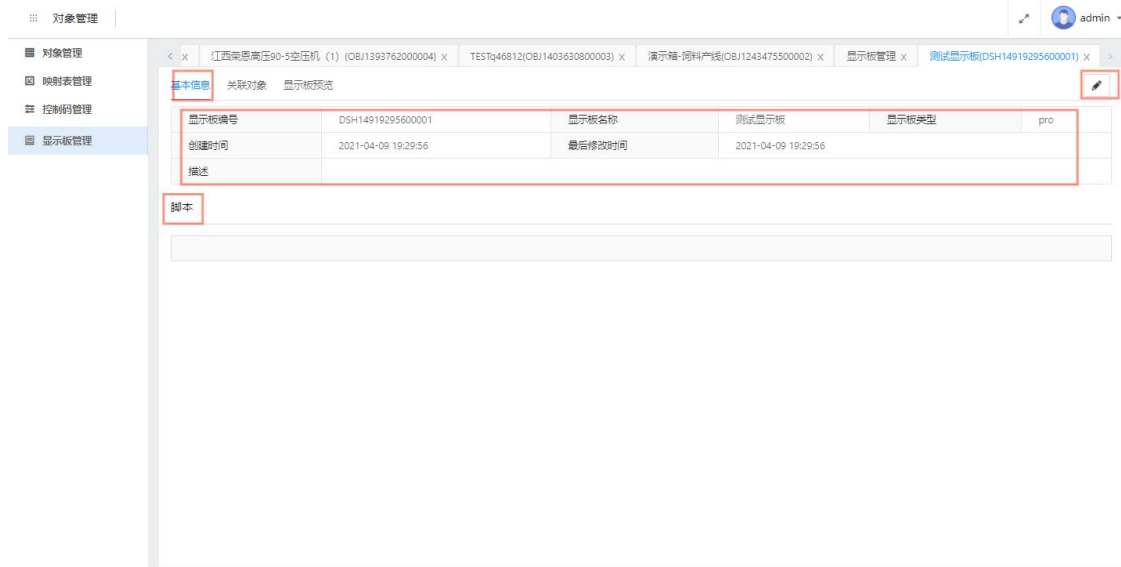
显示板列表页

展示配置的所有显示板列表，在本页面可以简单了解所有显示板的编号、名称、显示板类型及创建时间等基本信息，支持按显示板编号和显示板名称进行搜索及刷新，同时提供了显示板的添加、编辑和删除功能。



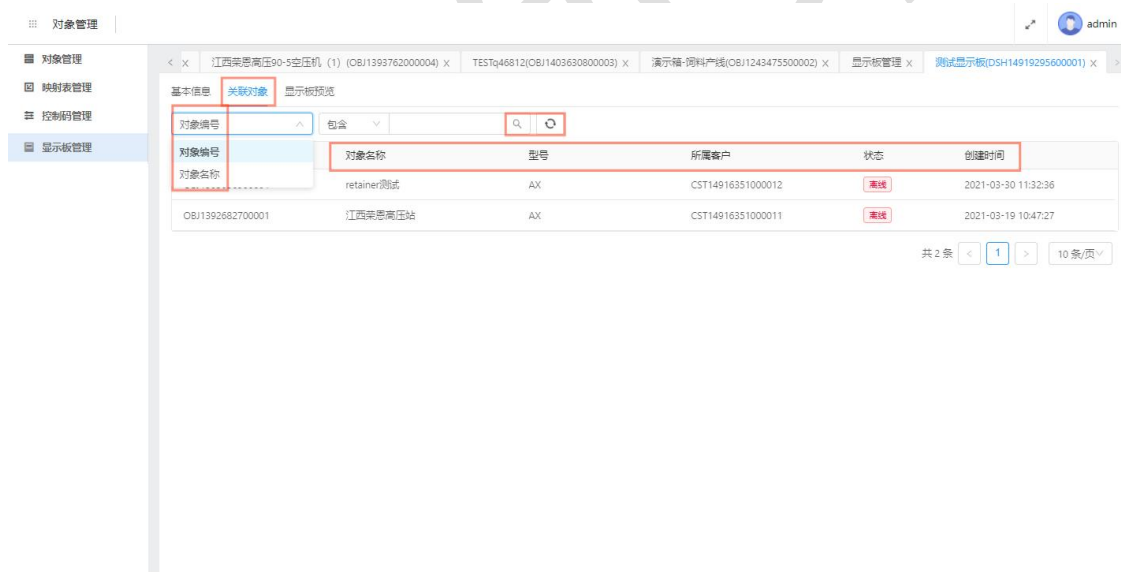
显示板详情页-基本信息

展示所选显示板的详细信息，包括显示板编号、名称、显示板类型、创建时间、最后修改时间及显示板脚本，同时提供显示板的编辑功能。



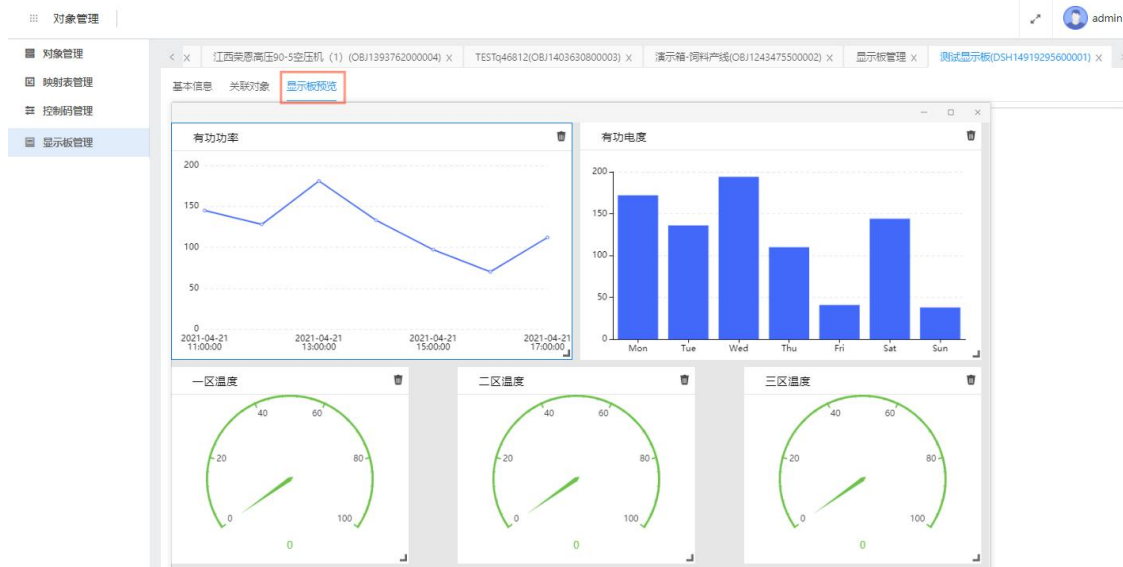
显示板详情页-关联对象

展示该显示板绑定的对象的基本信息，包括对象编号、对象名称、对象型号、所属客户、对象状态及创建时间。支持按对象编号和对象名称进行搜索和刷新。



显示板详情页-显示板预览

展示该显示板的预览模型，了解显示板所选组件的基本搭配效果。



2.2 终端管理-Terminal

2.2.1 应用场景

终端管理主要用于终端的添加、管理、grid 数据查询以及终端 lua 的合成和升级等。

2.2.2 菜单功能

2.2.2.1 终端

终端管理菜单，进行终端的添加、编辑、查看、删除等操作

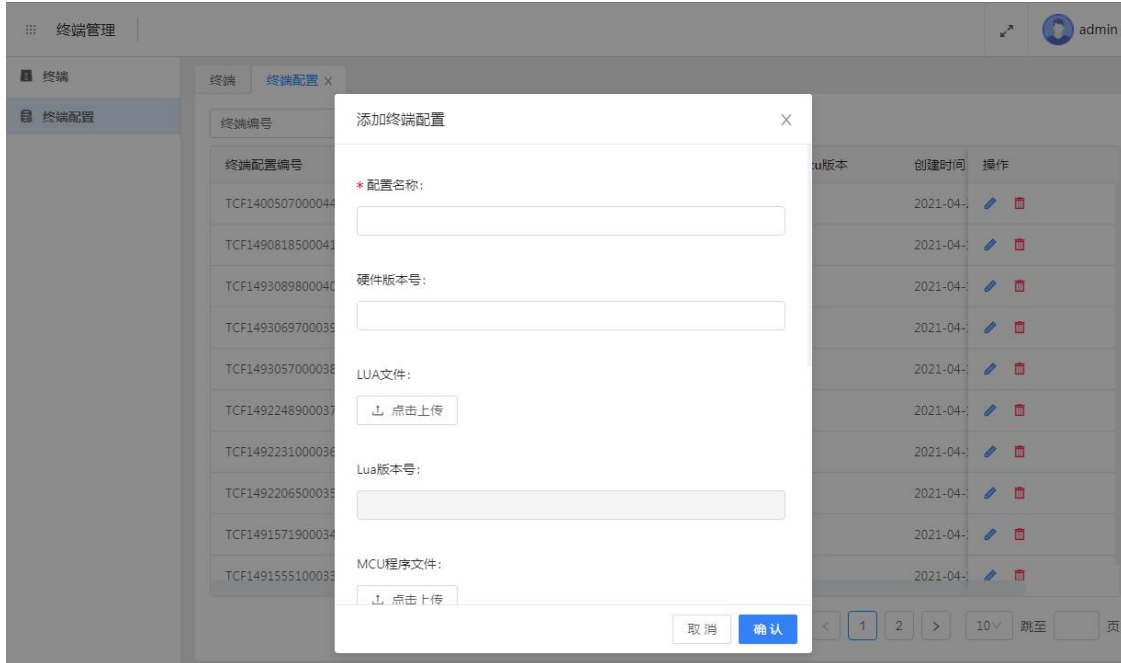
2.2.2.2 终端配置

终端配置管理菜单，进行终端配置的添加、编辑、查看、删除等操作

2.2.3 使用流程

2.2.3.1 添加终端配置

点击进入【终端配置】->【添加】->输入必要信息->【确认】保存



2.2.3.2 Lua 合成

从终端配置列表点击进入某一具体的终端配置详情->【Lua 脚本】->【编辑】输入 Aprus 脚本->【编辑】输入 Config 脚本->【合成文件】

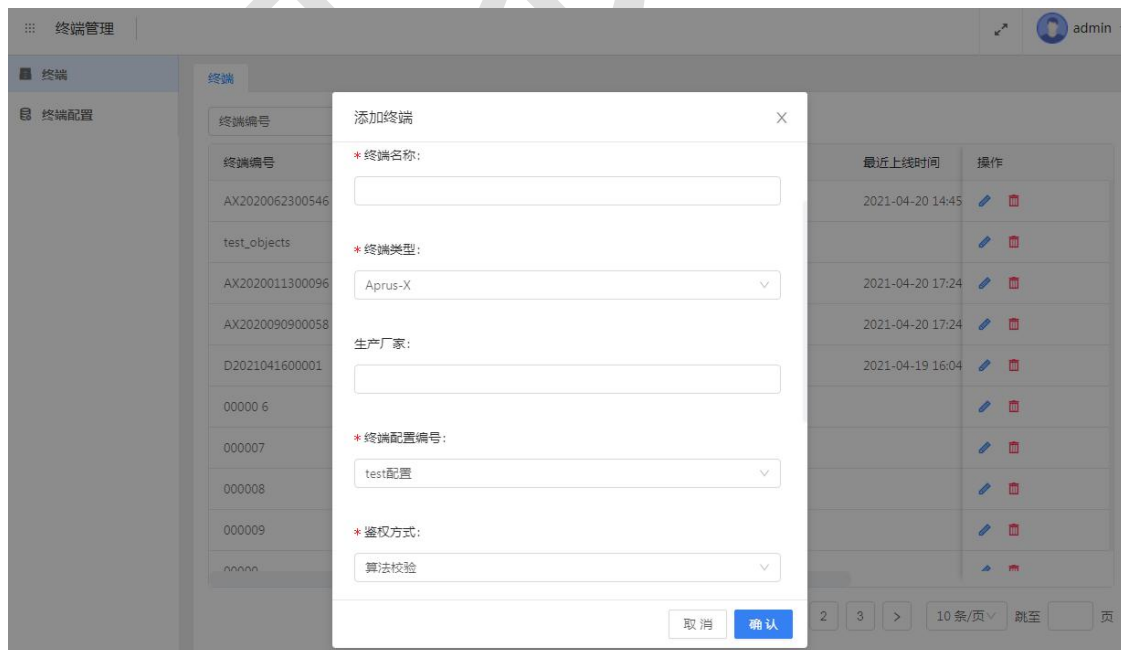


2.2.3.3 添加终端

点击进入【终端】->【添加】->输入必要信息->【确认】保存

终端配置编号：关联上述添加的终端配置

鉴权方式：算法校验（系统自动计算匹配密码校验，主要用于真实适配器），密码校验（通过鉴权用户名和验证码校验）



2.2.3.4 Lua 升级

从终端列表点击进入某一具体的终端详情->【基本信息】->Lua【升级版本】

终端编号	AX2020090900058	终端名称	升级测试适配器	创建时间	0001-01-01 00:00:00
终端类型	Aprus-X	生产厂家		最后修改时间	0001-01-01 00:00:00
终端状态	在线	信号值	0.0	最后上线时间	2021-04-20 17:24:47.198614223
鉴权方式	密码校验	鉴权用户名	mix	鉴权验证码	mix123
终端配置编号	TCF1464026500001	报文权限	全部	报文主题	all
Lua版本号	MAX.LUA.V032700.R	Muc版本号	MAX.APP.V020024.RC 升级版本	Remosu版本号	MA5207.R.V100.R
流量卡号					
描述					

2.2.3.5 查询在线记录

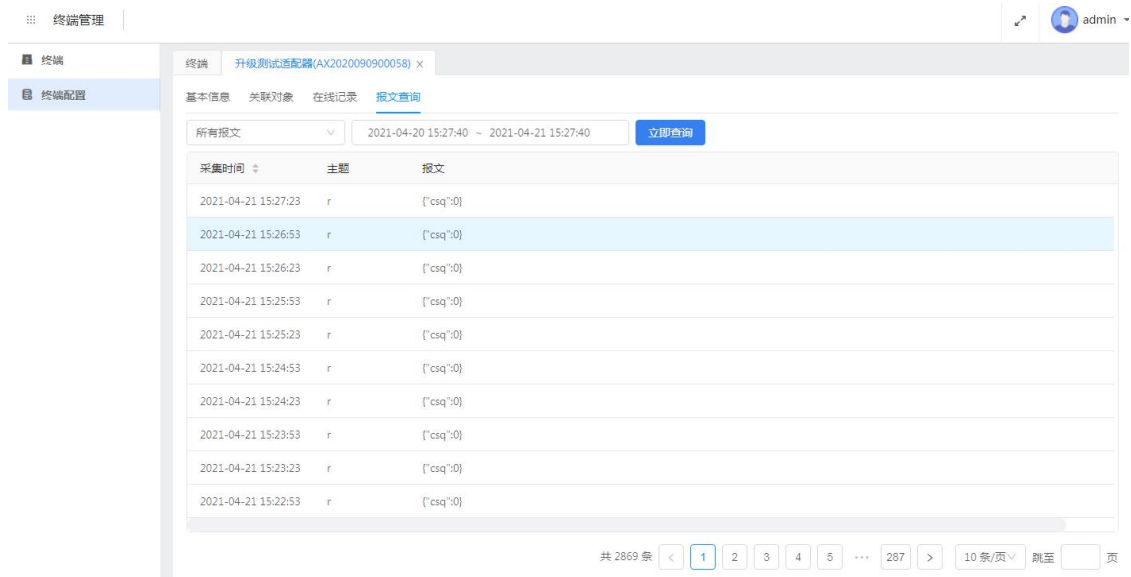
终端详情中点击【在线记录】

上下线状态	状态描述	发生时间
上线		2021-04-20 17:24:47
上线		2021-04-18 17:19:29
上线		2021-04-17 16:55:15

共 3 条 < 1 > 10 条/页

2.2.3.6 报文查询

终端详情中点击【报文查询】



The screenshot shows the 'Terminal Management' (终端管理) interface. The selected terminal is '升级测试透配器(AX2020090900058)'. The 'Report Query' (报文查询) tab is active, displaying a table of reports for the date range 2021-04-20 15:27:40 to 2021-04-21 15:27:40. The table has three columns: '采集时间' (Collection Time), '主题' (Subject), and '报文' (Message). The '报文' column contains the value '['csq':0]' for all entries. The interface also includes a search bar, a '立即查询' (Query Now) button, and a pagination bar at the bottom showing '共 2869 条' (Total 2869 items) and '10 条/页' (10 items per page).

