物联网应用接口文档

修订日期	提交人	审核人	更新内容	版本记录
2022-08-09	杜宇			v1.0.0
2022-11-11	杜宇		参数设备编号sn调整为deviceId	v1.0.1
2022-11-25	杜宇		支持风机、交通信号灯、卷帘门控制，照明设备控制，情报板控制	v1.0.2

> 最新版本都是基于上一版进行扩展和调整。

第一章、说明

物联网应用接口主要用于设备控制，设备信息查询，设备状态查询，为上层应用提供基于http、https协议的接口，报文格式采用Json。

如果需要获取实时数据，可以通过订阅/发布方式，获取实时数据。

一、接口规则

1、设计总则

(1) http接口基于Restful规范，根据不同的设备类型，定义接口地址和接口报文。

> 接口uri格式：/项目名/子项目名/业务类型/设备类型/接口方法
>
> 如：/rhy/iot/deviceQuery/fan/getFanStatus

说明：

- 项目名：项目唯一标识 ， 固定位rhy。
- 子项目名：固定位iot。
- 业务类型：接口的所支持的业务，如deviceQuery。
- 设备类型：设备的具体类型，如风机fan。
- 接口方法：具体的接口方法，例如：getFanStatus。

> 设备控制类接口为异步模式，提供通用接口用于查询控制指令执行情况。
>
> 接口详见第一章，第三节通用接口。

(2) 消息主题，根据不同设备类型，定义topic和报文

> topic格式：项目名_子项目名_receive_设备类型_指令
>
> 如：rhy_iot_receive_fan_runStatus

说明：

- 项目名：项目唯一标识 ， 固定位rhy。
- 子项目名：固定位iot。

- 操作类型：接收(receive) 。
- 设备类型：设备的具体类型，如风机fan。
- 指令：向设备发出的命令，例如：runStatus。

2、数据内容规则

数据格式采用 json 格式字符串，输入报文格式详见各设备接口报文。

输出报文格式，采用通用格式，如下：

```
{
    "code": "200",
    "msg": "成功",
    "data": {}
}
```

key	说明	数据类型	备注
code	状态码	String	状态码
msg	返回信息	String	返回信息
data	返回数据	Object	详见各设备输出报文

> data字段可以为数组，也可以为对象，详细报文参考各设备输出报文。

注意：key 采用驼峰命名规则，首字母需要小写。

二、安全验证

交互过程中报文为密文，Json报文会使用非对称方式加密，使用sdk包中提供的encode和decode方法，及公钥私钥，对报文进行加解密。

三、通用接口

1、查询指令执行情况

> 通用接口不存在设备类型，接口前缀为/rhy/iot/deviceCommon

Tips：设备控制方法调用成功后，data标签会返回actionId，用于查询指令执行情况。

接口方法：/getExecStatus

调用方式：post

输入报文：

```
{
    "actionId": "1"
}
```

key	说明	数据类型	备注
actionId	指令编号	String	指令编号

输出报文：

```
{
    "code": "200",
    "msg": "成功",
    "data": {
        "actionId": "1",
        "execStatus": "01"
    }
}
```

key	说明	数据类型	备注
actionId	指令编号	String	指令编号
execStatus	运行状态	String	00：失败，01：成功，99：执行中

2、获取设备运行状态枚举

> 通用接口不存在设备类型，接口前缀为/rhy/iot/deviceCommon

接口方法：/getRunStatusEnum?deviceType={deviceType}

调用方式：get

输入报文：

deviceType=1329000002

key	说明	数据类型	备注
deviceType	设备类型	String	设备类型

输出报文：

```
{
  "code": "200",
  "msg": "成功",
  "data": {
    "00": "关闭",
    "FF": "故障",
    "01": "正绿反红",
    "02": "正红反绿",
    "03": "正红反红",
    "04": "正转向反红"
  }
}
```

key	说明	数据类型	备注
data	枚举值字	Map	枚举值字典

第二章、风机

一、接口说明

> 风机接口前缀uri：/rhy/iot/{bizType}/fan

1、修改风机运行状态

接口方法：/rhy/iot/deviceCtrl/fan/alterFanRunStatus

调用方式：post

输入报文：

```
{
    "deviceId": "1",
    "runStatus": "00"
}
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
runStatus	运行状态	String	00：停止，01：正转，02：反转，FF：未知

输出报文：

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

2、实时获取运行状态

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_fan_runStatus

协议报文如下：

