苏州市垃圾分类源头计量设备

接入说明

目 录

帐号申领

识别二维码，进入申领页面：

帐号申领

苏州市垃圾分类源头计量平台接入帐号申领

***01. 申领单位**

请输入内容

***02. 区域**

申领单位所在区域

省/市/区/街道

***03. 联系人姓名**

请输入内容

***04. 联系人电话**

示例格式：13688888888

***05. 联系人邮箱**

用于接收开通的账号

请输入内容

***06. 设备描述**

请输入内容

***07. 设备检测报告**

+

相关说明

1、申领单位：接入单位名称

2、区域：设备拟安装的行政区或者街道；

3、联系人：对接联系人的姓名；

4、联系电话：对接联系人的手机号码；

5、联系人邮箱：用于接收申领的用户名和密码；

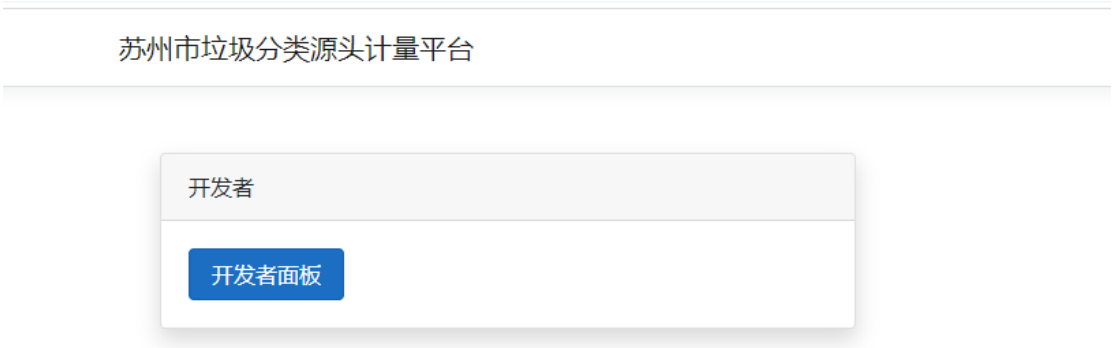
6、设备描述：简单描述一下仪表尺寸、智能设备相关信息；

7、设备检测报告：计量设备送至计量局的计量评估报告

8、设备接入许可：提供智能计量产品的生产许可证书

设备接入测试地址

https://bright.ljflytjl.cn:9101



点击开发者面板，进入登录页面。

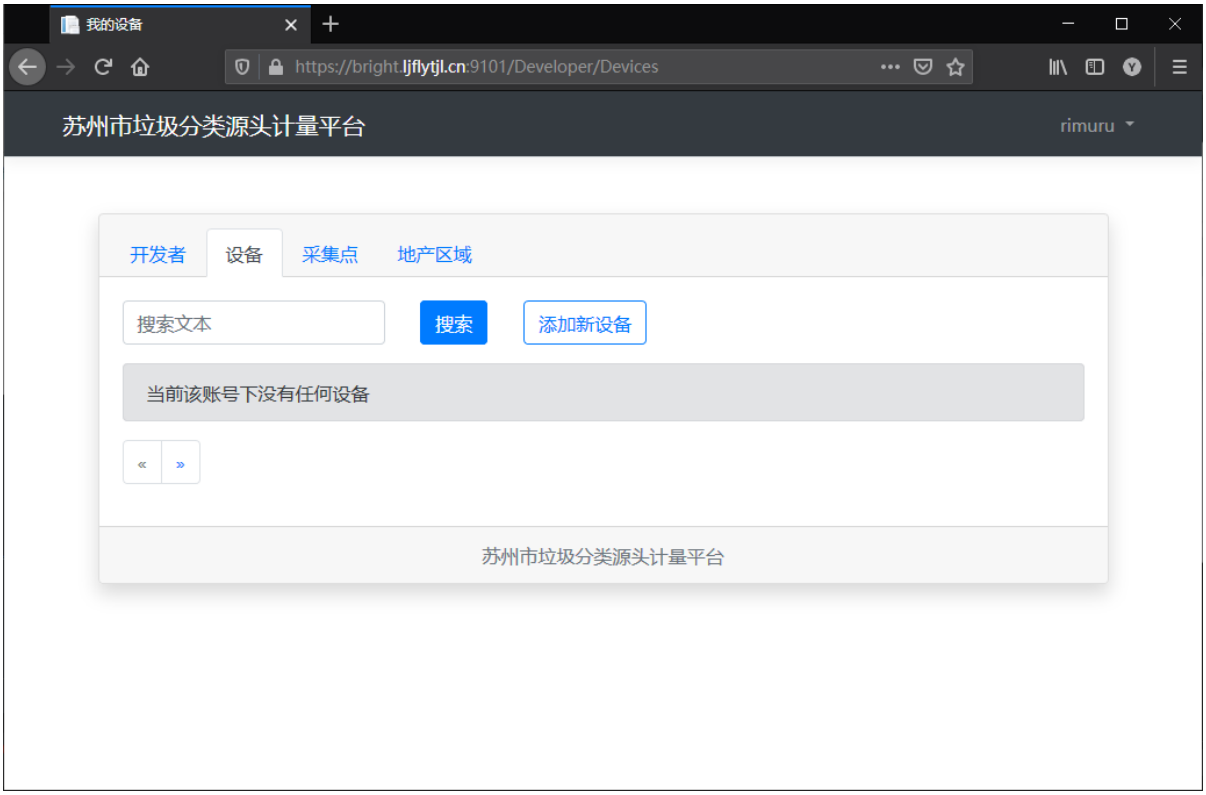
接入指南

准备工作

在申请得到平台账号后，登录平台。

设备