采集时间	主题	报文
2021-04-21 15:27:23	r	['csq':0]
2021-04-21 15:26:53	r	['csq':0]
2021-04-21 15:26:23	r	['csq':0]
2021-04-21 15:25:53	r	['csq':0]
2021-04-21 15:25:23	r	['csq':0]
2021-04-21 15:24:53	r	['csq':0]
2021-04-21 15:24:23	r	['csq':0]
2021-04-21 15:23:53	r	['csq':0]
2021-04-21 15:23:23	r	['csq':0]
2021-04-21 15:22:53	r	['csq':0]

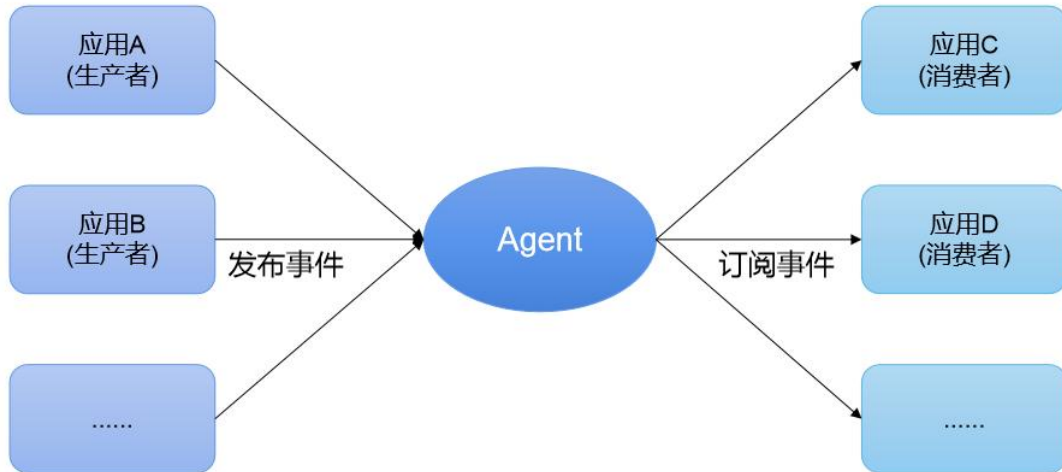
2.2.4 其他说明

终端中的【关联对象】需要在“对象管理”中操作关联后，方能查看相关联的对象列表

2.3 代理服务-Agent

2.3.1 简介

Agent 是中央事件代理服务，支持同步和异步两种不同代理方式。Mixiot 是一个事件驱动的实时高效系统，每一个应用服务即是事件的生产者、也是事件的消费者；生产者给 Agent 发布事件，消费者向 Agent 订阅事件。



2.3.2 事件订阅

事件订阅支持有两类：同步方式 和 异步方式。订阅者每次订阅成功之后，会获得关于这次订阅的一个 UID。同步方式返回 UID 前缀 EVNT，异步方式返回 UID 前缀 TASK。

2.3.2.1 同步方式

描述

同步方式是一种长连接方式。同步方式下，发起订阅的一方必须按照格式描述的要求发送 Json 数据到 Agent；在与 Agent 建立连接之后，需要与 Agent 保持连接。此后，Agent 会将匹配到的消息发通过这个连接发送给订阅方，一旦订阅方断开连接，则 Agent 会根据这个断开的连接自动取消订阅。

同步方式目前只支持 TCP,Websocket 方式

格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	sub_event	方法名
actionid	否	String		活动标识
event	是	Json		指明 agent 类型是 event

参数名称	是否必选	数据类型	数据约束	描述
+ event	是	String		
+ ...	否	依据实际情况而定		依据实际情况而定

示例

```
{
  "action": "sub_event",
  "actionid": "abc",
  "event": {
    "event": "timer_event",
    "type": "one_second_event"
  }
}
```

返回

```
{
  "resp": "sub_event",
  "actionid": "abc",
  "code": 200,
  "msg": "成功",
  "result": {
    "uid": "EVNT1463595100003"
  }
}
```

2.3.2.2 异步方式

描述

同步方式是一种短连接方式。异步方式下，发起订阅的一方必须按照格式描述的要求发送 Json 数据到 Agent；Agent 不关心当前的连接是否存在，当 Agent 匹配到消息之后，会根据

格式描述的回调地址和回调参数和消息进行回调。

异步方式支持 HTTP,TCP,Websocket,gRPC 等四种方式

格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	add_task	方法名
actionid	否	String		活动标识
task	是	Json		指明 agent 类型是 task
+ event	是	String		
+ ...	否	依据实际情况而定		依据实际情况而定
+ callback_addr	是	String		事件回调地址
++ callback_params	是	Json		
+++ action	是	String		
+++ method	否	String	"http tcp websocket grpc"	

订阅者必须确保 callback_addr 是一个可以访问的地址，并且 callback_addr 的方式和 callback_params.method 必须一致

示例

```
{
  "action": "add_task",
  "actionid": "abc",
  "task": {
```



```
"event": "timer_event",
"type": "one_second_event",
"callback_addr": "http://192.168.1.208:1235/mixlog/api",
"callback_params": {
    "action": "get_action_list",
    "actionid": "",
    "method": "http"
}
}
}
返回
{
    "resp": "add_task",
    "actionid": "abc",
    "code": 200,
    "msg": "成功",
    "result": {
        "uid": "TASK1460995400162"
    }
}
```

2.3.3 事件发布

描述

事件发布方将消息发送到 Agent 之后，Agent 会将发布的信息与订阅者信息进行匹配，匹配到之后，会将消息发送出去

采用同步订阅的方式，Agent 会将消息发送给订阅者

采用异步订阅的方式，Agent 则会将消息通过回调的方式发送出去

格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	publish_event	方法名
actionid	否	String		活动标识
event	是	Json		
+ event	是	String		事件标识
+ block	是	String		Block 标识(事件发布 Block)
+ uniqueid	是	String		事件发布 uniqueid 标识
+ datetime	是	String		事件发布时间
+ ...	是	依据实际情况而定		实际发布的数据

示例

```

{
  "action": "pub_event",
  "event": {
    "event": "timer_event",
    "block": "agent",
    "type": "one_second_event",
    "uniqueid": "PUB1344800006032",
    "datetime": "2021-03-04 17:13:48"
  }
}
    
```

返回

```

{
  "resp": "pub_event",
    
```

```

"code": 200,
"msg": "成功"
}
    
```

2.3.4 取消订阅

取消订阅的时候，必须通过 UID 来取消。特别的，在同步订阅模式下，并不一定需要显式的取消订阅，当订阅者断开连接的时候，Agent 会强制自动取消订阅。

2.3.4.1 同步方式

描述

格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	unsub_event	方法名
actionid	否	String		活动标识
uid	是	String		

示例

```

{
  "action": "unsub_event",
  "uid": "EVNT1343675700011"
}
    
```

返回

```

{
  "resp": "unsub_event",
  "code": 200,
  "msg": "成功"
}
    
```

```
}

```

2.3.4.2 异步方式

描述

格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	cancel_task	方法名
actionid	否	String		活动标识
uid	是	String		

示例

```
{
  "action": "cancel_task",
  "uid": "SUB1262842000001"
}
```

返回

```
{
  "resp": "cancel_task",
  "code": 200,
  "msg": "成功"
}
```

2.3.5 获取订阅列表

2.3.5.1 同步方式

描述

格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_event_list	方法名
actionid	否	String		活动标识
uid	是	String		

示例

```
{
  "action": "get_event_list"
}
```

返回

```
{
  "resp": "get_event_list",
  "code": 200,
  "msg": "成功",
  "result": [
    "EVNT1342988000006"
  ]
}
```

2.3.5.2 异步方式

描述

格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_task_list	方法名
actionid	否	String		活动标识

示例

```
{
  "action": "get_task_list"
}
```

返回

```
{
  "resp": "get_task_list",
  "code": 200,
  "msg": "成功",
  "result": [
    "TASK1344865700014"
  ]
}
```

2.3.6 获取订阅详情

2.3.6.1 同步方式

描述

格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_event_detail	方法名
actionid	否	String		活动标识

示例

```
{
  "action": "get_event_detail",
```

```

        "uid": "EVNT1342988000006"
    }

```

返回

```

{
    "resp": "get_event_detail",
    "code": 200,
    "msg": "成功",
    "result": {
        "event": "timer_event",
        "type": "one_second_event"
    }
}

```

2.3.6.2 异步方式

描述

格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_task_detail	方法名
actionid	否	String		活动标识

示例

```

{
    "action": "get_task_detail",
    "uid": "TASK1334562600001"
}

```

返回

```

{
  "resp": "get_task_detail",
  "code": 200,
  "msg": "成功",
  "result": {
    "callback_addr": "http://192.168.1.208:1235/mixlog/api",
    "callback_params": {
      "action": "get_action_list",
      "method": "http"
    },
    "event": "timer_event",
    "type": "one_second_event"
  }
}
    
```

2.3.7 Agent 事件

Agent 服务自身会发布定时事件，即每分钟，每 5 分钟，每 30 分钟，每小时都会发布一个事件。Mixiot 系统的其他服务可以通过订阅定时事件执行定时任务。

事件名称	描述
one_minute_event	每分钟事件
five_minute_event	每 5 分钟事件
thirty_minute_event	每 30 分钟事件
one_hour_event	每小时事件

示例

```

{
  "action": "pub_event",
  "event": {
    
```



```

        "event": "timer_event",
        "block": "agent",
        "type": "one_minute_event",
        "uniqueid": "PUB1344800006032",
        "datetime": "2021-04-21 18:08:08"
    }
}
    
```

2.3.8 用户事件

暂不支持

2.3.9 Agent 应用

Mixiot 提供 Agent 管理应用，支持界面化配置事件异步订阅；其中，同步订阅不支持界面化配置。Agent 管理应用通过 Fidis 门户进入。



2.3.9.1 列表

Agent 应用负责配置管理事件异步的订阅，列表页面可以搜索订阅记录、添加、编辑、删除

订阅记录。

代理编号	代理事件	创建时间	操作
TASK1472442820089	trigger_event	2021-04-17 16:10:24	
TASK1435861700148	update_fv	2021-04-13 14:31:58	
TASK1435231200080	trigger_event	2021-04-13 14:30:52	

共 3 条 < 1 > 10 条/页

2.3.9.2 添加

添加页面实现订阅脚本的界面化配置，添加一条记录效果和请求调用异步订阅命令一样（action: add_task）。

添加代理

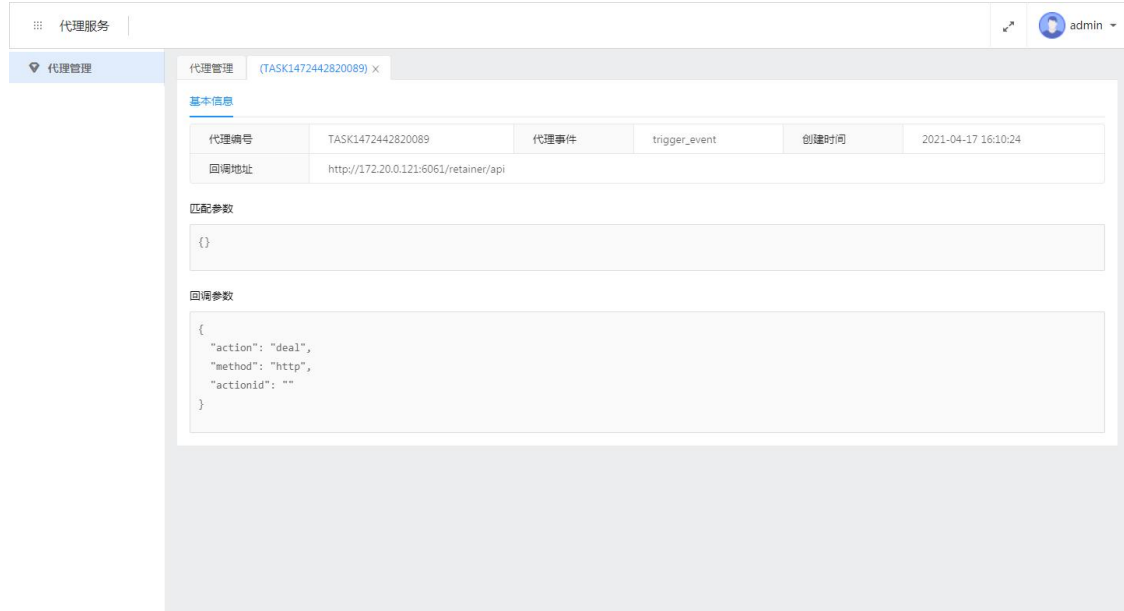
- * 代理事件: trigger_event
- 匹配参数: {"block_id": "mapping"}
- * 回调地址: http://172.20.0.121:6061/retainer/api
- * 回调参数: {"action": "deal", "method": "http", "actionid": ""}

```

{
  "task": {
    "event": "trigger_event",
    "block_id": "mapping",
    "callback_addr": "http://172.20.0.121:6061/retainer/api",
    "callback_params": {
      "action": "deal",
      "method": "http",
      "actionid": ""
    }
  }
}
    
```

2.3.9.3 详细

详细页面查看订阅的详细信息。



2.3.10 附录

Agent 的 ip 地址通过 consul 动态获取，接口调用方式为：<http://ip:port/agnet/api>；其中 HTTP,TCP,Websocket,gRPC 的端口如下：

协议	端口
HTTP	6061
TCP	6063
Websocket	6065
gRPC	6067

2.4 离线数据-Collect

2.4.1 简介

Collect 是离线数据服务,提供接口采集数据、并将数据推送给 Mapping 拼接马赛克。Collect 离线数据采集前,要先创建采集数据类型;在 Block 模式下,建立一个项目就是创建一个离线数据类型。

2.4.2 Collect 项目脚本

离线数据脚本是采集数据的变量配置脚本。创建一个离线数据项目,采集的数据可以是单值或者多值,一行脚本则指定一个值。

```
[  
  {  
    "key": "length",  
    "label_en": "Label_En",  
    "label_loc": "长度",  
    "type": "int",  
    "unit": "米",  
    "source": "API",  
    "args": {}  
  }  
]
```

key 是数据的键,自定义填写,一般是英文与数字标识;

label_en 是英文标签;

label_loc 是中文标签;

type 是离线数据类型,比如 Float(小数)、Integer(整数)、String(字符)等;

unit 是数据的单位,比如°C、m、kpa 等;

source 是数据的来源,目前有 API(手工 API 录入),STT(统计计算),INDASS(Indass 结果);

args 是数据来源需要配置的一些参数,当 source 是 API 时不需要填,其他来源需要配置

一些特定的参数;

2.4.2.1 source 与 args 字段说明

当 source 不是 API 手工录入时, 需要配置特定来源的参数

当 source 为 STT 时, args 需要 4 个字段:

type 定义获取统计值的方式, 可选 cycle 或 event, cycle 为定时获取对应统计值并录入, event 为监听对应事件, 事件触发时将相应统计值录入。

cycle 当 type 为 cycle 时必填, 为定时获取的周期, 格式为形如 12h12m12s 的字符串, 表示 12 小时 12 分钟 12 秒。

object_id 统计项目对应的对象 id

statistics_type 统计计算类型, 可以选择 statistics 或者 calculate

statistics_id 统计项目 id

例子:

```
{
  "type": "cycle",
  "cycle": "5m",
  "object_id": "OBJ1",
  "statistics_type": "statistics", // statistics: 统计 calculate: 计算
  "statistics_id": "STATISTICS1"
}
```

当 source 为 INDASS 时, args 需要 4 个字段:

type 同上

cycle 同上

block_id indass 的 block_id, 例如 index, map 等等

object_id 对象 id

project_id indass 项目 id

data_key indass 结果是一组键值对, 此项决定获取这组结果中的哪一个值录入

例子:

```
{  
  "type": "cycle",  
  "cycle": "5m",  
  "block_id": "index",  
  "object_id": "OBJ1",  
  "porject_id": "index1",  
  "data_key": "index_change_rate"  
}
```

2.4.2.2 脚本示例

```
[  
  {  
    "key": "length",  
    "label_en": "Label_En",  
    "label_loc": "长度",  
    "type": "int",  
    "unit": "米",  
    "source": "API",  
    "args": {}  
  },  
  {  
    "key": "how_many",  
    "label_en": "Label_En",  
    "label_loc": "几个",  
    "type": "int",  
    "unit": "个",  
    "source": "API",  
    "args": {}  
  }  
]
```

```

    },
    {
      "key": "indass_index_value",
      "label_en": "Label_En",
      "label_loc": "indass 计算值",
      "type": "float",
      "unit": "s",
      "source": "INDASS",
      "args": {
        "object_id": "OBJ1243475500003",
        "block_id": "index",
        "project_id": "test",
        "data_key": "index_change_rate",
        "type": "cycle",
        "cycle": "5s"
      }
    }
  ]

```

2.4.3 通过 API 进行数据采集

首先需要配置 Collect 的项目，创建一个新的离线项目，按脚本格式配置好，并启动对应项目。

数据采集接口会校验 collect_id 是否正确，对应项目是否启动。

value 为一组数据值，value 中的 key 需要与对应项目脚本中的 key 一一对应，脚本中不存在的数据会被丢弃。

数据采集接口参数示例如下

```

{
  "action": "add_collect_value",
  "actionid": ""
}

```

```

"name": "多数据采集",
"description": "",
"collect_id": "COL1332295612308",
"object_id": "OBJ1001",
"value": {
    "temp": 88.5,
    "pressure": 202
},
"file": "",
"collect_at": "2021-03-03 10:88:10",
"collector": "mixuser",
"reference": ""
}
    
```

2.4.4 推送

接口采集数据后，一方面将数据保存 MySQL 保存历史记录，另一方面将数据推送给 Mapping 拼接马赛克。推送具体方式以 Mapping 通道为准。

2.4.5 Collect 接口 action 说明

2.4.5.1 获取离线项目列表

获取离线项目列表

参数

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_project_list	方法名
actionid	否	String		活动标识
page_index	否	int		页数，不传则获取所有记录

参数名称	是否必选	数据类型	数据约束	描述
page_size	否	int		页大小，不传则获取所有记录
where_and	否	String		and 条件
where_or	否	String		or 条件
where_in	否	String		in 条件

示例

```
{
  "action": "get_project_list",
  "page_index": 1,
  "page_size": 20,
  "where_and": "",
  "where_or": "",
  "where_in": ""
}
```

响应

```
{
  "resp": "get_project_list",
  "actionid": "",
  "code": 200,
  "msg": "成功",
  "result": {
    "page_index": 1,
    "page_size": 20,
    "total_pages": 0,
    "total_records": 0,
    "data": [
      {
```

```
"id": 1969,  
"uid": "COL1472557300005",  
"name": "042701",  
"description": "",  
"block_id": "",  
"class_id": "",  
"is_active": 0,  
"option": null,  
"script": [  
  {  
    "args": [],  
    "key": "1#Temp",  
    "label_en": "",  
    "label_loc": "1 机组温度",  
    "source": "API",  
    "type": "Float",  
    "unit": ""  
  },  
  {  
    "args": [],  
    "key": "2#Temp",  
    "label_en": "",  
    "label_loc": "2 机组温度",  
    "source": "API",  
    "type": "Float",  
    "unit": ""  
  }  
],  
"created": "2021-04-27T15:28:25+08:00",
```

```

        "last_modified": "",
        "last_modified_by": 0,
        "is_available": 1
    }
]
}
}

```

2.4.5.2 获取离线项目

获取离线项目

参数

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_project_item	方法名
actionid	否	String		活动标识
uid	是	String		项目 uid

示例

```

{
  "action": "get_project_item",
  "uid": "COL1"
}

```

响应

```

{
  "resp": "get_project_item",
  "actionid": "",
  "code": 200,
  "msg": "成功",
  "result": {

```

```
"id": 1969,  
"uid": "COL1472557300005",  
"name": "042701",  
"description": "",  
"block_id": "",  
"class_id": "",  
"is_active": 0,  
"option": null,  
"script": [  
  {  
    "args": [],  
    "key": "1#Temp",  
    "label_en": "",  
    "label_loc": "1 机组温度",  
    "source": "API",  
    "type": "Float",  
    "unit": ""  
  },  
  {  
    "args": [],  
    "key": "2#Temp",  
    "label_en": "",  
    "label_loc": "2 机组温度",  
    "source": "API",  
    "type": "Float",  
    "unit": ""  
  }  
],  
"created": "2021-04-27T15:28:25+08:00",
```

```

        "last_modified": "",
        "last_modified_by": 0,
        "is_available": 1
    }
}
    
```

2.4.5.3 创建离线项目

创建离线项目

参数

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	create_project	方法名
actionid	否	String		活动标识
name	是	String		项目名称
description	否	String		项目描述
script	是	List		项目脚本

示例

```

{
    "action": "create_project",
    "name": "离线采集 1",
    "description": "离线采集 1",
    "script": [
        {
            "key": "length", // 离线 key
            "label_en": "Label_En", // 英文标签
            "label_loc": "长度", // 中文标签
            "type": "Integer", // 类型, 可选值有 Integer Float String Bool
        }
    ]
}
    
```

```

        "unit": "米", // 单位
        "source": "API", // 数据源
        "args": {} // 数据源具体参数
    }
]
}
响应
{
    "resp": "create_project",
    "actionid": "",
    "code": 200,
    "msg": "成功",
    "result": {
        "uid": "COL1472557300005"
    }
}

```

2.4.5.4 更新离线项目

更新离线项目

参数

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	update_project	方法名
actionid	否	String		活动标识
uid	是	String		项目 uid
name	否	String		项目名称
description	否	String		项目描述