```
[{
    "deviceId": "1",
    "runStatus": "00"
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
runStatus	运行状态	String	00：停止，01：正转，02：反转，FF：未知

3、实时获取设备状态

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_devStatus

协议报文如下：

```
[{
    "deviceId": "1",
    "onlineStatus": "01",
    "isFault": "0",
    "deviceType": "",
    "facilitiesId": ""
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
onlineStatus	设备状态	String	00：离线，01：在线
isFault	是否故障	String	0：无故障，1：故障
deviceType	设备类型	String	设备类型
facilitiesId	所属设施id	String	所属设施id

4、实时获取设备故障数据

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_devFault

协议报文如下：

```
[{
    "deviceId": "1",
    "id": "",
    "deviceType": "",
    "faultLevel": "",
    "faultCode": "",
    "faultDetail": "",
    "startTime": "2022-09-19 18:58:00",
    "alwaysTime": "2022-09-19 18:58:00"
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
id	故障唯一标识	String	故障唯一标识
deviceType	设备类型	String	设备类型
faultLevel	故障等级	String	故障等级 0:一般故障,1:重大故障,2:特大故障
faultCode	故障状态码	String	故障状态码
faultDetail	故障详情	String	故障详情
startTime	故障开始时间	String	格式：yyyy-MM-dd HH:mm:ss
alwaysTime	故障持续时间	String	格式：yyyy-MM-dd HH:mm:ss

第三章、交通信号灯

一、接口说明

> 交通信号灯接口前缀uri：/rhy/iot/{bizType}/trafficLight

1、修改交通信号灯运行状态

接口方法：/rhy/iot/deviceCtrl/trafficLight/alterTrafficLightRunStatus

调用方式：post

输入报文：

```
{
    "deviceId": "1",
    "runStatus": "01"
}
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
runStatus	运行状态	String	00：关闭，01：红灯，02：绿灯，03：黄灯，04：左转+红灯，FF：未知

输出报文：

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

2、实时获取运行状态

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_trafficLight_runStatus

协议报文如下：

```
[{
    "deviceId": "1",
    "runStatus": "00"
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
runStatus	运行状态	String	00：关闭，01：红灯，02：绿灯，03：黄灯，04：左转绿灯，FF：未知

3、实时获取设备状态

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_devStatus

协议报文如下：

```
[{
    "deviceId": "1",
    "onlineStatus": "01",
    "isFault": "0",
    "deviceType": "",
    "facilitiesId": ""
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
onlineStatus	设备状态	String	00：离线，01：在线
isFault	是否故障	String	0：无故障，1：故障
deviceType	设备类型	String	设备类型
facilitiesId	所属设施id	String	所属设施id

4、实时获取设备故障数据

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_devFault

协议报文如下：

```
[{
    "deviceId": "1",
    "id": "",
    "deviceType": "",
    "faultLevel": "",
    "faultCode": "",
    "faultDetail": "",
    "startTime": "2022-09-19 18:58:00",
    "alwaysTime": "2022-09-19 18:58:00"
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
id	故障唯一标识	String	故障唯一标识
deviceType	设备类型	String	设备类型
faultLevel	故障等级	String	故障等级 0:一般故障,1:重大故障,2:特大故障
faultCode	故障状态码	String	故障状态码
faultDetail	故障详情	String	故障详情
startTime	故障开始时间	String	格式：yyyy-MM-dd HH:mm:ss
alwaysTime	故障持续时间	String	格式：yyyy-MM-dd HH:mm:ss

第四章、车道指示器

一、接口说明

> 车道指示器接口前缀uri：/rhy/iot/{bizType}/laneIndicator

1、修改车道指示器运行状态

接口方法：/rhy/iot/deviceCtrl/laneIndicator/alterLaneIndicatorRunStatus

调用方式：post

输入报文：

```
{
    "deviceId": "1",
    "runStatus": "00"
}
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
runStatus	运行状态	String	00：关闭，01：正绿反红，02：正红反绿，03：正红反红，04：转向，FF：未知

输出报文：

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

2、实时获取运行状态

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_laneIndicator_runStatus

协议报文如下：

```
[{
    "deviceId": "1",
    "runStatus": "00"
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
runStatus	运行状态	String	00：关闭，01：正绿反红，02：正红反绿，03：正红反红，04：转向，FF：未知