开发者可以在平台的开发者面板管理自己的设备。



当设备未绑定到有效的**采集点**时，该设备相关的 **API** 将不可用.

采集点与地产区域

地产区域：通常指一个商圈、一个小区等。

采集点：地产区域通常应该至少包括一个采集点，如通常所说的“清洁屋”就是一个采集点。

在开发者面板的“采集点”和“地产区域”选项卡中，可以查看当前账户下的采集点和地产区域信息，如需添加等操作信息，请使用 API 进行操作。

API 接入指南

当前平台以开发如下 API 接入形式:

- RESTFUL WebAPI (HTTPS)

Web API 验证

向开发平台发起 Web API 请求时，需要携带一些数据以确认请求者的身份并确保请求本身的安全性。

Web API 分为两种验证形式：

1. 开发者账号信息验证
2. 设备信息验证

具体使用哪种验证方式请查阅具体 API 的说明。

其中，两种 API 验证方式都需要以下验证数据：

数据项	数据类型	描述	备注
secret	**Text**	Secret Hash	该值由服务端生成
nonce	**Number(int32)**	流水号	开发者需要确保该值在 **2** 分钟内不重复，否则会被拒绝请求
sign	**Text**	数据签名 Signature	签名规则请见下方说明
time	**Number**	UNIX 时间戳	请求发起时的时间戳.如果时间不一致会导致请求失败