参数名称	是否必选	数据类型	数据约束	描述
script	否	List		项目脚本

示例

```

{
  "action": "update_project",
  "uid": "COL1",
  "name": "edit",
  "description": "edit",
  "script": [
    {
      "key": "length",
      "label_en": "Label_En",
      "label_loc": "修改标签",
      "type": "int",
      "unit": "米",
      "source": "API",
      "args": {}
    }
  ]
}
    
```

响应

```

{
  "resp": "update_project",
  "actionid": "",
  "code": 200,
  "msg": "成功",
  "result": {
    "uid": "COL1472557300005"
  }
}
    
```

```
}

```

2.4.5.5 删除离线项目

删除离线项目

参数

参数名称	是否必选	数据类型	数据约束	描述
------	------	------	------	----

action	是	String	delete_project	方法名
actionid	否	String		活动标识
uid	是	String		项目 uid

示例

```
{
  "action": "delete_project",
  "uid": "COL1472557300005"
}
```

响应

```
{
  "resp": "delete_project",
  "actionid": "",
  "code": 200,
  "msg": "成功",
  "result": {
    "uid": "COL1472557300005"
  }
}
```

2.4.5.6 启动离线项目

启动离线项目

参数

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	start_project	方法名
actionid	否	String		活动标识
uid	是	String		项目 uid

示例

```
{
  "action": "start_project",
  "uid": "COL1472557300005"
}
```

响应

```
{
  "resp": "start_project",
  "actionid": "",
  "code": 200,
  "msg": "成功",
  "result": {
    "uid": "COL1472557300005"
  }
}
```

2.4.5.7 停止离线项目

停止离线项目

参数

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	stop_project	方法名

参数名称	是否必选	数据类型	数据约束	描述
actionid	否	String		活动标识
uid	是	String		项目 uid

示例

```
{
  "action": "stop_project",
  "uid": "COL1"
}
```

响应

```
{
  "resp": "stop_project",
  "actionid": "",
  "code": 200,
  "msg": "成功",
  "result": {
    "uid": "COL1472557300005"
  }
}
```

2.4.5.8 添加离线数据

添加上报一条离线数据，离线 ID 对应的项目需要启动并且上报的 value 中的 key 与对应项目配置的脚本匹配

参数

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	add_collect_value	方法名
actionid	否	String		活动标识

参数名称	是否必选	数据类型	数据约束	描述
name	是	String		名称
description	否	String		描述
collect_id	是	String		离线项目 id
object_id	是	String		对象 id
value	是	Object		数据 kv 对
file	否	String		文件 url
collector	否	String		上传者
reference	否	String		描述

示例

```

{
  "action": "add_collect_value",
  "actionid": "",
  "name": "数据采集 1",
  "description": "",
  "collect_id": "COL1310231900002",
  "object_id": "OBJ1243475500001",
  "value": {
    "temp": 88.5,
    "pressure": 202
  },
  "file": "",
  "collector": "mixuser",
  "reference": ""
}
    
```

响应

```

{
  "resp": "add_collect_value",
  "code": 200,
  "msg": "成功",
  "result": {
    "collect_id": "COL1310231900002",
    "name": "数据采集 1",
    "object_id": "OBJ1243475500001",
    "value": {
      "temp": 88.5,
      "pressure": 202,
      "object_id": "OBJ1243475500001"
    }
  }
}
    
```

2.4.5.9 获取最新离线数据

根据离线 ID 和对象 ID 获取最新一条 API 类型手工上报的离线数据

参数

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_last_collect_value	方法名
actionid	否	String		活动标识
collect_id	是	String		离线项目 id
object_id	是	String		对象 id

示例

```

{
  "action": "get_last_collect_value",
    
```

```
"collect_id": "COL1405447400006",  
"object_id": "OBJ1403630800003"  
}
```

响应

```
{  
  "resp": "get_last_collect_value",  
  "actionid": "",  
  "code": 200,  
  "msg": "成功",  
  "result": {  
    "id": 95,  
    "uid": "COLVAL1432906700001",  
    "name": "a",  
    "description": "",  
    "collect_id": "COL1405447400006",  
    "object_id": "OBJ1403630800003",  
    "value": {  
      "aa": true,  
      "aa2": 12313  
    },  
    "file": "",  
    "collector": "admin",  
    "collect_at": "2021-04-23 15:56:29",  
    "reference": "",  
    "created": "2021-04-23 15:56:29",  
    "is_available": 1  
  }  
}
```

2.4.5.10 获取离线值历史列表

获取历史的离线值

参数

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_collect_value_list	方法名
actionid	否	String		活动标识
page_index	否	int		页数，不传则获取所有记录
page_size	否	int		页大小，不传则获取所有记录
where_and	否	String		and 条件
where_or	否	String		or 条件
where_in	否	String		in 条件

示例

```
{
  "action": "get_collect_value_list",
  "page_index": 1,
  "page_size": 20,
  "where_and": "",
  "where_or": "",
  "where_in": "",
  "order_by": ""
}
```

响应

```
{
  "resp": "get_collect_value_list",
  "actionid": ""
}
```

```
"code": 200,
"msg": "成功",
"result": {
  "page_index": 1,
  "page_size": 20,
  "total_pages": 0,
  "total_records": 1,
  "data": [
    {
      "id": 95,
      "uid": "COLVAL1432906700001",
      "name": "a",
      "description": "",
      "collect_id": "COL1405447400006",
      "object_id": "OBJ1403630800003",
      "value": {
        "aa": true,
        "aa2": 12313
      },
      "file": "",
      "collector": "admin",
      "collect_at": "2021-04-23 15:56:29",
      "reference": "",
      "created": "2021-04-23 15:56:29",
      "is_available": 1
    }
  ]
}
```

2.4.5.11 UID 获取指定离线值

根据离线值 ID 获取单条离线值记录

参数

参数名称	是否必选	数据类型	数据约束	描述
------	------	------	------	----

action	是	String	get_collect_value	方法名
--------	---	--------	-------------------	-----

actionid	否	String		活动标识
----------	---	--------	--	------

uid	是	String		离线值 uid
-----	---	--------	--	---------

示例

```
{
  "action": "get_collect_value",
  "uid": "COLVAL1432906700001"
}
```

响应

```
{
  "resp": "get_collect_value",
  "actionid": "",
  "code": 200,
  "msg": "成功",
  "result": {
    "id": 95,
    "uid": "COLVAL1432906700001",
    "name": "a",
    "description": "",
    "collect_id": "COL1405447400006",
    "object_id": "OBJ1403630800003",
    "value": {
      "aa": true,

```



```

        "aa2": 12313
    },
    "file": "",
    "collector": "admin",
    "collect_at": "2021-04-23 15:56:29",
    "reference": "",
    "created": "2021-04-23 15:56:29",
    "is_available": 1
    }
}
    
```

2.4.6 Collect event 说明

离线值更新事件

```

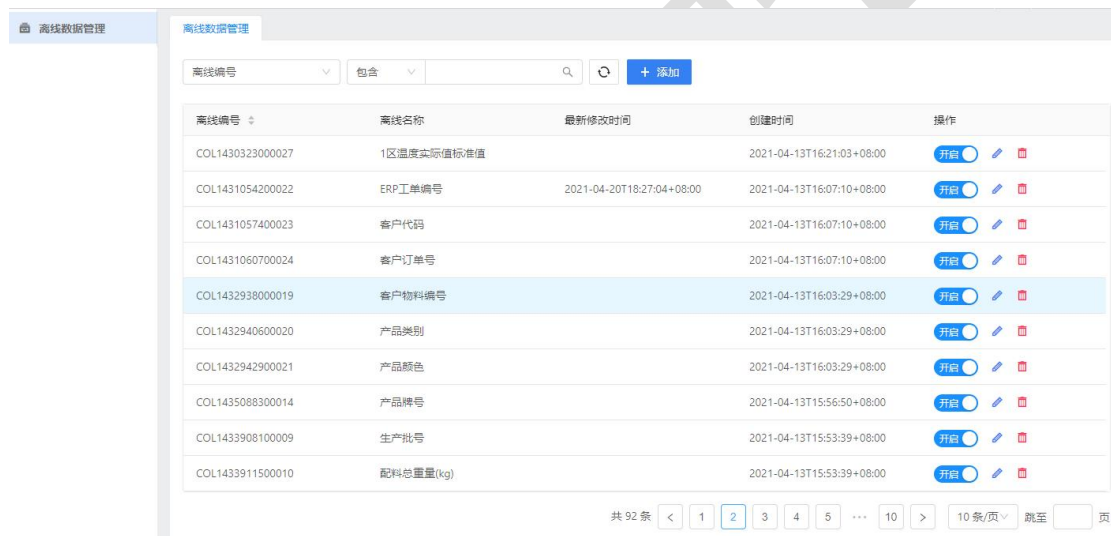
{
  "event": {
    "event": "collect_add_value_event",
    "block": "collect",
    "uniqueid": "COLEVT1",
    "datetime": "2006-01-02 15:04:05",
    "collect_id": "COLVAL1303892400001",
    "object_id": "OBJ1243475500001",
    "name": "test",
    "value": {"Tem":32,"num2":64},
  }
}
    
```

2.4.7 Collect 应用界面使用

点击离线数据应用图标进入离线应用界面



离线项目界面管理包含所有已配置的离线项目，可以增加新的离线项目，删除修改已有离线项目，以及启动或停止离线项目。



添加或修改离线项目，可以配置离线项目包含的 key

添加离线数据
✕

*** 离线数据名称:**

子采集项名称:

参数KEY	中文名	英文名	参数单位	数据类型	数据来源	
my_voltage	电压	voltage	伏特	浮点型	API录入	✕
my_current	电流	current	安	浮点型	API录入	✕

+ 添加子采集项

描述:

取消 确认

配置完成后启动项目，即可在对象管理页面进行采集录入。将离线项目关联到一个具体的对象后，可以在对象的离线数据选项中找到所有关联的离线项。

对象管理
TESTq46812(OBJ1403630800003) ✕

设备监控
基本信息
FV列表
离线数据
适配器列表
子对象列表
事件列表
历程列表

录入采集
采集记录

包含

🔍
🔄

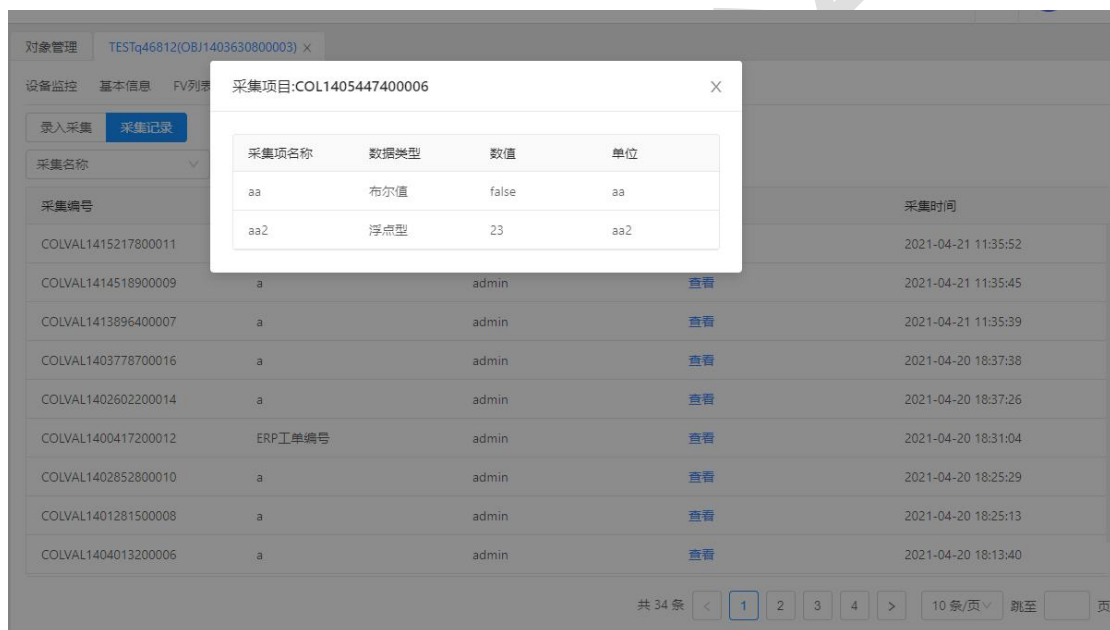
采集编号	采集名称	最新采集时间	创建时间	操作
COL1405447400006	a	2021-04-21T09:30:45+08:00	2021-04-20T17:25:54+08:00	采集
COL1434233900030	2区温度实际值下界值		2021-04-13T16:22:42+08:00	采集
COL1431054200022	ERP工单编号	2021-04-20T18:27:04+08:00	2021-04-13T16:07:10+08:00	采集

共 3 条
< 1 >
10 条/页

点击离线项目的采集按钮，即可开始进行一次离线数据采集。采集项即在离线项目配置的所有 key，在对应的数值输入框输入采集值，点击确认完成采集录入。



采集记录页面可以查看历史的采集记录列表，点击查看可以查看历史采集上报的数据。



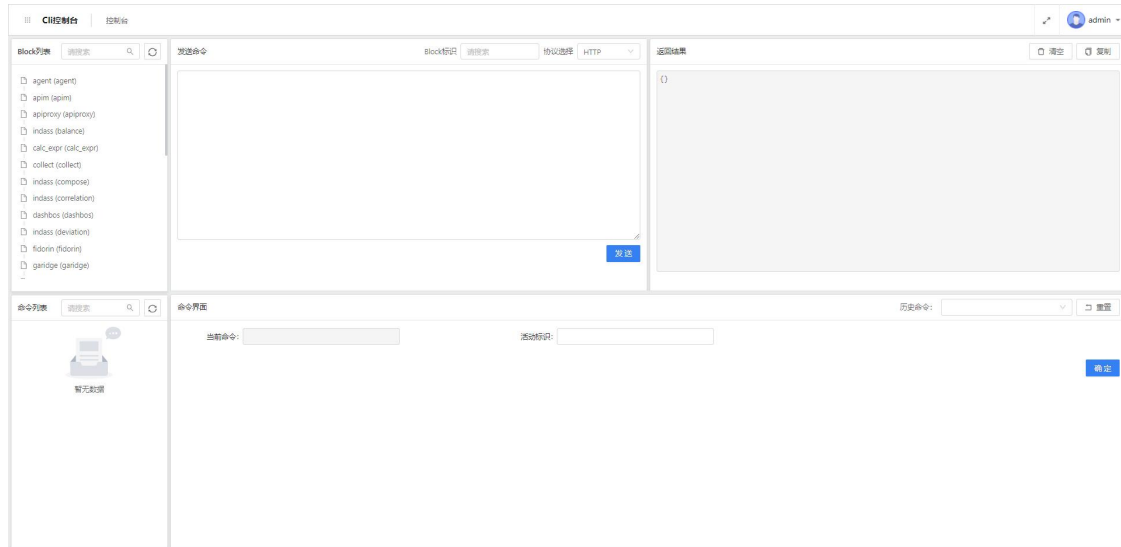
2.5 CLI 控制台-CLI

cli 客户端 (Command Line Interface, 命令行界面) 是 Mixiot 平台一个 Block 调试 (Debug) 应用，可以在这个应用中，访问所有注册在 consul 的服务的接口，主要给调试、开发本平台的技术人员使用。

从界面上来看，分为 Block 列表、命令列表、命令页面、发送命令和返回结果 5 部分。

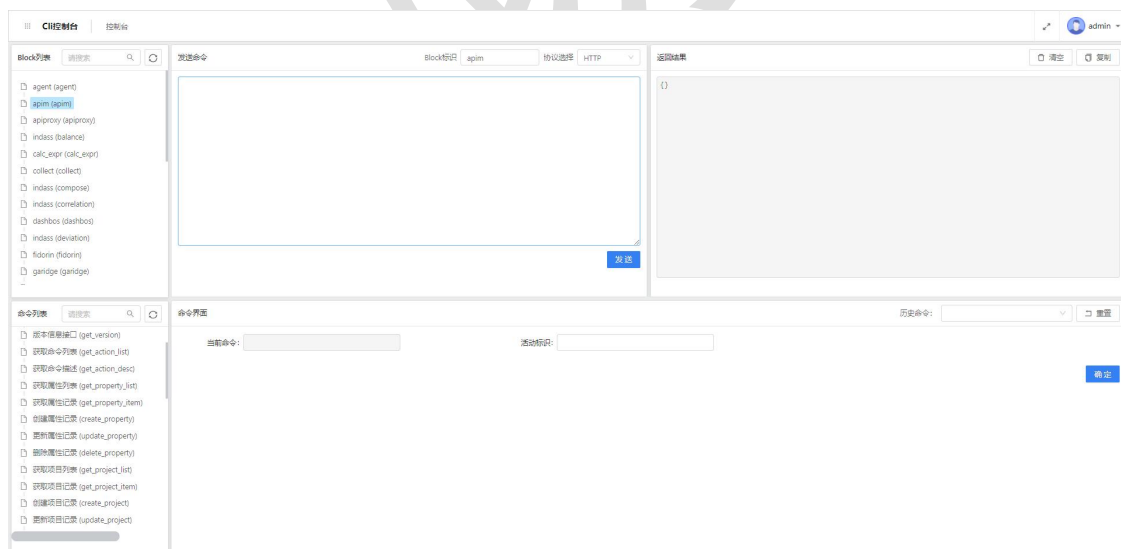
2.5.1 Block 列表

Block 列表就是本平台所包含的 Block，也可以理解为本平台内包含的应用列表。在这里可以指定要请求的接口的 block，假如选定了 apim，则会自动显示 apim 的所有命令列表。



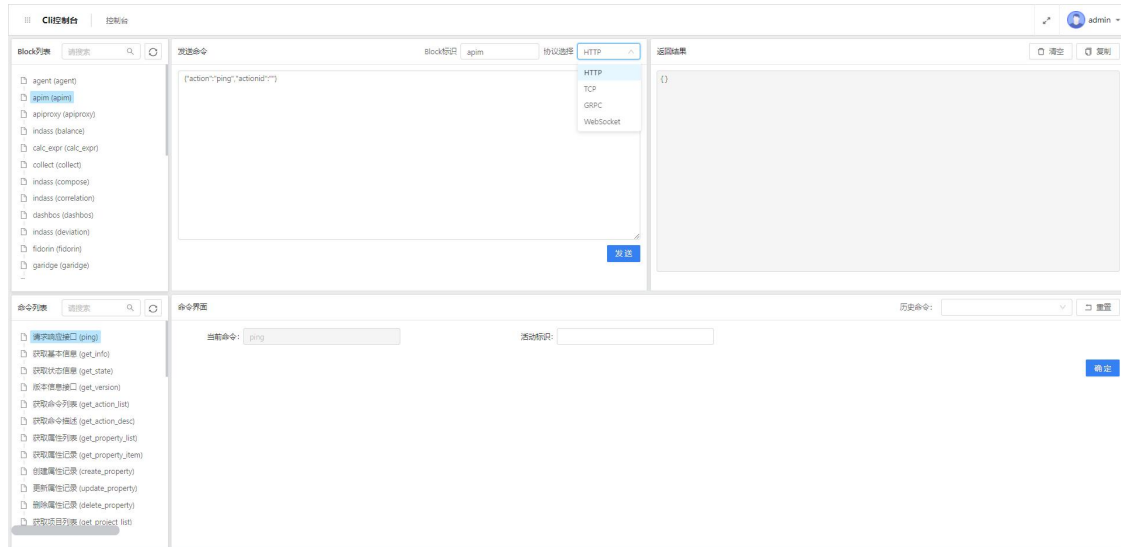
2.5.2 命令列表

命令列表就是选定应用所提供的所有接口，也叫命令组成的列表。这些命令的详细用法将在各个 Block 的使用说明中详细介绍，本说明不多讲解。指定了某一个命令之后，就可以进行下一步发送命令。