3、实时获取设备状态

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_devStatus

协议报文如下：

```
[{
    "deviceId": "1",
    "onlineStatus": "01",
    "isFault": "0",
    "deviceType": "",
    "facilitiesId": ""
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
onlineStatus	设备状态	String	00：离线，01：在线
isFault	是否故障	String	0：无故障，1：故障
deviceType	设备类型	String	设备类型
facilitiesId	所属设施id	String	所属设施id

4、实时获取设备故障数据

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_devFault

协议报文如下：

```
[{
    "deviceId": "1",
    "id": "",
    "deviceType": "",
    "faultLevel": "",
    "faultCode": "",
    "faultDetail": "",
    "startTime": "2022-09-19 18:58:00",
    "alwaysTime": "2022-09-19 18:58:00"
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
id	故障唯一标识	String	故障唯一标识
deviceType	设备类型	String	设备类型
faultLevel	故障等级	String	故障等级 0:一般故障,1:重大故障,2:特大故障
faultCode	故障状态码	String	故障状态码
faultDetail	故障详情	String	故障详情
startTime	故障开始时间	String	格式：yyyy-MM-dd HH:mm:ss
alwaysTime	故障持续时间	String	格式：yyyy-MM-dd HH:mm:ss

第五章、卷帘门

一、接口说明

> 卷帘门接口前缀uri：/rhy/iot/{bizType}/rollDoor

1、修改卷帘门运行状态

接口方法：/rhy/iot/deviceCtrl/rollDoor/alterRollDoorRunStatus

调用方式：post

输入报文：

```
{
    "deviceId": "1",
    "runStatus": "00"
}
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
runStatus	运行状态	String	00：下降，01：上升，02：停止，FF：未知

输出报文：

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

2、实时获取运行状态

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_rollDoor_runStatus

协议报文如下：

```
[{
    "deviceId": "1",
    "runStatus": "00"
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
runStatus	运行状态	String	00：全闭，01：全开，02：半开，FF：故障

3、实时获取设备状态

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_devStatus

协议报文如下：

```
[{
    "deviceId": "1",
    "onlineStatus": "01",
    "isFault": "0",
    "deviceType": "",
    "facilitiesId": ""
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
onlineStatus	设备状态	String	00：离线，01：在线
isFault	是否故障	String	0：无故障，1：故障
deviceType	设备类型	String	设备类型
facilitiesId	所属设施id	String	所属设施id

4、实时获取设备故障数据

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_devFault

协议报文如下：

```
[{
    "deviceId": "1",
    "id": "",
    "deviceType": "",
    "faultLevel": "",
    "faultCode": "",
    "faultDetail": "",
    "startTime": "2022-09-19 18:58:00",
    "alwaysTime": "2022-09-19 18:58:00"
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
id	故障唯一标识	String	故障唯一标识
deviceType	设备类型	String	设备类型
faultLevel	故障等级	String	故障等级 0:一般故障,1:重大故障,2:特大故障
faultCode	故障状态码	String	故障状态码
faultDetail	故障详情	String	故障详情
startTime	故障开始时间	String	格式：yyyy-MM-dd HH:mm:ss
alwaysTime	故障持续时间	String	格式：yyyy-MM-dd HH:mm:ss

第六章、CO/VI

一、接口说明

> CO/VI接口前缀uri：/rhy/iot/{bizType}/covi

1、查询CO/VI当天整点的检测值

接口方法：/rhy/iot/deviceQuery/covi/getCoviHistoryPerHour

调用方式：post

输入报文：

```
{
    "deviceId": "1"
}
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id

输出报文：

```
[
    {
      "co": "99.00",
      "vi": "21.00",
      "time": "0:00"
    },
    {
      "co": "90.00",
      "vi": "18.00",
      "time": "9:00"
    },
    {
      "co": "99.00",
```

```
      "vi": "21.00",
      "time": "10:00"
    }
]
```

key	说明	数据类型	备注
co	一氧化碳含量	String	单位：ppm
vi	能见度	String	单位：m
time	时间	String	整点时刻

2、实时获取业务属性

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_covi_bizAttr

协议报文如下：

```
[{
  "deviceId": "1",
  "co": "300",
  "vi": "50",
  "no": "150",
  "collectTime": "2022-08-08 17:46:00"
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
co	一氧化碳含量	String	单位：ppm
vi	能见度	String	单位：m
no	一氧化氮含量	String	单位：ppm
collectTime	采集时间	String	格式：yyyy-MM-dd hh:mm:ss