2.5.3 发送命令

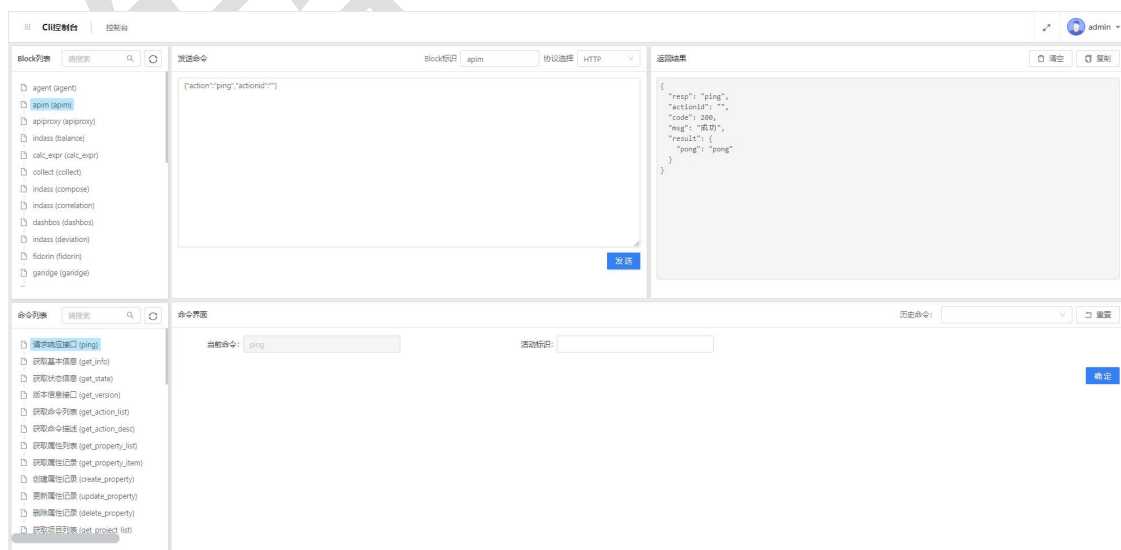
比如我们选择第一个命令，请求响应接口，含义就是测试这个应用的接口是否能够响应。



这个命令无需其余参数，直接点击发送即可。同时，还可以选择使用什么协议发送本命令，目前支持 4 种格式，分别是 HTTP、TCP、GRPC 和 WEBSOCKET。协议的具体区分本说明也不赘述，各位使用者可以查询相关资料。确定发送命令后，点击发送按钮，如果 Block 工作正常，则可以收到相应的结果。

2.5.4 返回结果

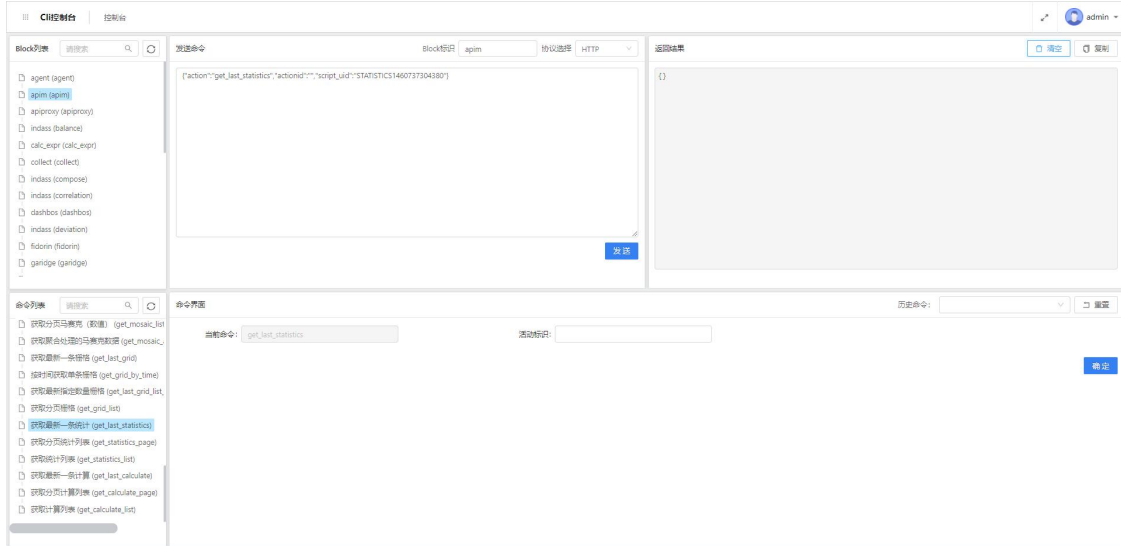
返回结果就显示了命令返回的内容，结果使用方式也将在各个 block 的使用说明中描述，或者按照实际需求使用。



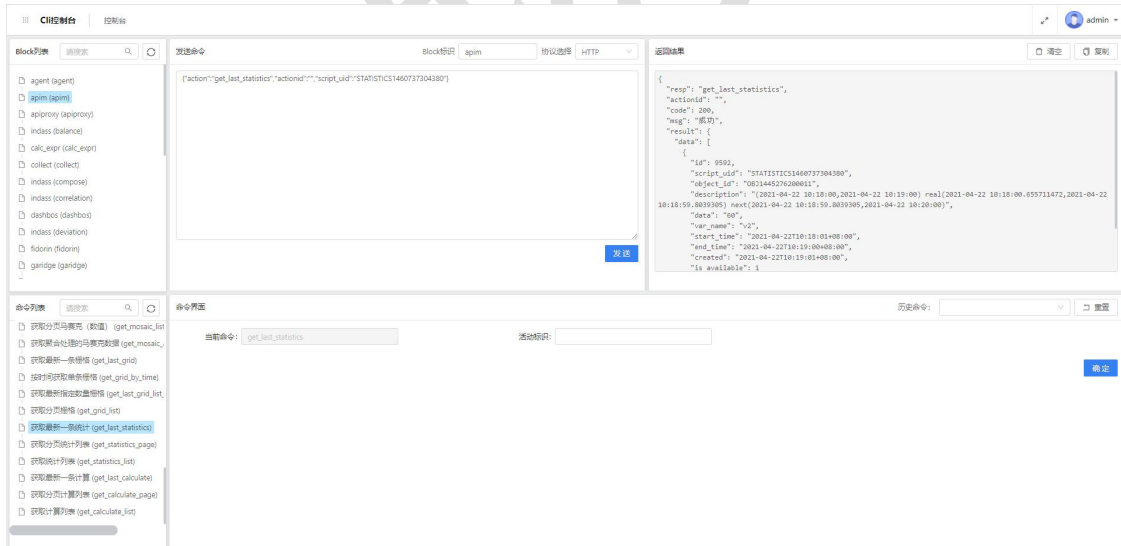
2.5.5 命令界面

对于一些复杂的命令，需要一个或者多个参数的，则可以使用命令界面填写参数。

图待补充



也可以不使用命令列表，直接在发送命令中填写相应参数，也可以取到相应结果。



2.6 门户中心-Portal

2.6.1 登录方法

第一次进入 MIXIOT，以及后期登录时效性过期后，又回到这个界面进行登录。



配置一：填写登录账号

配置二：输入登录密码

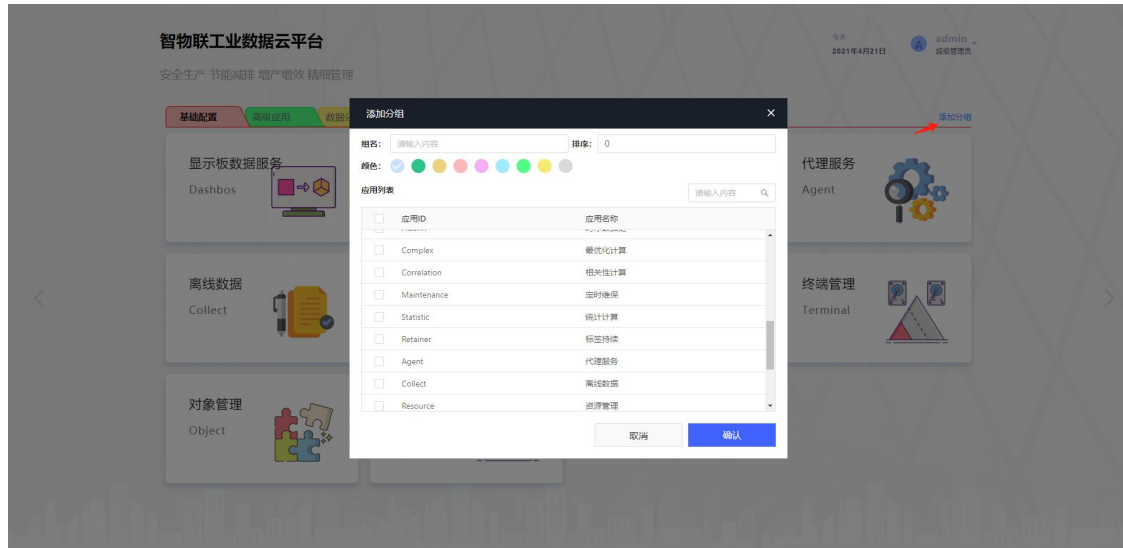
配置三：拖动滑块验证，并点击登录，即可登录

登录成功后，即可进入主页。



2.6.2 添加分组

主页默认仅有一个全部分组，另外可以建立、管理分组，按照实际需求管理分页。



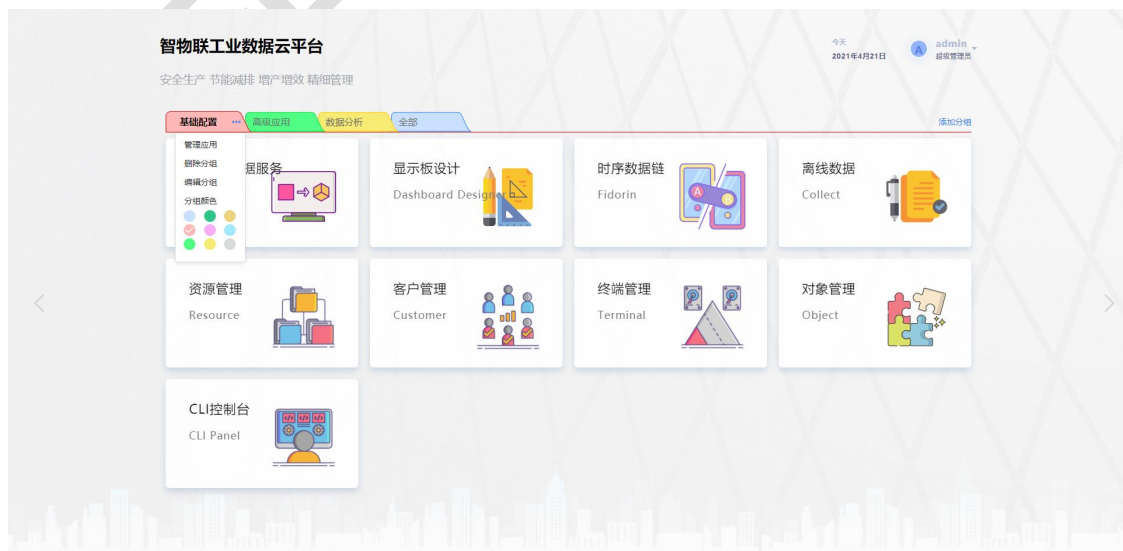
配置一：点击添加分组

配置二：按照实际需要填写组名

配置三：按照实际需要设置排序、颜色

配置四：按照实际需求选择需要归为一组的应用

配置五：点击确认即可保存



对于已有的分组，也可以进行编辑、管理。

2.7 门户设置-Portal Setting

在门户设置界面，可以设计平台的相关标语、图片等，下图为各个设置项对应的位置：

2.7.1 主页设置



2.7.2 登录页设置

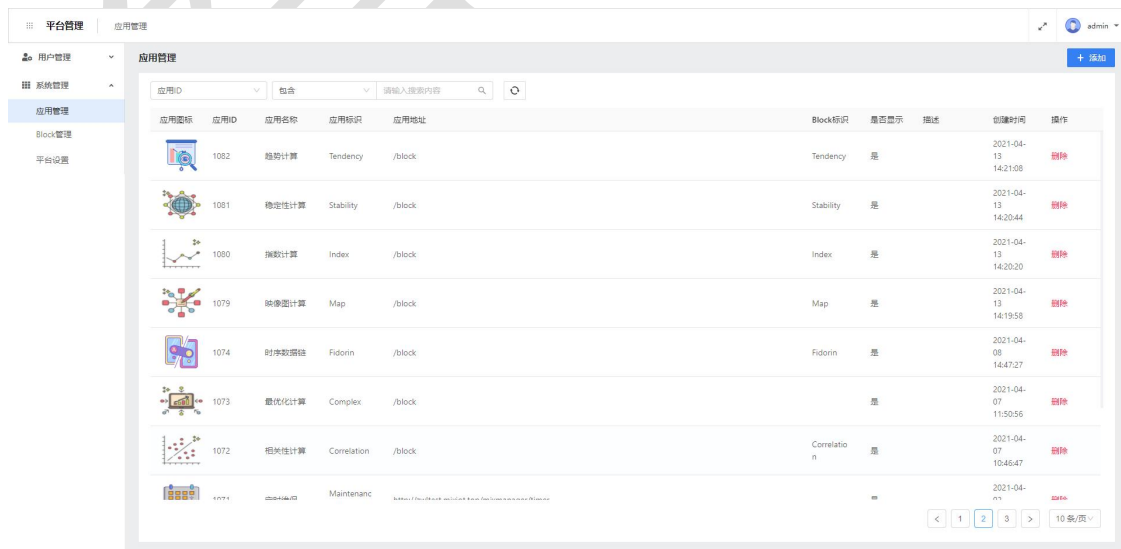


应用尺寸：主页中每个应用块的大小

保持登录时间：每次登录能够保持的时间，默认是 2 小时，即 2 小时候，需要重新登录

2.8 应用中心-Shortcut

2.8.1 应用管理



应用管理是管理开发应用的前段跳转网址的地方。如果要新增应用模块，则点击增加按钮：



增加应用，需要填写以下信息：

应用标识：建议填写英文，为界面上显示的英文名

应用名称：建议填写中文，为界面上显示的中文名

应用地址：这里有 2 种填写方式，如果开发的应用属于 block 形式，那么在应用地址处就填写/block。如果开发了其他的界面，那么就按照实际情况填写网址。

Block：如果服务开发的形式符合 block 形式，则在此处选择对应的 block，如果没有，可以不填写

排序：在此处可以决定该应用在主页所有分页中的排序

应用类别：应用类别的含义分别是：

管理类：数据管理类，例如对象管理、适配器管理等

结果类：结果展示类，例如大屏、设备看板、数据看板等

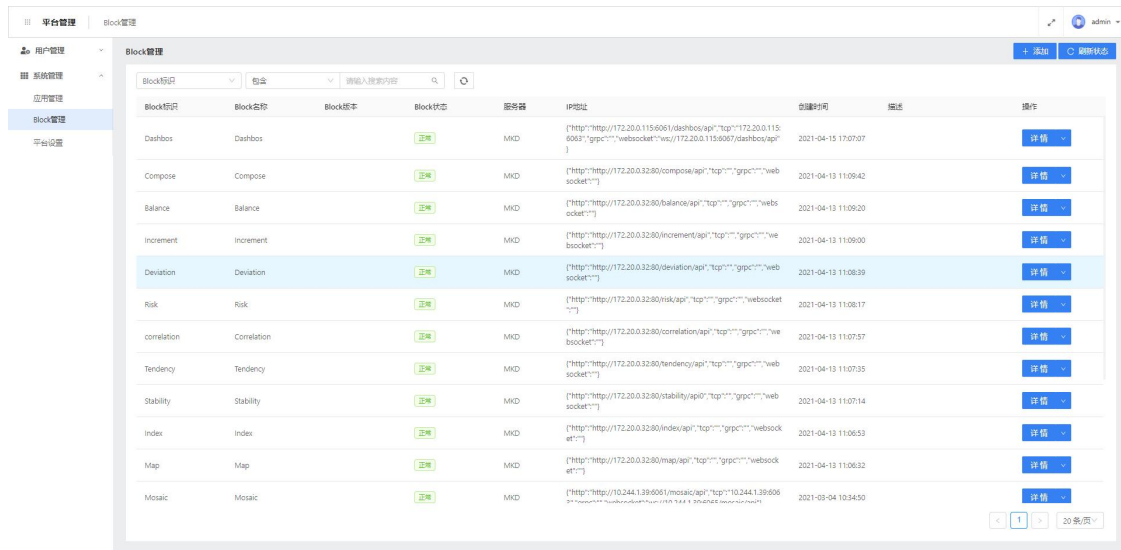
应用类：项目应用类，例如维保、工单等

是否显示：可以决定是否要在主页中显示出来

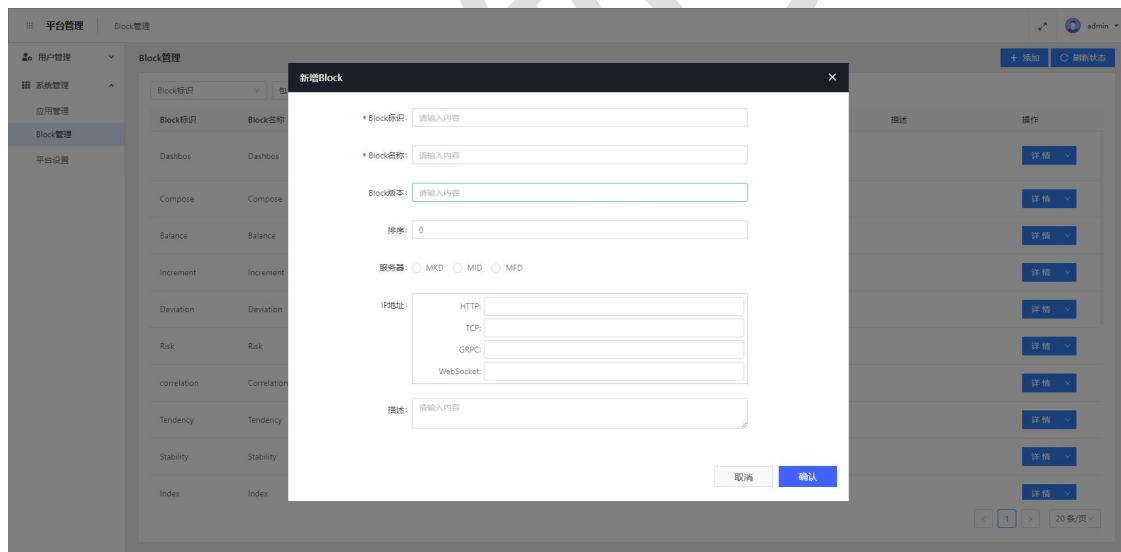
2.8.2 Block 管理

在当前的平台中，有一种 block 开发模式，简单来说，就是通过脚本的形式，直接生成项目配置界面，无需重新开发项目配置界面。具体的使用方式参考《Block 设计详解》，在此处