3、实时获取设备状态

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_devStatus

协议报文如下：

```
[{
    "deviceId": "1",
    "onlineStatus": "01",
    "isFault": "0",
    "deviceType": "",
    "facilitiesId": ""
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
onlineStatus	设备状态	String	00：离线，01：在线
isFault	是否故障	String	0：无故障，1：故障
deviceType	设备类型	String	设备类型
facilitiesId	所属设施id	String	所属设施id

4、实时获取设备故障数据

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_devFault

协议报文如下：

```
[{
    "deviceId": "1",
    "id": "",
    "deviceType": "",
    "faultLevel": "",
    "faultCode": "",
    "faultDetail": "",
    "startTime": "2022-09-19 18:58:00",
    "alwaysTime": "2022-09-19 18:58:00"
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
id	故障唯一标识	String	故障唯一标识
deviceType	设备类型	String	设备类型
faultLevel	故障等级	String	故障等级 0:一般故障,1:重大故障,2:特大故障
faultCode	故障状态码	String	故障状态码
faultDetail	故障详情	String	故障详情
startTime	故障开始时间	String	格式：yyyy-MM-dd HH:mm:ss
alwaysTime	故障持续时间	String	格式：yyyy-MM-dd HH:mm:ss

第七章、风速风向仪

一、接口说明

> 风速风向仪接口前缀uri：/rhy/iot/{bizType}/anemoclinograph

1、查询风速风向仪当天整点的检测值

接口方法：/rhy/iot/deviceQuery/anemoclinograph/getAnemoclinographHistoryPerHour

调用方式：post

输入报文：

```
{
    "deviceId": "1"
}
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id

输出报文：

```
[
    {
      "windSpeed": "111.00",
      "windDirection": "24.00",
      "temperature": "23.00",
      "humidity": "24.00",
      "time": "8:00"
    },
    {
      "windSpeed": "111.00",
      "windDirection": "24.00",
      "temperature": "23.00",
      "humidity": "24.00",
      "time": "9:00"
    },
    {
      "windSpeed": "124.00",
      "windDirection": "22.00",
      "temperature": "23.00",
      "humidity": "24.00",
      "time": "17:00"
    }
]
```

key	说明	数据类型	备注
windSpeed	风速	String	单位：m/s
windDirection	风向	String	单位：度（正北向0度）
temperature	温度	String	温度，单位：℃
humidity	湿度	String	湿度，单位：%RH
time	时间	String	整点时刻

2、实时获取业务属性

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_anemoclinograph_bizAttr

协议报文如下：

```
[{
    "deviceId": "1",
    "windSpeed": "",
    "windDirection": "",
    "temperature": "",
    "humidity": "",
    "collectTime" : "2022-09-14 10:02:00",
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
windSpeed	风速	String	单位：m/s
windDirection	风向	String	单位：度（正北向为0度）
temperature	温度	String	温度，单位：℃
humidity	湿度	String	湿度，单位：%RH
collectTime	采集时间	String	格式：yyyy-MM-dd hh:mm:ss

3、实时获取设备状态

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_devStatus

协议报文如下：

```
[{
    "deviceId": "1",
    "onlineStatus": "01",
    "isFault": "0",
    "deviceType": "",
    "facilitiesId": ""
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
onlineStatus	设备状态	String	00：离线，01：在线
isFault	是否故障	String	0：无故障，1：故障
deviceType	设备类型	String	设备类型
facilitiesId	所属设施id	String	所属设施id

4、实时获取设备故障数据

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_devFault

协议报文如下：

```
[{
    "deviceId": "1",
    "id": "",
    "deviceType": "",
    "faultLevel": "",
    "faultCode": "",
    "faultDetail": "",
    "startTime": "2022-09-19 18:58:00",
    "alwaysTime": "2022-09-19 18:58:00"
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
id	故障唯一标识	String	故障唯一标识
deviceType	设备类型	String	设备类型
faultLevel	故障等级	String	故障等级 0:一般故障,1:重大故障,2:特大故障
faultCode	故障状态码	String	故障状态码
faultDetail	故障详情	String	故障详情
startTime	故障开始时间	String	格式：yyyy-MM-dd HH:mm:ss
alwaysTime	故障持续时间	String	格式：yyyy-MM-dd HH:mm:ss

第八章、洞内照度检测仪

一、接口说明

> 亮度检测仪接口前缀uri：/rhy/iot/{bizType}/illuminance

1、实时获取业务属性

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_illuminance_bizAttr

协议报文如下：

```
[{
    "deviceId": "1",
    "illuminance": "",
    "collectTime" : "2022-09-14 10:02:00",
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
illuminance	洞内照度值	String	单位：cd/m2
collectTime	采集时间	String	格式：yyyy-MM-dd hh:mm:ss

2、实时获取设备状态

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_devStatus

协议报文如下：