不做展开，我们这里只介绍如何把已开发的 block 产品放在主页中显示。



在这个界面可以管理 block，点击详情可以查看，下拉后还可以进行安装、卸载、删除操作。点击添加后，就可以新增一个以 block 形式开发的应用。



Block 标识： 建议填写英文，是本应用的唯一标识

Block 名称： 建议填写中文，是本应用的中文名称

Block 版本： 为本应用的版本号

服务器： 有三个选项

MKD

MID

MFD

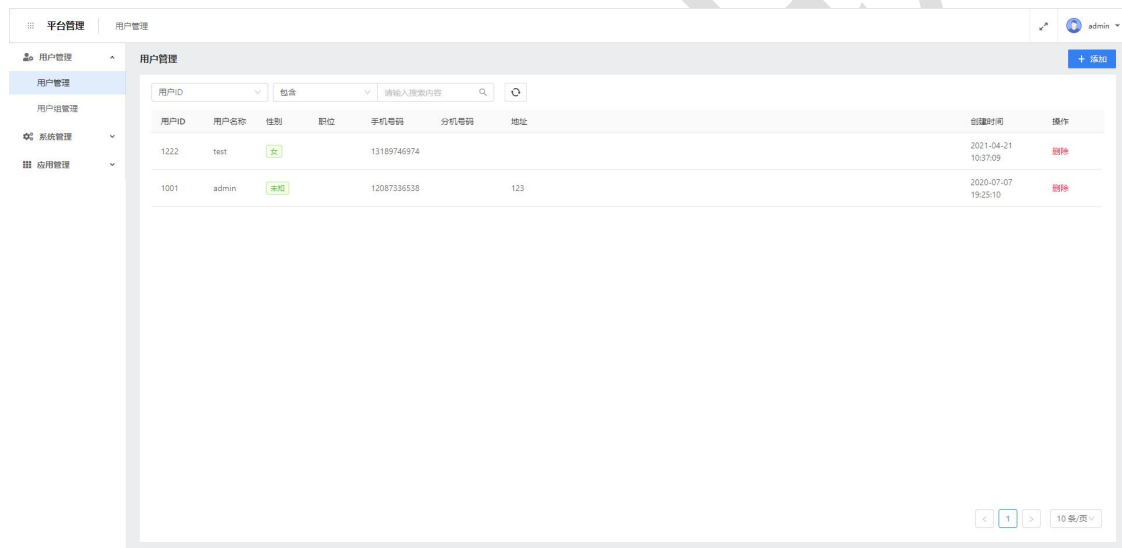
IP 地址：本平台可以提供 4 中形式的传输协议，按照相应的协议填写相应网址

描述：可根据实际情况填写

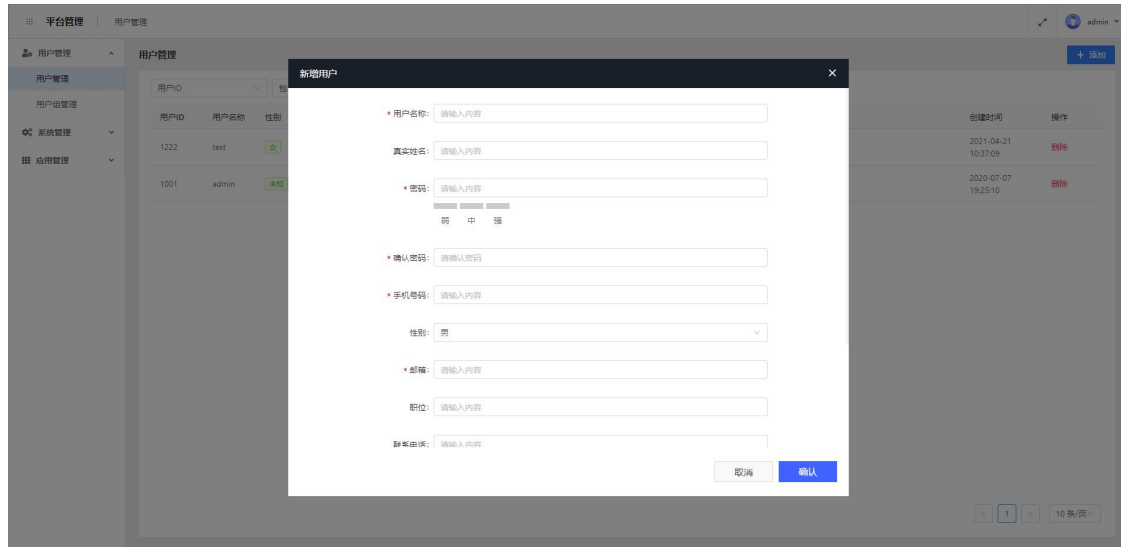
建立好 block 信息后，可在应用管理中进行绑定。

2.9 用户中心-User

2.9.1 用户管理

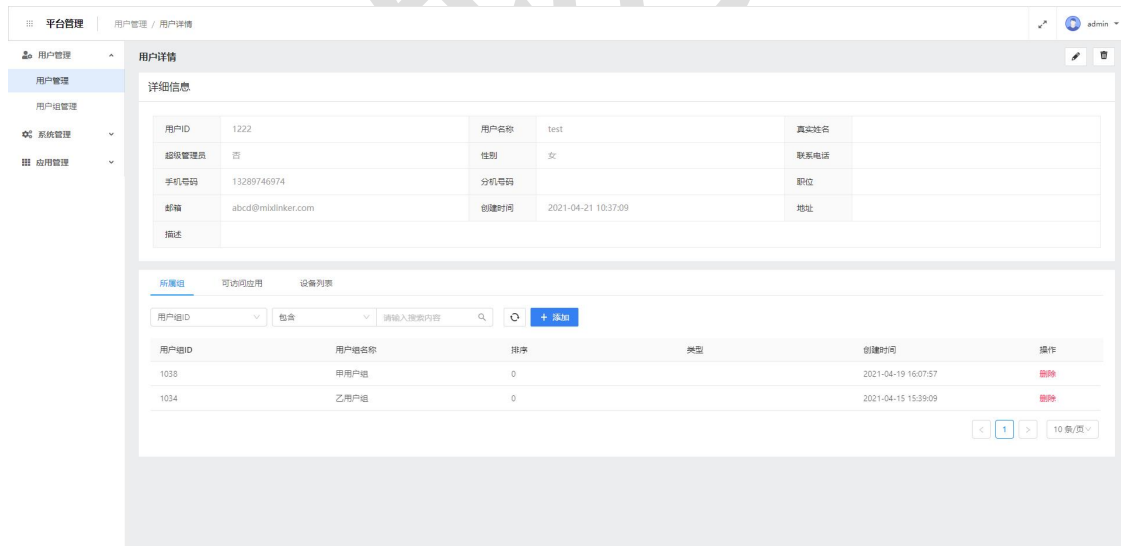


这里可以看到本平台的所有用户，也可以根据用户 ID 和用户名称等条件进行筛选。



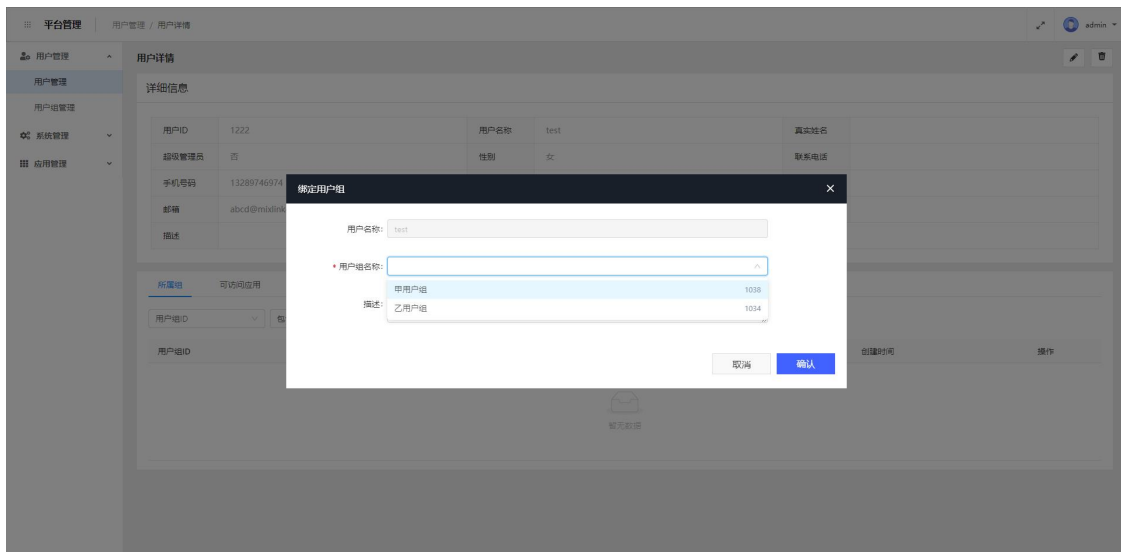
用户管理可以管理所有使用这个平台的用户，本平台没有注册用户的功能，如果要增加用户，需要在用户管管理界面点击添加按钮，按照实际情况填写相关信息，点击保存即可新增一位用户。

配置：其中用户名称、密码、确认密码、手机号码、邮箱是必填项
添加好了之后，点击某个指定的用户，可以进去看到用户的详情。



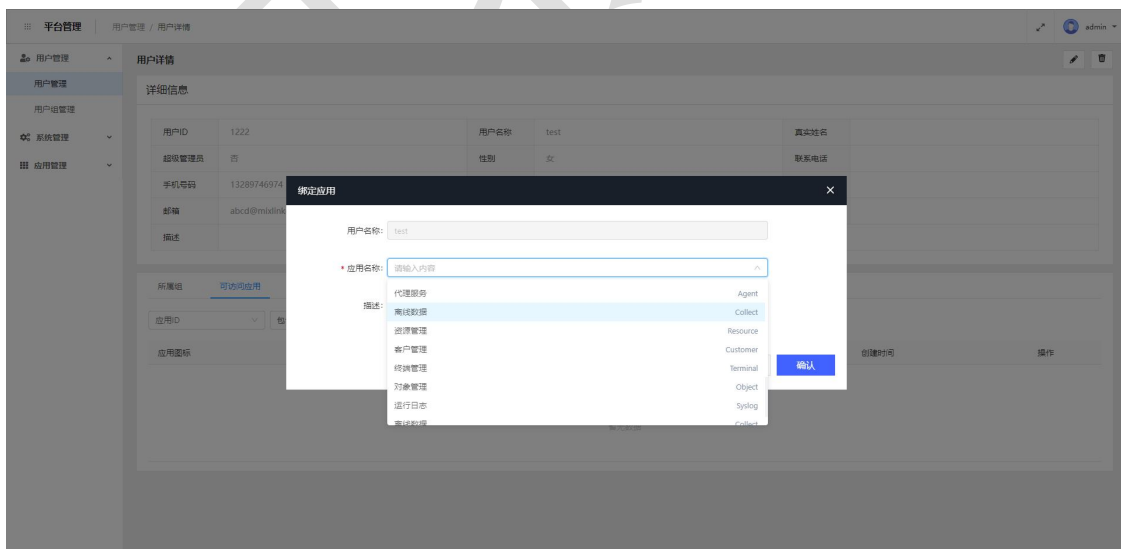
除了基本信息外，还可以管理用户的权限，比如本用户所属的应用组，本用户可访问的应用，本用户所能看的设备列表。

2.9.1.1 绑定用户组



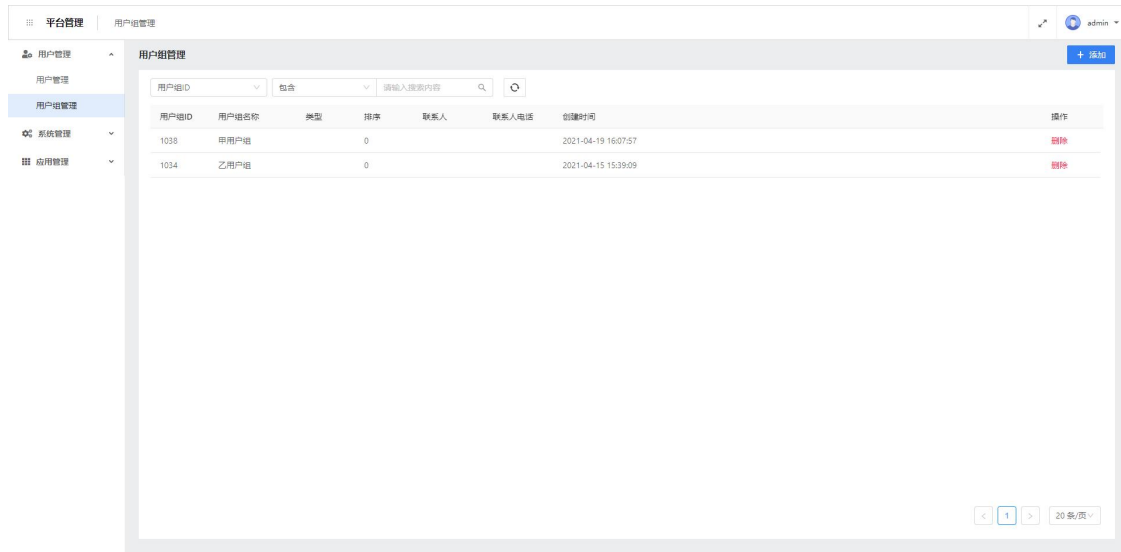
点击添加用户组后，这个用户就拥有了跟用户组一样的权限，如果选择了多个用户组，这个用户的权限就是所有用户组的并集，拥有这些用户组所拥有的应用、设备权限。

2.9.1.2 绑定应用

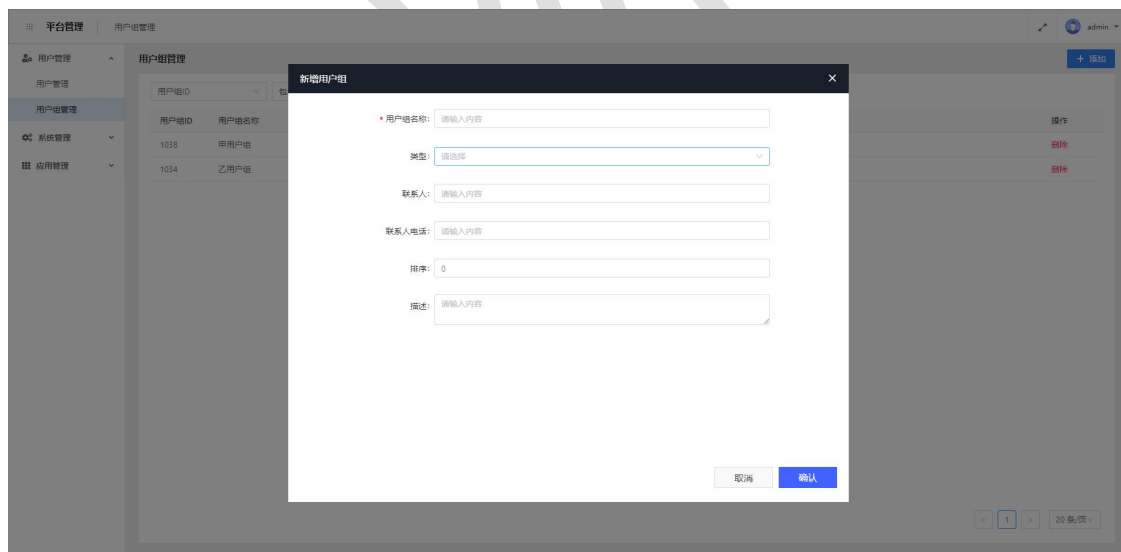


在可访问应用中点击添加按钮，可以添加已有应用的使用权限。设备列表也是相同的情况，在设备列表中添加对应设备权限，即可有该设备的查看权限。

2.9.2 用户组管理

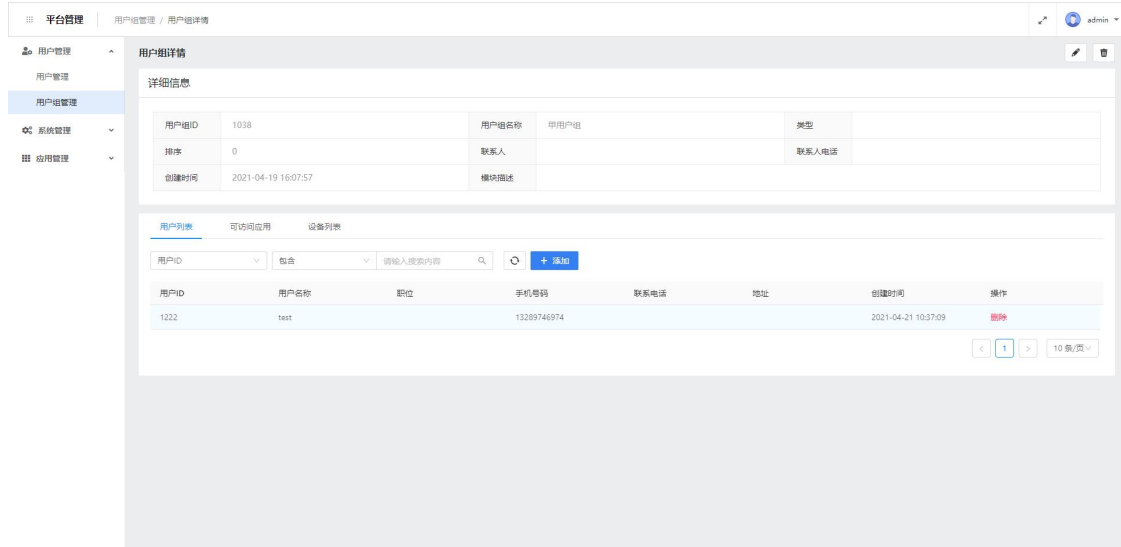


在用户组管理，可以查看现有的用户组，可以进行增加、删除、筛选，点击用户组，可以查看用户组的详情。点击添加后，也可以按照需要增加用户组。



配置：其中用户组名称是必填项

按照和用户相同的方法，可以管理用户组的用户、可访问应用、设备列表。



2.10 统计计算-Statistics

2.10.1 脚本含义说明

2.10.1.1 统计

- name:指定统计项目名称
- func:指定统计方法，就是对设备的某个 fv 要进行何种统计,支持的统计函数有
 - max:最大
 - min:最小
 - count:计数
 - sum:求和
 - avg:平均（算术平均）
 - last:最新值(周期内最后一个值)

- first:最旧值(周期内最早一个值)
- integral:积分
- median:中位数
- spread:极差(最大 - 最小)
- distinct:去重后数量(枚举值个数)
- hour:耗时统计
- accumulate:读数转用量
- stddev:标准差
- object_id:要统计的设备 Id
- object_list:要统计的设备列表
- key:指定要统计的设备的 FV
- schedule
 - spec:统计周期，指定多久生成一次结果，周期不可小于一分钟

spec详解

beego中的spec设计参考Linux中cron的配置。其中前6个字段依次分别表示：秒钟：0-59、分钟：0-59、小时：1-23、日期：1-31、月份：1-12、星期：0-6（0表示周日）。下面是一些特殊符号以及简单示例。

符号	含义	示例
*	表示任何时刻	
,	表示分割	如第三段里：2,4，表示 2 点和 4 点执行
-	表示一个段	如第三段里：1-5，就表示 1 到 5 点
/n	表示每个n的单位执行一次	如第三段里，*/1，就表示每隔 1 个小时执行一次命令。也可以写成1-23/1

示例	详细含义
0/30 * * * * *	每 30 秒 执行
0 43 21 * * *	21:43 执行
0 0 17 * * 1	每周一的 17:00 执行
0 0,10 17 * * 0,2,3	每周日,周二,周三的 17:00和 17:10 执行
0 0 21 * * 1-6	周一到周六 21:00 执行
0 */10 * * * *	每隔 10 分 执行

- result
 - precision:指定结果输出的精度
 - var_name:指定结果的 key, 不填写则 Statistics Block 自动生成~~
- filter:指定统计项目引用的过滤模型
- condition:指定统计项目引用的条件模型
- func_params:函数参数设置
 - hour_params:hour 函数参数设置
 - threshold:阈值设置

使用数学中集合的概念表示区间, $(,)$ 表示区间的范围 $()$ 表示开区间, $[,]$ 表示闭区间

1. 退化区间(单值): $[a] = \{a\}$
2. 闭区间: $[a,b] = \{a \leq x \leq b\}$
3. 开区间: $(a,b) = \{a < x < b\}$
4. 左闭右开区间: $[a,b) = \{a \leq x < b\}$
5. 左开右闭区间: $(a,b] = \{a < x \leq b\}$
6. 单侧无界,有下界但无上界,左闭: $[a,)$ 或者 $[a,] = \{x \geq a\}$
7. 单侧无界,有下界但无上界,左开: $(a,)$ 或者 $(a,] = \{x > a\}$
8. 单侧无界,有上界但无下界,右闭: $(,b]$ 或者 $[,b] = \{x \leq b\}$
9. 单侧无界,有上界但无下界,右开: $(,b)$ 或者 $[,b) = \{x < b\}$
10. 双侧无界:不存在这种写法,hour 函数必须配置 threshold

* ``unit``:hour 函数结果的单位设置

1. 可填写值: ``ms``(毫秒), ``s``(秒), ``m``(分), ``h``(小时), ``d``(天)
2. 默认值: ``s``(秒)

2.10.1.2 计算

- `name`:指定计算项目的名称
- `expr`:指定计算项目使用的表达式
 - 支持使用的 `+ - * / ()`
 - 指定的变量必须在 `statis_var_list` 或者 `obj_var_list` 指定
- `triggert`:结果由统计脚本触发,这里填写统计项目的 UID,支持填写多个
- `statis_var_list`:指定用于给 `expr` 引用的统计结果的 `var_name` 列表
- `obj_var_list`:指定用于给 `logic_expr` 引用的对象 FV 列表
- `result`
 - `precision`:指定结果输出的精度
 - `var_name`:指定结果的 `key`,不填写则 `Statistics Block` 自动生成

2.10.1.3 条件模型

- `logic_expr`:必须填写逻辑表达式
 - 支持使用的 `+ - * / () && || & | == !=`
 - 指定的变量必须在 `obj_var_list` 指定
- `obj_var_list`:指定用于给 `logic_expr` 引用的对象 FV 列表

- default:指定默认值（若 logic_expr 计算错误，使用 default 的值。true 为保留，false 为不保留）

2.10.1.4 过滤模型

过滤模型目前仅在函数 accumulate 会使用

- first_min:阈值过滤最小值
- first_max:阈值过滤最大值
- sigma:过滤系数
- mean_factor:规划阈值过滤

2.10.2 支持 ACTION 列表

2.10.2.1 创建统计/计算

2.10.2.1.1 创建统计

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	create_statistics_task	方法名
actionid	否	String		活动标识
name	是	String		统计名称
func	是	String		统计方法
object_id	是	String		统计设备
object_list	是	List		要统计的设备 Id 列表

参数名称	是否必选	数据类型	数据约束	描述
key	是	String		统计 FV
filter	否	String		过滤 Id
result	否	Json		统计结果
+precision	否	float64		结果精度
+var_name	否	String	不填，自动生成	结果变量 key
schedule	是	Json		统计周期
+spec	是	String		指定多久生成一次结果
object_list	是	列表		对象列表

- 示例

```

{
  "action": "create_statistics_task",
  "key": "v1",
  "func": "count",
  "name": "123",
  "filter": "sff",
  "result": {
    "precision": 4
  },
  "schedule": {
    "spec": "0 */1 * * * *"
  },
  "object_list": ["OBJ1243475500002","OBJ1243475500001"]
}
    
```

- 返回

- **uid**: uid 唯一标识一个统计项目
- **var_name:var_name** 唯一表示一个统计项目的结果

```

{
  "resp": "create_statistics_task",
  "code": 200,
  "msg": "成功",
  "result": {
    "uid": "STATISTICS1395846800005"
  }
}
    
```

2.10.2.1.2 计算命题

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	create_calculate_task	方法名
actionid	否	String		活动标识
expr	是	String		用于计算的表达式
trigger	是	列表		指定触发计算的一个或者多个统计 uid
statis_var_list	否	列表		需要取出统计结果值做计算的统计 var_name 列表
object_id	否	String	和 mapping_id 同时只	需要取出的用于计

参数名称	是否必选	数据类型	数据约束	描述
			能存在一个	算的设备
obj_var_list	否	列表		需要取出的用于计算的设备变量 FV 列表
mapping_id	否	String	和 object_id 同时只能存在一个	需要取出的用于计算的用同一个 mapping_id 的多个设备
spec	是	String		指定多久生成一次结果
result	否	Json		结果
+ precision	否	float64		结果精度
+ var_name	否	String		结果变量 key

- 示例

```

{
  "action": "create_calculate_task",
  "expr": "STATISTICS1391005100002",
  "trigger": ["STATISTICS1391005100002"],
  "result": {
    "precision": 3
  }
}
    
```

- 返回

- uid: uid 唯一标识一个统计项目

- `var_name:var_name` 唯一表示一个统计项目的结果

```
{
  "resp": "create_calculate_task",
  "code": 200,
  "msg": "成功",
  "result": {
    "uid": "CALCULATE1302391300002"
  }
}
```

2.10.2.2 启动统计/计算

2.10.2.2.1 启动统计

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	start_statistics_task	方法名
actionid	否	String		活动标识
uid	是	String		创建统计命题的时候生成的 uid

- 示例

```
{
  "action": "start_statistics_task",
  "uid": "STATISTICS1395587100001"
}
```

- 返回

```

{
  "resp": "start_statistics_task",
  "code": 200,
  "msg": "成功",
  "result": {
    "next": "2021-03-30 09:59:54"
  }
}
    
```

2.10.2.2.2 启动计算

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	start_calculate_task	方法名
actionid	否	String		活动标识
uid	是	String		创建计算命题的时候生成的 uid

- 示例

```

{
  "action": "start_calculate_task",
  "uid": "CALCULATE1302391300002"
}
    
```

- 返回

```

{
  "resp": "start_calculate_task",
  "code": 200,
  "msg": "成功"
}
    
```

2.10.2.3 停止统计/计算

2.10.2.3.1 停止统计

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	stop_statistics_task	方法名
actionid	否	String		活动标识
uid	是	String		创建统计命题的时候生成的 uid

- 示例

```

{
  "action": "stop_statistics_task",
  "uid": "STATISTICS1341891700001"
}
    
```

- 返回

```

{
  "resp": "stop_statistics_task",
  "code": 200,
  "msg": "成功"
}
    
```

2.10.2.3.2 停止计算

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	stop_calculate_task	方法名
actionid	否	String		活动标识
uid	是	String		创建计算命题的时候生成的 uid

- 示例

```

{
  "action": "stop_calculate_task",
  "uid": "CALCULATE1381575800012"
}
    
```

- 返回

```

{
  "resp": "stop_calculate_task",
  "code": 200,
}
    
```

```

    "msg": "成功"
}
    
```

2.10.2.4 取消统计/计算

2.10.2.4.1 取消统计

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	cancel_statistics_task	方法名
actionid	否	String		活动标识
uid	是	String		创建统计命题的时候生成的 uid

- 示例

```

{
  "action": "cancel_statistics_task",
  "uid": "STATISTICS1391005100002"
}
    
```

- 返回

```

{
  "resp": "cancel_statistics_task",
  "code": 200,
}
    
```

```

        "msg": "成功"
    }

```

2.10.2.4.2 取消计算

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	cancel_calculate_task	方法名
actionid	否	String		活动标识
uid	是	String		创建统计命题的时候生成的 uid

- 示例

```

{
    "action": "cancel_calculate_task",
    "uid": "CALCULATE1302904900002"
}

```

- 返回

```

{
    "resp": "cancel_calculate_task",
    "code": 200,
    "msg": "成功"
}

```

2.10.2.5 获取统计/计算项目列表

2.10.2.5.1 获取统计项目列表

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_statistics_task_list	方法名
actionid	否	String		活动标识

- 示例

```
{
  "action": "get_statistics_task_list"
}
```

- 返回

```
{
  "resp": "get_statistics_task_list",
  "code": 200,
  "msg": "成功",
  "result": [
    "STATISTICS1305433500001"
  ]
}
```


2.10.2.5.2 获取计算项目列表

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_calculate_task_list	方法名
actionid	否	String		活动标识

- 示例

```
{
  "action": "get_calculate_task_list"
}
```

- 返回

```
{
  "resp": "get_calculate_task_list",
  "code": 200,
  "msg": "成功",
  "result": [
    "CALCULATE1305256100003",
    "CALCULATE1305483400004",
    "CALCULATE1305595400005",
    "CALCULATE1305665000006"
  ]
}
```

2.10.2.6 获取统计/计算最新结果值(内存值)

2.10.2.6.1 获取统计最新结果值(内存值)

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_statistics_value	方法名
actionid	否	String		活动标识
uid	是	String		创建统计命题的时候生成的 uid

- 示例

```
{
  "action": "get_statistics_value",
  "uid": "STATISTICS1354093700001"
}
```

- 返回

```
{
  "resp": "get_statistics_value",
  "code": 200,
  "msg": "成功",
  "result": {
    "time_pair": {
      "start": "2021-03-30 14:12:22.001010625",
```

```

        "end": "2021-03-30 14:12:23.000961867"
    },
    "data": 538
}
}
    
```

2.10.2.6.2 获取计算最新结果值(内存值)（暂不提供）

2.10.2.7 获取统计/计算最新结果值(磁盘)

2.10.2.7.1 获取计算最新结果值(磁盘)

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_newest_statistics_value	方法名
actionid	否	String		活动标识
uid	是	String		创建计算命题的时候生成的 uid
object_id	是	String		创建计算命题的时候 object_list 中的对象

- 示例

```

{
  "action": "get_newest_statistics_value",
  "uid": "STATISTICS1584816000007",
  "object_id": "OBJ1570163200002"
}
    
```

- 返回

```
{
  "resp": "get_newest_statistics_value",
  "code": 200,
  "msg": "成功",
  "result": {
    "id": 668631,
    "name": "123",
    "script_uid": "STATISTICS1584816100008",
    "description": "(2021-05-18 15:04:00,2021-05-18 15:05:00) real(2021-05-18
15:04:03.017409582,2021-05-18 15:04:57.965702444) next(2021-05-18
15:04:57.965702444,2021-05-18 15:06:00)",
    "data": "10.0000",
    "object_id": "OBJ1570163200002",
    "mapping_id": "",
    "var_name": "RSTATISTICS1584816100008",
    "start_time": "2021-05-18T15:04:00+08:00",
    "end_time": "2021-05-18T15:05:00+08:00",
    "created": "2021-05-18T15:05:03+08:00",
    "is_available": 1
  }
}
```

2.10.2.7.2 获取计算任务最新值(磁盘)

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_newest_calculate_value	方法名
actionid	否	String		活动标识
uid	是	String		创建计算命题的时候生成的 uid
object_id	是	String		创建计算命题的时候 object_list 中的对象

- 示例

```
{
  "action": "get_newest_calculate_value",
  "uid": "CALCULATE1581540600028",
  "object_id": "OBJ1570163200002"
}
```

- 返回

```
{
  "resp": "get_newest_calculate_value",
  "code": 200,
  "msg": "成功",
  "result": {
    "id": 668631,
    "name": "123",
    "script_uid": "CALCULATE1581540600028",
    "description": "(2021-05-18 15:04:00,2021-05-18 15:05:00) real(2021-05-18 15:04:03.017409582,2021-05-18 15:04:57.965702444) next(2021-05-18 15:04:57.965702444,2021-05-18 15:06:00)",
    "data": "10.0000",
```

```

        "object_id": "OBJ1570163200002",
        "mapping_id": "",
        "var_name": "RCALCULATE1581540600028",
        "start_time": "2021-05-18T15:04:00+08:00",
        "end_time": "2021-05-18T15:05:00+08:00",
        "created": "2021-05-18T15:05:03+08:00",
        "is_available": 1
    }
}
    
```

2.10.2.8 获取统计/计算当前状态

当前状态只有 start, stop 两种值

2.10.2.8.1 获取统计状态

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_statistics_stat	方法名
actionid	否	String		活动标识
uid	是	String		创建统计命题的时候生成的uid

- 示例

```

{
    "action": "get_statistics_stat",
    
```

```
"uid": "STATISTICS1354093700001"
}
```

- 返回

```
{
  "resp": "get_statistics_stat",
  "code": 200,
  "msg": "成功",
  "result": "start"
}
```

2.10.2.8.2 获取计算状态

- 描述
- 格式

参数名称	是否可选	数据类型	数据约束	描述
action	是	String	get_calculate_stat	方法名
actionid	否	String		活动标识
uid	是	String		创建计算命题的时候生成的uid

- 示例

```
{
  "action": "get_calculate_stat",
  "uid": "CALCULATE1341392400154"
}
```

- 返回

```
{
  "resp": "get_calculate_stat",
  "code": 200,
  "msg": "成功",
  "result": "stop"
}
```

2.10.2.9 获取统计/计算脚本详情

2.10.2.9.1 获取统计脚本详情

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_statistics_task_detail	方法名
actionid	否	String		活动标识
uid	是	String		创建统计命题的时候生成的 uid

- 示例

```
{
  "action": "get_statistics_task_detail",
  "uid": "STATISTICS1303162000001"
}
```


- 返回

```

{
  "resp": "get_statistics_task_detail",
  "code": 200,
  "msg": "成功",
  "result": {
    "func": "accumulate",
    "key": "v1",
    "object_list": ["OBJ1343477200007"],
    "result": {
      "precision": 4,
      "var_name": "f"
    },
    "schedule": {
      "spec": "0/10 * * * * *"
    }
  }
}
    
```

2.10.2.9.2 获取计算脚本详情

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_calculate_task_detail	方法名
actionid	否	String		活动标识

参数名称	是否必选	数据类型	数据约束	描述
uid	是	String		创建统计命题的时候生成的 uid

- 示例

```
{
  "action": "get_calculate_task_detail",
  "uid": "CALCULATE1344002700003"
}
```

- 返回

```
{
  "resp": "get_calculate_task_detail",
  "code": 200,
  "msg": "成功",
  "result": {
    "expr": "STATISTICS1301522700002",
    "result": {
      "precision": 3,
      "var_name": "Y1"
    },
    "trigger": [
      "STATISTICS1301522700002"
    ]
  }
}
```

2.10.2.10 创建过滤/条件模型

2.10.2.10.1 创建过滤模型

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	create_filter_model	方法名
actionid	否	String		活动标识
first_min	否	String		
first_max	否	String		
sigma	否	String		
mean_factor	否	String		

- 示例

```
{
  "action": "create_filter_model",
  "first_min": 4,
  "first_max": 3,
  "sigma": 2,
  "mean_factor": 1
}
```

- 返回

```
{
  "resp": "create_filter_model",
}
```

```

"code": 200,
"msg": "成功",
"result": {
    "uid": "FILTER1412781700013"
}
}
    
```

2.10.2.10.2 创建条件模型

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	create_condition_model	方法名
actionid	否	String		活动标识
logic_expr	是	String		逻辑表达式
obj_var_list	是	String		
default	是	String		

- 示例

```

{
    "action": "create_condition_model",
    "logic_expr": "v1 + 5 > 0",
    "obj_var_list": ["v1"],
    "default": false
}
    
```

- 返回

```

{
  "resp": "create_condition_model",
  "code": 200,
  "msg": "成功",
  "result": {
    "uid": "CONDITION1414618400001"
  }
}
    
```

2.10.2.11 更新过滤/条件模型

2.10.2.11.1 更新过滤模型

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	update_filter_model	方法名
actionid	否	String		活动标识
uid	否	String		
first_min	否	String		
first_max	否	String		
sigma	否	String		
mean_factor	否	String		

- 示例

```
{
```

```

    "action": "update_filter_model",
    "uid": "FILTER1412781700013",
    "first_min": 4,
    "first_max": 3,
    "sigma": 2,
    "mean_factor": 1
}
    
```

- 返回

```

{
  "resp": "update_filter_model",
  "code": 200,
  "msg": "成功",
  "result": {
    "uid": "FILTER1412781700013"
  }
}
    
```

2.10.2.11.2 更新条件模型

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	update_condition_model	方法名
actionid	否	String		活动标识
uid	否	String		
logic_expr	是	String		逻辑表达式

参数名称	是否必选	数据类型	数据约束	描述
obj_var_list	是	String		
default	是	String		

- 示例

```

{
  "action": "update_condition_model",
  "uid": "CONDITION1414618400001",
  "logic_expr": "v1 + 5 > 4200",
  "obj_var_list": ["v1"],
  "default": true
}
    
```

- 返回

```

{
  "resp": "update_condition_model",
  "code": 200,
  "msg": "成功",
  "result": {
    "uid": "CONDITION1414618400001"
  }
}
    
```

2.10.2.12 获取过滤/条件模型详情

2.10.2.12.1 获取过滤模型详情

- 描述

- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_filter_detail	方法名
actionid	否	String		活动标识
uid	否	String		

- 示例

```
{
  "action": "get_filter_detail",
  "uid": "FILTER1410561000074"
}
```

- 返回

```
{
  "resp": "get_filter_detail",
  "code": 200,
  "msg": "成功",
  "result": {
    "first_min": 4,
    "first_max": 3,
    "sigma": 2,
    "mean_factor": 1
  }
}
```


2.10.2.12.2 获取条件模型详情

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_condition_detail	方法名
actionid	否	String		活动标识
uid	否	String		