```
[{
    "deviceId": "1",
    "onlineStatus": "01",
    "isFault": "0",
    "deviceType": "",
    "facilitiesId": ""
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
onlineStatus	设备状态	String	00：离线，01：在线
isFault	是否故障	String	0：无故障，1：故障
deviceType	设备类型	String	设备类型
facilitiesId	所属设施id	String	所属设施id

3、实时获取设备故障数据

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_devFault

协议报文如下：

```
[{
    "deviceId": "1",
    "id": "",
    "deviceType": "",
    "faultLevel": "",
    "faultCode": "",
    "faultDetail": "",
    "startTime": "2022-09-19 18:58:00",
    "alwaysTime": "2022-09-19 18:58:00"
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
id	故障唯一标识	String	故障唯一标识
deviceType	设备类型	String	设备类型
faultLevel	故障等级	String	故障等级 0:一般故障,1:重大故障,2:特大故障
faultCode	故障状态码	String	故障状态码
faultDetail	故障详情	String	故障详情
startTime	故障开始时间	String	格式：yyyy-MM-dd HH:mm:ss
alwaysTime	故障持续时间	String	格式：yyyy-MM-dd HH:mm:ss

第九章、洞外光强检测仪

一、接口说明

> 亮度检测仪接口前缀uri：/rhy/iot/{bizType}/brightDetector

1、实时获取业务属性

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_brightDetector_bizAttr

协议报文如下：

```
[{
    "deviceId": "1",
    "brightness": "",
    "collectTime" : "2022-09-14 10:02:00",
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
brightness	洞内照度值	String	单位：cd/m2
collectTime	采集时间	String	格式：yyyy-MM-dd hh:mm:ss

2、实时获取设备状态

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_devStatus

协议报文如下：

```
[{
    "deviceId": "1",
    "onlineStatus": "01",
    "isFault": "0",
    "deviceType": "",
    "facilitiesId": ""
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
onlineStatus	设备状态	String	00：离线，01：在线
isFault	是否故障	String	0：无故障，1：故障
deviceType	设备类型	String	设备类型
facilitiesId	所属设施id	String	所属设施id

3、实时获取设备故障数据

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_devFault

协议报文如下：

```
[{
    "deviceId": "1",
    "id": "",
    "deviceType": "",
    "faultLevel": "",
    "faultCode": "",
    "faultDetail": "",
    "startTime": "2022-09-19 18:58:00",
    "alwaysTime": "2022-09-19 18:58:00"
}]
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
id	故障唯一标识	String	故障唯一标识
deviceType	设备类型	String	设备类型
faultLevel	故障等级	String	故障等级 0:一般故障,1:重大故障,2:特大故障
faultCode	故障状态码	String	故障状态码
faultDetail	故障详情	String	故障详情
startTime	故障开始时间	String	格式：yyyy-MM-dd HH:mm:ss
alwaysTime	故障持续时间	String	格式：yyyy-MM-dd HH:mm:ss

第十章、控制器

一、接口说明

> 控制器接口前缀uri：/rhy/iot/{bizType}/controller

第十一章、照明设备

一、接口说明

> 照明设备接口前缀uri：/rhy/iot/{bizType}/lighting

照明设备目前直接对接系统，控制命令执行结果按同步模式返回。

1、照明设备控制

接口方法：/rhy/iot/deviceCtrl/lighting/lightingCtrl

调用方式：post

输入报文：

```
{
    "deviceId": "1",
    "runStatus": "00"
    "ext": {
        "brightness": "",
        "step": ""
    }
}
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
runStatus	运行状态	String	00：关，01：开
ext	扩展信息	Map	扩展信息
brightness	调光值	String	调光值
step	段号	String	0：入口段1，1：入口段2，2：过渡段1，3：过渡段2，4：基础段，5：出口段1，6：出口段2

输出报文：

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

第十二章、情报板

一、接口说明

> 情报板接口前缀uri：/rhy/iot/{bizType}/board

1、 发送内容

接口方法：/rhy/iot/deviceCtrl/vms/sendContent

调用方式：post

输入报文：

```
{
    "content": [
        {
            "backColor": "",
            "bmpNo": "",
            "delay": 1,
            "fontName": "",
            "foreColor": "",
            "param": 1,
            "playText": "",
            "playType": 1,
            "posX": 1,
            "posY": 1,
            "shadowColor": "",
            "transition": 1,
```