- 示例

```
{
  "action": "get_condition_detail",
  "uid": "CONDITION1414932900005"
}
```

- 返回

```
{
  "resp": "get_condition_detail",
  "code": 200,
  "msg": "成功",
  "result": {
    "logic_expr": "v1 + 5 > 4200",
    "obj_var_list": [
      "v1"
    ],
    "default": false
  }
}
```

```

    }
}
    
```

2.10.2.13 获取过滤/条件模型列表

2.10.2.13.1 获取过滤模型列表

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_filter_list	方法名
actionid	否	String		活动标识

- 示例

```

{
  "action": "get_filter_list"
}
    
```

- 返回

```

{
  "resp": "get_filter_list",
  "code": 200,
  "msg": "成功",
  "result": [
    "FILTER1414932900005",
    "FILTER1410561000074"
  ]
}
    
```

```

    ]
}
    
```

2.10.2.13.2 获取条件模型列表

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_condition_list	方法名
actionid	否	String		活动标识

- 示例

```

{
  "action": "get_condition_list"
}
    
```

- 返回

```

{
  "resp": "get_condition_list",
  "code": 200,
  "msg": "成功",
  "result": [
    "CONDITION1410556800016",
    "CONDITION1413955100001"
  ]
}
    
```

2.10.2.14 删除过滤/条件模型

2.10.2.14.1 删除过滤模型

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	delete_filter_model	方法名
actionid	否	String		活动标识

- 示例

```
{
  "action": "delete_filter_model",
  "uid": "FILTER1412818200003"
}
```

- 返回

```
{
  "resp": "delete_filter_model",
  "code": 200,
  "msg": "成功"
}
```

2.10.2.14.2 删除条件模型

- 描述
- 格式

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	delete_condition_model	方法名
actionid	否	String		活动标识

- 示例

```
{
  "action": "delete_condition_model",
  "uid": "CONDITION1410556800016"
}
```

- 返回

```
{
  "resp": "delete_condition_model",
  "code": 200,
  "msg": "成功"
}
```

2.10.3 EVENT 列表

2.10.3.1 生成统计/计算结果

- 示例

```
{
  "action": "pub_event",
  "event": {
    "event": "statistics_result",
    "block": "statistics",
    "uniqueid": "PUB1395846800005",

```

```

        "datetime":"2021-04-06 16:10:00",
        "uid":"STATISTICS1395846800005",
        "object_id":"1001",
        "data":"17.8"
    }
}
    
```

2.11 事件持续-Retainer

2.11.1 简介

Retainer 是 Mixiot 体系的通用历程服务，或者叫标签持续服务。一个历程通过标签定义，比如故障、报警、流水线生产等；Retainer 可以配置各种不同标签的历程，记录每个历程的开始、结束、持续时长和发生次数等持续状态。进一步地，记录标签持续过程的具体细节；比如流水线生产过程中，什么时间点执行了一个操作，什么时间点发了某件事情。

2.11.2 历程应用

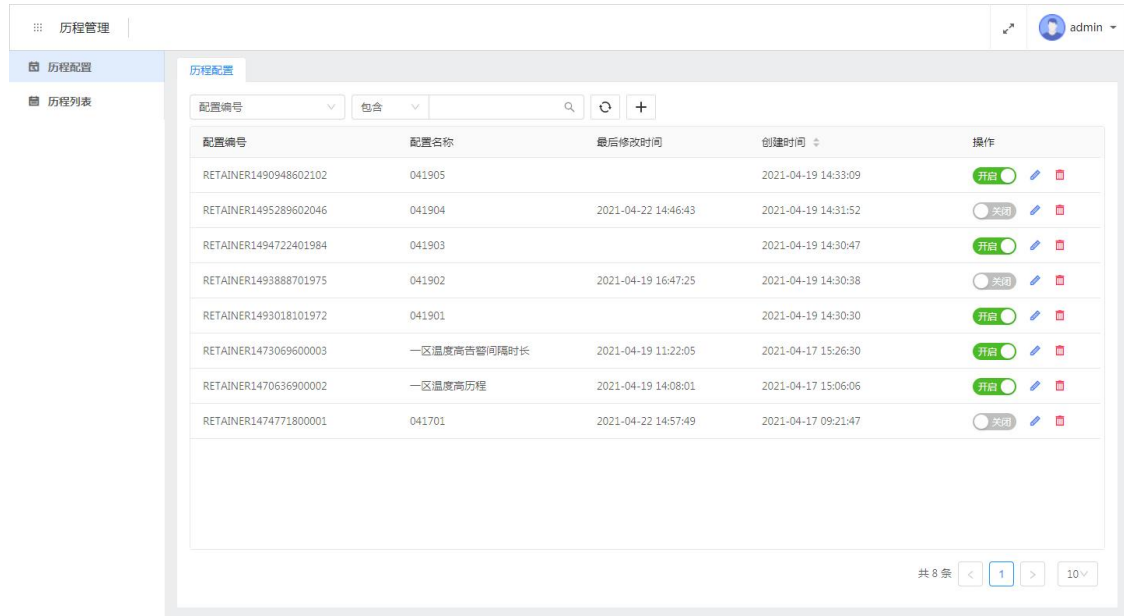
Mixiot 提供 Retainer 应用，界面化管理配置历程。Retainer 以项目方式配置，一个历程项目对 Mixiot 的所有对象有效。



2.11.2.1 历程配置（历程项目）

2.11.2.1.1 列表

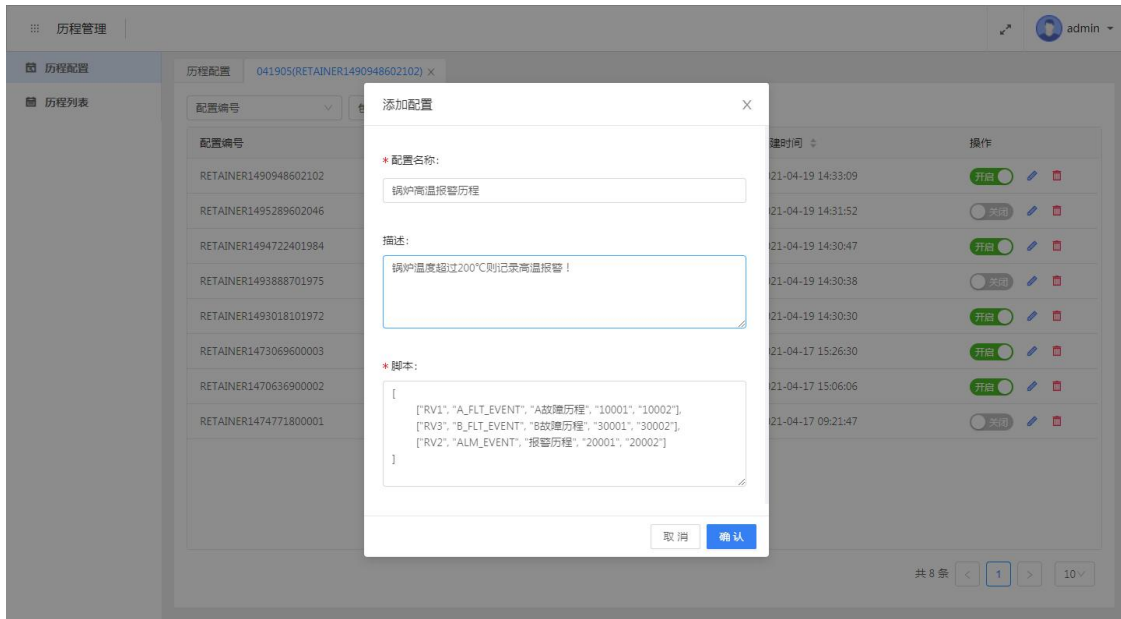
历程配置列表显示配置的所有项目，列表页面可以搜索历程项目记录、添加、编辑、删除记录，以及启动、停止项目等。



配置编号	配置名称	最后修改时间	创建时间	操作
RETAINER1490948602102	041905		2021-04-19 14:33:09	启动
RETAINER1495289602046	041904	2021-04-22 14:46:43	2021-04-19 14:31:52	关闭
RETAINER1494722401984	041903		2021-04-19 14:30:47	启动
RETAINER1493888701975	041902	2021-04-19 16:47:25	2021-04-19 14:30:38	关闭
RETAINER1493018101972	041901		2021-04-19 14:30:30	启动
RETAINER1473069600003	一区温度高告警间隔时长	2021-04-19 11:22:05	2021-04-17 15:26:30	启动
RETAINER1470636900002	一区温度高历程	2021-04-19 14:08:01	2021-04-17 15:06:06	启动
RETAINER1474771800001	041701	2021-04-22 14:57:49	2021-04-17 09:21:47	关闭

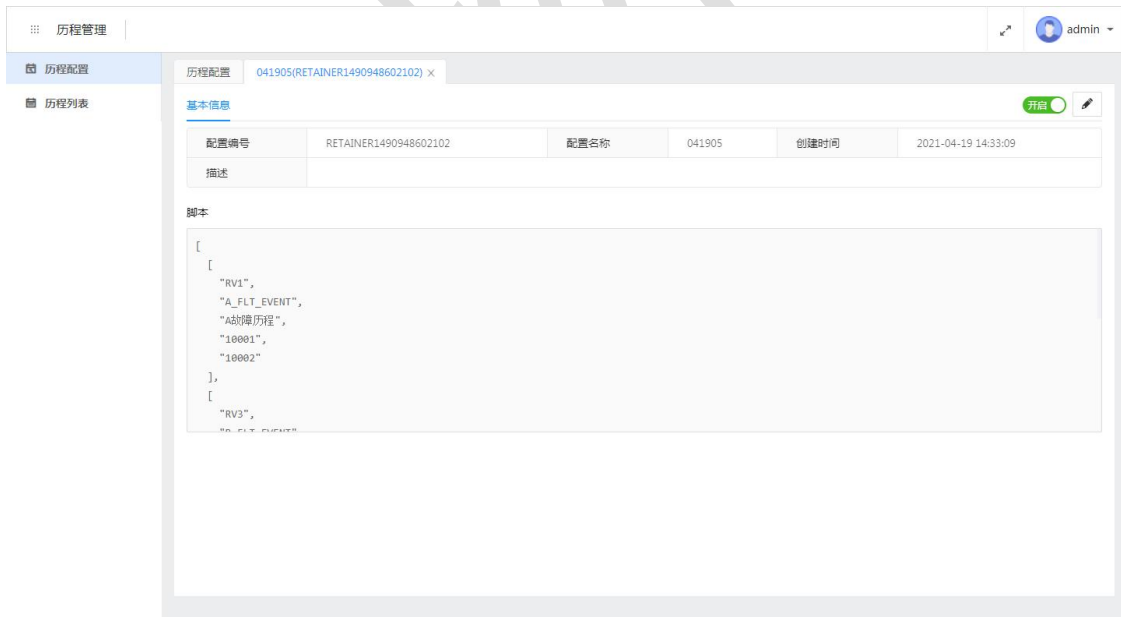
2.11.2.1.2 添加/编辑

添加/编辑页面需要填写配置名称、描述和脚本字段。添加历程项目后，需要启动项目才运行，停止则不运行；删除项目时，则要先停止项目。



2.11.2.1.3 详细

详细页面查看历程配置的信息。



2.11.2.2 历程列表

历程项目启动后，会生成不同的历程记录；历程包括还在持续的开放历程和已经结束的关闭

历程。

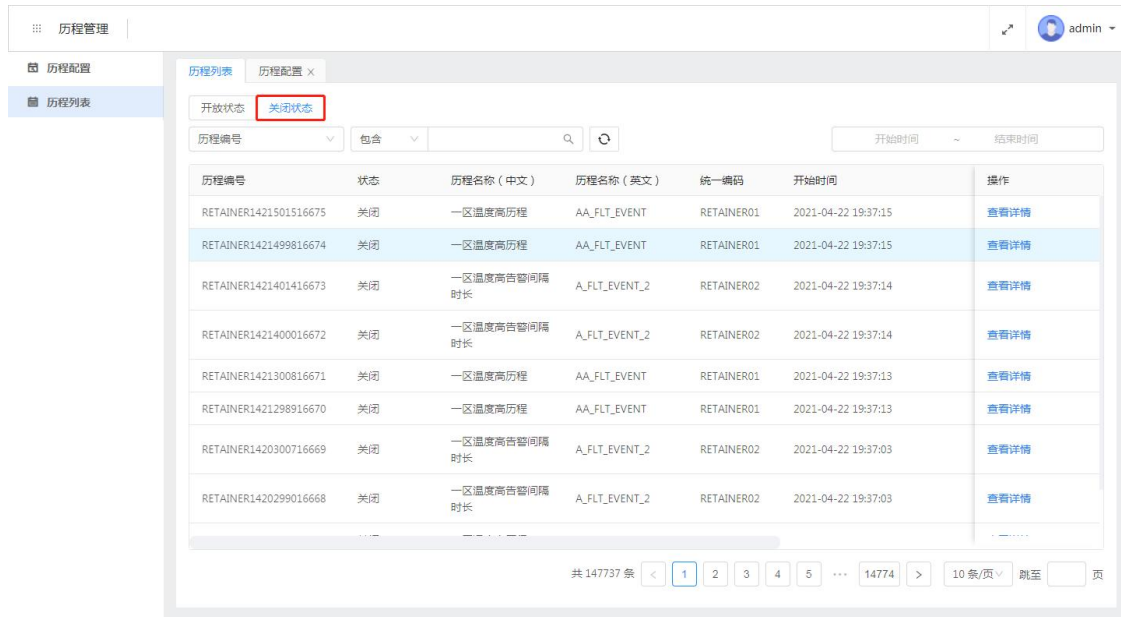
2.11.2.2.1 开放历程（持续历程）

开放历程列表可以查看当前所有的开放历程，开放历程列表页面中可以手动关闭历程（针对一些异常情况时，则可以手动关闭历程；正常情况下无需手动关闭）。

历程编号	状态	历程名称 (中文)	历程名称 (英文)	统一编码	开始时间	操作
RETAINER1422883712417	开放	一区温度高告警间隔时长	A_FLT_EVENT_2	RETAINER02	2021-04-22 17:18:29	手动关闭
RETAINER1422881412416	开放	一区温度高告警间隔时长	A_FLT_EVENT_2	RETAINER02	2021-04-22 17:18:29	手动关闭
RETAINER1494813603839	开放	报警历程	ALM_EVENT	RV2	2021-04-19 16:01:48	手动关闭

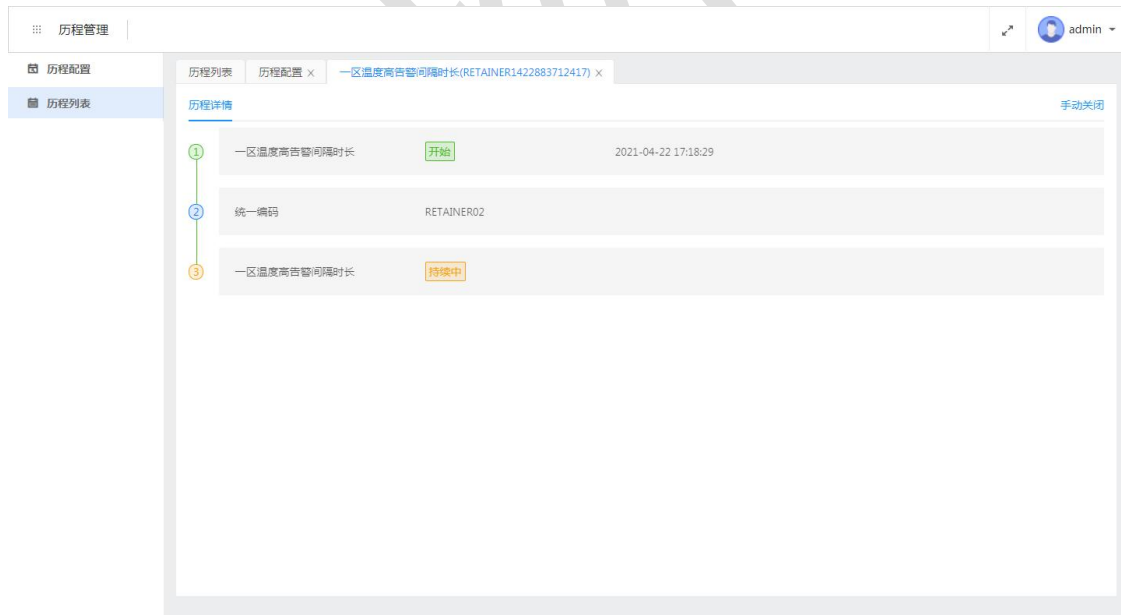
2.11.2.2.2 关闭历程

关闭历程列表可以查看当前所有的关闭历程，开放历程列表页面中可以查看历程详细。



2.11.2.2.3 详细

详细页面查看历程的详细信息。



2.11.3 脚本

Retainer 服务是对 Mixiot 事件的提炼，通过两个事件编码（code）来确定两个事件之间的过

程，并记录分析相关信息，比如一个告警的持续时间，一个设备的运行时长等。脚本格式如下：

[RV, Label_En, Label_Cn, EventCode_Start, EventCode_Stop]

RV 为历程变量(历程编码)，代表一个报警，一个故障或者一个工艺流程

Label_En 为脚本的英文注释

Label_Cn 为脚本的中文注释

EventCode_Start 为对应事件的开始码（事件编码：code a）

EventCode_Stop 为对应事件的结束码（事件编码：code b）

2.11.4 命令列表（action）

2.11.4.1 数据接收

用于接收 agent 的订阅数据。可以直接调用，相当于发送一个需要处理的事件。

api:

http://127.0.0.1:6061/retainer/api

body:

```
{
  "action": "deal",
  "object_id": "OBJ_TEST1001",
  "code": "1001"
}
```

curl:

```
curl -X POST --data '{"action":"deal","object_id":"OBJ_TEST1001","code":"1001"}'
http://127.0.0.1:6061/retainer/api
```

2.11.4.2 手动关闭

用于手动关闭一个历程。需要一个历程的对应属性，包括对象的 uid（object_id），历程统一

编码（code），历程项目（project_id），以这三个属性查找出唯一的一个历程，然后进行关闭。

api:

http://127.0.0.1:6061/retainer/api

body:

```
{  
  "action": "close",  
  "object_id": "OBJ_TEST1001",  
  "code": "a2b",  
  "project_id": "RETAINER1323077500001"  
}
```

curl:

```
curl -X POST --data  
'{"action":"close","object_id":"OBJ_TEST1001","code":"a2b","project_id":"RETAINER1323077500001"}' http://127.0.0.1:6061/retainer/api
```

2.11.4.3 查看历程中的列表

查看正在进行的历程列表。

api:

http://127.0.0.1:6061/retainer/api

body:

```
{  
  "action": "get_open_retainer_list"  
}
```

curl:

```
curl -X POST --data '{"action":"get_open_retainer_list"}' http://127.0.0.1:6061/retainer/api
```

2.11.5 示例

当定义了两个项目，如下

project_id	script
P1	[[{"a2b": "a2b_en", "a2b_cn": "1001", "a2b_end": "1002"}, {"b2c": "b2c_en", "b2c_cn": "1002", "b2c_end": "1003"}]]
P2	[[{"b2c": "b2c_en", "b2c_cn": "1002", "b2c_end": "1003"}, {"c2a": "c2a_en", "c2a_cn": "1003", "c2a_end": "1001"}]]

当项目启动时，存在一个映射关系：`code::rv::project_id=startcode/endcode(0/1)`

具体如下：

```
{
  "1001": {
    "a2b": {
      "P1": 0
    },
    "c2a": {
      "P2": 1
    }
  },
  "1002": {
    "a2b": {
      "P1": 1
    },
    "b2c": {
      "P1": 0,
      "P2": 0
    }
  },
}
```

```

"1003": {
  "b2c": {
    "P1": 1,
    "P2": 1
  },
  "c2a": {
    "P2": 0
  }
}
}
    
```

当前存在三个事件：

object_id	code
O1	1001
O1	1002
O2	1003

当事件上报时，查询是否存在 `object_id+project_id+rv=start/end(0/1)`

当 O1 的 1001 上报时，获取 code 对应关系下，需要处理的 rv。

由于 1001 该 code 找到两个对应关系 a2b.P1 和 c2a.P2，且 a2b.P1 是开始编码，c2a.P2 是结束编码。

所以此时需要开启一个历程：`O1.a2b.P1=0`

此刻的处理完的 rv 的缓存关系如下：

```

{
  "O1.a2b.P1": 0, //新增
  "O1.c2a.P2": 1 //不操作
}
    
```

当 O1 的 1002 上报时，获取 code 对应关系下，需要处理的 rv。

由于 1002 该 code 找到两个对应关系 a2b.P1 和 b2c.P1 和 b2c.P2，且 b2c.P1 和 b2c.P2 是开始编码，a2b.P1 是结束编码。

所以此时需要开启两个历程: O1.b2c.P1=0 和 O1.b2c.P2=0; 同时关闭一个历程: O1.a2b.P1=1

此刻的处理完的 rv 的缓存关系如下:

```
{
    "O1.a2b.P1": 1, //更新
    "O1.b2c.P1": 0, //新增
    "O1.b2c.P2": 0, //新增
    "O1.c2a.P2": 1, //不操作
}
```

当 O2 的 1003 上报时, 获取 code 对应关系下, 需要处理的 rv。

由于 1003 该 code 找到两个对应关系 a2b.P1 和 b2c.P1 和 b2c.P2, 且是 a2b.P1 开始编码, b2c.P1 和 b2c.P2 是结束编码。

所以此时需要开启一个历程: O2.a2b.P1=0

此刻的处理完的 rv 的缓存关系如下:

```
{
    "O1.a2b.P1": 1, //不操作
    "O1.b2c.P1": 0, //不操作
    "O1.b2c.P2": 0, //不操作
    "O1.c2a.P2": 1, //不操作
    "O2.a2b.P1": 0, //新增
    "O2.b2c.P1": 1, //不操作
    "O2.b2c.P2": 1, //不操作
}
```

2.12 维保任务-MixWork

2.12.1 应用场景

维保指针对设备的维护和保养, 类型分为自然时间和运行时间两种:

自然时间: 根据设置的维保周期计算维保到期时间, 当设备的维保到期时间小于等于提醒时

间，就会产生维保信息推送给用户，提醒用户进行维保；

运行时间：根据设置的维保周期和设备统计计算得出的运行时间计算出剩余时间，当设备的剩余时间小于等于提醒时间，就会产生维保信息推送给用户，提醒用户进行维保。

定时维保主要是以项目方式运行，用于设备维保的添加、管理以及记录处理等。

2.12.2 菜单功能

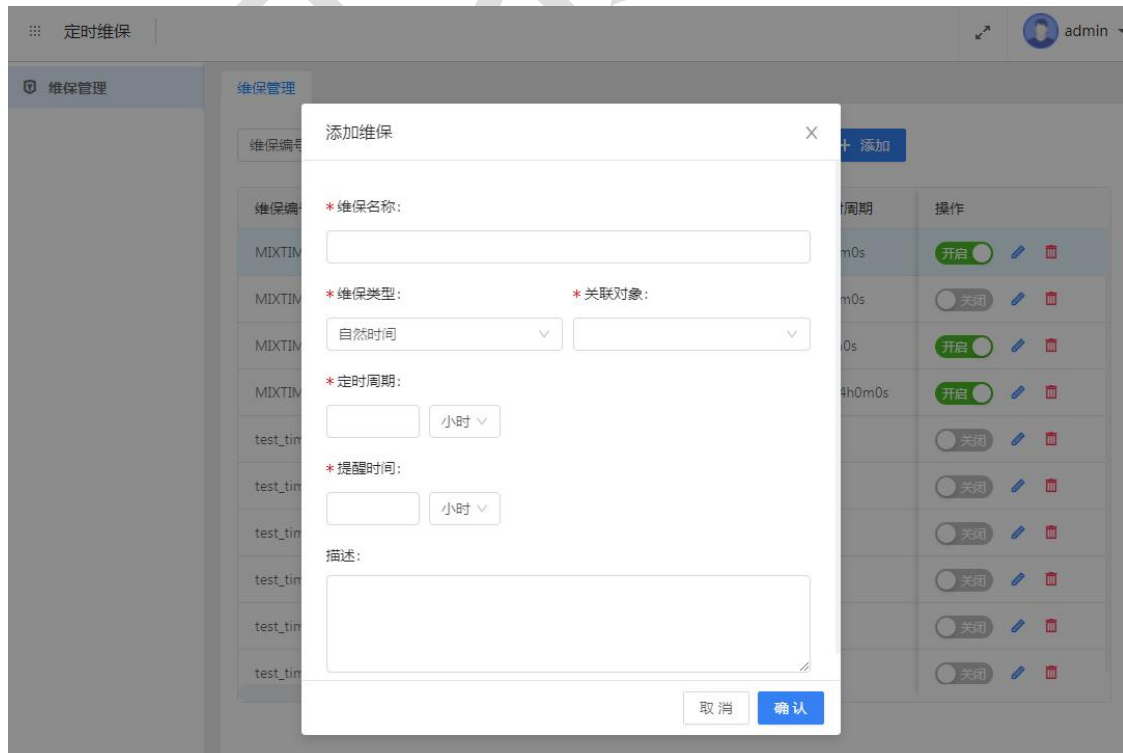
2.12.2.1 维保管理

维保管理菜单，进行维保项目的添加、编辑、查看、删除等操作

2.12.3 使用流程

2.12.3.1 添加维保

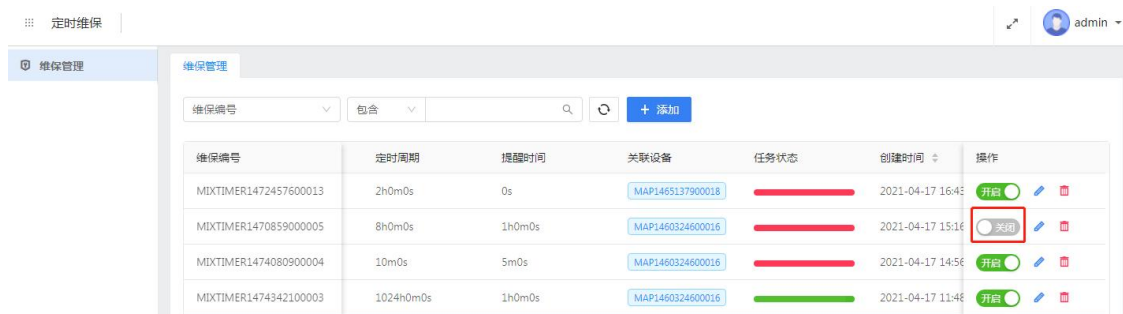
点击进入【维保管理】->【添加】->输入必要信息->【确认】保存



提示：维保是针对设备的，需要关联对象，请参考《对象管理使用文档》添加对象；运行时间类型需要关联统计计算，请参考《统计计算使用文档》添加相关项目

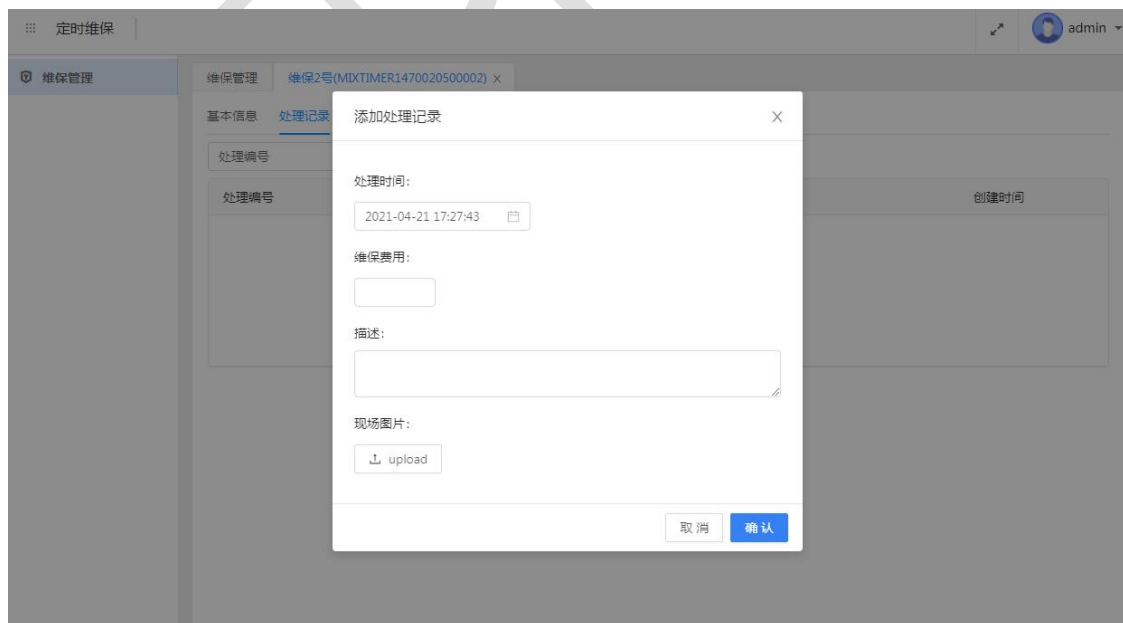
2.12.3.2 项目开启

维保列表->操作拖动【开启】



2.12.3.3 添加处理记录

在维保列表中点击进入某一过期的维保详情->【处理记录】->【添加】->输入必要信息->【确认】保存



提示：维保列表下任务状态为红色的是过期维保，绿色的是正常维保，只有当前为过期

状态的维保才能添加处理记录

2.13 显示板数据服务-Dashbos

Dashbos 是数据推送服务，提供 TCP 和 Websocket 两种订阅方式。使用需要先配置项目脚本来确定订阅的数据的相关信息，再通过 TCP 或 Websocket 的方式对指定的项目发起订阅。发起订阅时，首先会推送一批当前最新的历史数据，数量由脚本确定，之后持续推送实时更新的单条数据。

2.13.1 Dashbos 项目脚本

Dashbos 脚本是推送数据的变量配置脚本，脚本格式大致如下：

```
[
  {
    "source": "mosaic",
    "object_id": "OBJ1001",
    "mapping_id": "MAP1001",
    "var_list": ["key1", "key2"],
    "history": 10
  }
]
```

脚本是一个对象的列表，其中每个对象表示订阅的规则一条订阅数据的规则，发起一次订阅，所有此项目脚本下的订阅规则都会通过长连接发送到客户端。

source 表示数据的来源，目前支持马赛克数据 mosaic，indass 数据 indass，统计计算数据 statistics，不同的数据来源订阅规则的字段有不同，下面分别说明：

mosaic 数据源

```
{
  "source": "mosaic",
  "object_id": "OBJ1001",
  "var_list": ["key1", "key2"],
```

```

        "history": 10
    }

```

object_id 是对象 id

mapping_id 是映射表 id，可不填，如果填了则会订阅此映射表下所有对象

var_list 是对象的 key 的列表

history 是发起订阅时初始历史数据的数量

statistics 数据源

```

{
    "source": "statistics",
    "statistics_id": "STATISTICS1001",
    "object_id": "OBJ1001",
    "history": 10
}

```

相比 mosaic 数据源多一项统计计算的项目 id，因为一个统计项目只有单值的结果所以不需要 var_list 字段指定数据 key，其余字段含义相同。

indass 数据源

```

{
    "source": "indass",
    "indass_block_id": "risk",
    "indass_project_id": "RISK1001",
    "object_id": "OBJ1001",
    "var_list": ["risk_bound_upper", "risk_bound_lower"],
    "history": 10
}

```

相比 mosaic 数据源多了 indass_block_id 和 indass_project_id，具体的 indass block 和项目 id 参考 indass 说明，其余字段含义相同。

此外，每一个订阅项的 object_id 可以换成替换成写 mapping_id，如果写的是 mapping_id，订阅时会将此规则应用到此 mapping_id 下所有的 object_id。

脚本示例：

```
[
  {
    "source": "mosaic",
    "history": 3,
    "var_list": ["E01_1", "E02_1", "L01_1", "L03_1"],
    "object_id": "OBJ1393169400001"
  },
  {
    "source": "indass",
    "history": 30,
    "var_list": ["v_mat_max", "v_mat_min", "v_mat_mean"],
    "object_id": "OBJ1243475500002",
    "indass_block_id": "balance",
    "indass_project_id": "1"
  }
]
```

2.13.2 Dashbos 接口 action 说明

2.13.2.1 订阅数据

订阅数据，限定 TCP 或 Websocket 接口。

选填 object_id 用于订阅项的过滤，如果填写了 object_id 参数，只会订阅 object_id 在此参数列表中的规则。

参数

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	subscribe	方法名
actionid	否	String		活动标识

参数名称	是否必选	数据类型	数据约束	描述
project_id	是	String		项目 id
object_id	否	List		对象 id 列表

示例

```
{
  "action": "subscribe",
  "actionid": "",
  "object_id": ["OBJ1393169400001"], // 选填
  "project_id": "DASHBOS1381267900002"
}
```

推送数据示例

初始历史数据:

```
{
  "history": {
    "E01_1": [["2021-03-23 11:37:22", "1"], ["2021-03-23 11:36:22", "1"], ["2021-03-23 11:15:57", "1"]],
    "E02_1": [["2021-03-23 11:37:22", "1"], ["2021-03-23 11:36:22", "1"], ["2021-03-23 11:15:57", "1"]],
    "L01_1": [["2021-03-23 11:37:22", "0.58"], ["2021-03-23 11:36:22", "0.58"], ["2021-03-23 11:15:57", "0.58"]],
    "L03_1": [["2021-03-23 11:37:22", "206.8"], ["2021-03-23 11:36:22", "206.8"], ["2021-03-23 11:15:57", "207.9"]]
  },
  "object_id": "OBJ1393169400001",
  "project_id": "DASHBOS1381267900002"
}
```

实时更新数据:

```
{
```

```

    "data":{
        "E01_1": 1,
        "E02_1": 1,
        "L01_1": 0.58,
        "L03_1": 201.3
    },
    "object_id":"OBJ1393169400001",
    "project_id":"DASHBOS1381267900002",
    "time": "2021-03-23 12:12:12"
}
    
```

2.13.2.2 获取项目列表

获取项目列表

参数

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_project_list	方法名
actionid	否	String		活动标识
page_index	否	int		页数，不传则获取所有记录
page_size	否	int		页大小，不传则获取所有记录
where_and	否	String		and 条件
where_or	否	String		or 条件
where_in	否	String		in 条件

示例

```

{
    "action":"get_project_list",
    
```

```
"page_index": 1,  
"page_size": 20,  
"where_and": "",  
"where_or": "",  
"where_in": "",  
}
```

响应

```
{  
  "resp": "get_project_list",  
  "actionid": "",  
  "code": 200,  
  "msg": "成功",  
  "result": {  
    "page_index": 1,  
    "page_size": 20,  
    "total_pages": 0,  
    "total_records": 0,  
    "data": [  
      {  
        "id": 1969,  
        "uid": "DASHBOS1",  
        "name": "042701",  
        "description": "",  
        "block_id": "",  
        "class_id": "",  
        "is_active": 0,  
        "option": null,  
        "script": [  
          {
```

```

        "object_id": "OBJ1001",
        "mapping_id": "MAP1001",
        "var_list": ["key1", "key2"],
        "history": 10
    }
],
"created": "2021-04-27T15:28:25+08:00",
"last_modified": "",
"last_modified_by": 0,
"is_available": 1
}
]
}
}

```

2.13.2.3 获取项目

获取项目

参数

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	get_project_item	方法名
actionid	否	String		活动标识
uid	是	String		项目 uid

示例

```

{
    "action": "get_project_item",
    "uid": "DASHBOS1"
}

```


响应

```
{
  "resp": "get_project_item",
  "actionid": "",
  "code": 200,
  "msg": "成功",
  "result": {
    "id": 1969,
    "uid": "DASHBOS1",
    "name": "042701",
    "description": "",
    "block_id": "",
    "class_id": "",
    "is_active": 0,
    "option": null,
    "script": [
      {
        "object_id": "OBJ1001",
        "mapping_id": "MAP1001",
        "var_list": ["key1", "key2"],
        "history": 10
      }
    ],
    "created": "2021-04-27T15:28:25+08:00",
    "last_modified": "",
    "last_modified_by": 0,
    "is_available": 1
  }
}
```

2.13.2.4 创建项目

创建项目

参数

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	create_project	方法名
actionid	否	String		活动标识
name	是	String		项目名称
description	否	String		项目描述
script	是	List		项目脚本

示例

```

{
  "action": "create_project",
  "name": "dashbos1",
  "description": "dashbos1",
  "script": [
    {
      "object_id": "OBJ1001",
      "mapping_id": "MAP1001",
      "var_list": ["key1", "key2"],
      "history": 10
    }
  ]
}
    
```

响应

```

{
  "resp": "create_project",
    
```

```

"actionid": "",
"code": 200,
"msg": "成功",
"result": {
    "uid": "DASHBOS1"
}
}
    
```

2.13.2.5 更新项目

更新项目

参数

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	update_project	方法名
actionid	否	String		活动标识
uid	是	String		项目 uid
name	否	String		项目名称
description	否	String		项目描述
script	否	List		项目脚本

示例

```

{
    "action": "update_project",
    "uid": "DASHBOS1",
    "name": "edit",
    "description": "edit",
    "script": [
        {
    
```

```

        "object_id": "OBJ1001",
        "mapping_id": "MAP1001",
        "var_list": ["key3", "key4"],
        "history": 10
    }
]
}

```

响应

```

{
    "resp": "update_project",
    "actionid": "",
    "code": 200,
    "msg": "成功",
    "result": {
        "uid": "DASHBOS1"
    }
}

```

2.13.2.6 删除项目

删除项目

参数

参数名称	是否必选	数据类型	数据约束	描述
action	是	String	delete_project	方法名
actionid	否	String		活动标识
uid	是	String		项目 uid

示例

```
{
```

```
"action": "delete_project",  
"uid": "DASHBOS1"  
}
```

响应

```
{  
  "resp": "delete_project",  
  "actionid": "",  
  "code": 200,  
  "msg": "成功",  
  "result": {  
    "uid": "DASHBOS1"  
  }  
}
```