```
            "wordHeight": 1,
            "wordSpace": 1,
            "wordWidth": 1
        }
    ],
    "deviceId": ""
}
```

key	说明	数据类型	备注
backColor	背景颜色	String	字符颜色RGB的十六进制，例如#0000FF，则为0000FF

key	说明	数据类型	备注
bmpNo	图片编号	String	图片编号，当play_type=2时有效
delay	停留时间	int	单位为百分之一秒，范围 2-30000，缺省为 2
fontName	字体代码	String	h 表黑体、k 表楷体
foreColor	字符颜色	String	字符颜色RGB的十六进制，例如#0000FF，则为0000FF
param	特效速度	int	当出字方式为 0 或 1 时，param 无用；当出字方式为2-21 时，param 表速度，范围0-49，缺省为0。其中0表示最快，即每幅画面停留 20 毫秒，param 每增加1停留时间就增加 20 毫秒。
playText	播放内容	String	
playType	播放类型	int	1：文字；2：图片
posX	x 轴坐标	int	
posY	y 轴坐标	int	
shadowColor	阴影颜色	String	字符颜色RGB的十六进制，例如#0000FF，则为0000FF

key	说明	数据类型	备注
transition	出字方式	int	范围 0-21，缺省为 0。0: 清屏（全黑）1: 立即显示 2: 上移 3： 下移 4: 左移 5: 右移 6: 横百叶窗 7: 竖百叶窗 8: 上下合拢 9: 上下展开 10: 左右合拢 11: 左右展开 12: 中心合拢 13: 中心展开 14: 向下马赛克 15: 向右马赛克 16: 淡入 17: 淡出 18: 字符闪烁（闪后消失） 19: 字符闪烁（闪后停留） 20: 区域闪烁（闪后复原） 21: 区域闪烁（闪后区域为黑）
wordHeight	字高度	int	
wordSpace	字间距	int	
wordWidth	字宽度	int	

输出报文：

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

2、设置亮度

接口方法：/rhy/iot/deviceCtrl/vms/setBrightness

调用方式：post

输入报文：

```
{
    "adjustmentMode": 0,
    "brightLevel": 0,
    "deviceId": ""
}
```

key	说明	数据类型	备注
adjustmentMode	亮度调节模式	int	0、自动调节，1手动调节
brightLevel	亮度等级	int	范围为 0-31，0 最暗（不是全黑），31 最亮
deviceId	设备ID	String	设备ID

输出报文：

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

第十三章、电力设备

一、接口说明

> 电力设备接口前缀uri：/rhy/iot/{bizType}/power

1、查询电力设备当前月份每天的用电量

接口方法：/rhy/iot/deviceQuery/power/getPowerHistoryPerDay

调用方式：get

输入报文：

无

输出报文：

```
[
    {
      "energyUse": "20.0",
      "time": "11.1"
    },
    {
      "energyUse": "20.0",
      "time": "11.2"
    },
    {
      "energyUse": "30.0",
      "time": "11.3"
    }
  ]
```

key	说明	数据类型	备注
energyUse	用电量	String	单位：kw/h
time	时间	String	日

2、实时获取电力设备报警消息

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_power_alert

协议报文如下：

```
{
    "deviceId": "1",
    "alertInfo": "【**预制舱】发生高压室温度越限报警。设备：高压室温度传感器；数据项：环境温度;当前报警值：31.02；设定阈值：27。"
}
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
alertInfo	报警信息	String	报警信息

第十四章、紧急电话

一、接口说明

> 电力设备接口前缀uri：/rhy/iot/{bizType}/emergencyPhone

1、实时获取紧急电话报警消息

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_emergencyPhone_alert

协议报文如下：

```
{
    "deviceId": "1"
}
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id

第十五章、火灾报警

一、接口说明

> 电力设备接口前缀uri：/rhy/iot/{bizType}/fire

1、实时获取火灾报警消息

> 由物联网平台实时向MQ服务推送，应用端监听topic即可
>
> topic：rhy_iot_receive_fire_alert

协议报文如下：

```
{
    "deviceId": "1",
    "alertInfo": "火警: 0号机1回路2号地址  智能感烟   层  办公室1 2015/10/12 14:16:52"
}
```

key	说明	数据类型	备注
deviceId	设备id	String	设备id
alertInfo	报警信息	String	事件:+空格+机号+回路号+地址号+空格+设备类型+空格+层号+空格+地理位置+空格+年月日+空格+时分秒