本系统采用标准 UNIX 时间戳，您的设备可连接至世界任意授时机构平台或使用 GPS、北斗等卫星信号进行校时。

开发者账号信息验证

使用开发者账号信息验证的 API，在上述验证数据的基础上，需要如下数据：

数据项	数据类型	描述	备注
user	**Text**	开发者账户 ID	该值由服务端生成

可通过开发者面板的“开发者”选项卡中的“配置 API 信息”按钮来生成用于账号信息验证的相关数据。

其中“Api Secret”为开发者自行设置，其长度不得小于 8 位，不得大于 32 位。打开页面时会随机生成一个字符串并填入。

> Api Secret 原始值请在设置时妥善保管，丢失后无法找回。

创建完成后，我们将在页面中得到需要的账户 ID 信息和 Secret Hash 信息：

Secret Hash 由开发者设置的 Secret 值通过一定方式计算得出，该计算过程不可逆且具有一定随机性，请以服务端计算结果为准，不要本地推算。

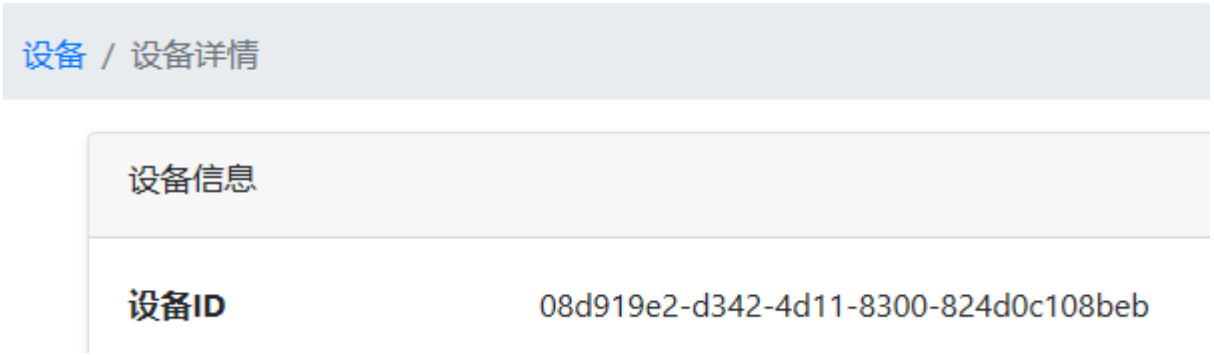
设备信息验证

使用设备信息验证的 API，在上述验证数据的基础上，需要如下数据：

数据项	数据类型	描述	备注
device	**Text**	设备 ID	该值由服务端生成

在添加或编辑设备时，开发者可以指定每个设备的 Secret 值，其规则与注意事项与开发者账号的 Api Secret 一致。

添加设备之后，开发者可在设备详情页查看到设备 ID 和 Secret Hash 值。



Web API 验证方式

本平台 Web API 采用 RESTFUL 设计风格，对于 **GET**、**DELETE** 等通常不在 **Body** 中携带参数的 HTTP 方法，将在 HTTP 请求 **header** 中验证所需的验证数据；对于 **POST** 等通常在 **Body** 中携带参数的 HTTP 方法，将在 HTTP 请求的 **Body** 中验证所需的数据。
其中在不同的参数编排格式中，**Body** 中传递的数据格式略有不同.

Web Api 请求签名规则

动态参数 ： 将**数据主体** 内的所有**必须字段**的数据**按照次序**拼接为字符串。

1. 获取**动态参数**字符串
2. 在**动态参数**字符串后方拼接 **Secret**（Hash 之前的原始文本），得到新字符串，记作“Secret 参数”
3. 将 **Secret 参数**计算获取其**小写 16 位** MD5，得到**签名**

签名参数计算注意事项：

1. 可选参数不计入运算，此外备注中如有特殊标注的参数亦不计入运算。
2. 在运算数字格式时，如数字无小数，请不要携带小数点后位数。

例如 **Hello World 的 API** 中，需要对 **greet** 参数后拼接 **Secret** 文本，然后对其取 16 位小写 MD5.

接入案例指引

下文演示实际编码接入本平台。

本文所用实例接口详见 API 文档: **Hello World**

约定说明

本平台的消息格式约定如下：

```
--[[
    使用开发者账号信息验证的 API
]]
local body = {
    "secret"    = "a490080e777f453b",   -- secret hash
    "nonce"     = 100,                  -- 流水号
    "sign"      = "c720230e678c632e",   -- 数据签名
    "time"      = 1619845237,           -- UNIX 时间戳
    "user"      = "9da14f23-766c-3357-bccd-2497463ac82c",
-- 账户 ID
    "data"      = {
        -- 具体的数据内容
        "greet" = "Hello, Meow!" -- 这里的数据内容按顺序排序来计
算数据签名
    }
}


--[[
    使用设备信息验证的 API
]]
local device_body = {
    "secret"    = "a490080e777f453b",   -- 设备 secret hash
```

```
    "nonce"     = 100,                  -- 流水号
    "sign"      = "c720230e678c632e",   -- 数据签名
    "time"      = 1619845237,           -- UNIX 时间戳
    "device"    = "3fa85f64-5717-4562-b3fc-2c963f66afa6",
-- 设备 ID
    "data"      = {
        -- 具体的数据内容
        "greet" = "Hello, Meow!" -- 这里的数据内容按顺序排序来计
算数据签名
    }
}
```

示例 1

```
package main

import (
    "fmt"
    "io/ioutil"
    "net/http"
    "strconv"
    "time"
)

func main() {

    apiSecretHash := "88ee325fab0791d7"
    userId := "8dd8c6e3-3634-4bc0-b523-9a0e29c7c40e"
    currentTime := time.Now()
    curTimeStamp := strconv.FormatInt(currentTime.Unix(),
10)

    client := &http.Client{}
    req, _ := http.NewRequest("GET",
"https://bright.ljflytjl.cn:9101" + "/api/Hello", nil)
    req.Header.Set("Accept", "text/plain")
    req.Header.Set("secret", apiSecretHash)
```

```
    req.Header.Set("nonce", "10") //实际请求中请确保流水号在两
分钟内不重复
    req.Header.Set("time", curTimeStamp)
    req.Header.Set("user", userId)
    resp, err := client.Do(req)
    if err != nil {
        fmt.Println("Err:" + err.Error())
    }

    defer resp.Body.Close()

    body, err := ioutil.ReadAll(resp.Body)
    if err != nil {

        fmt.Println("解析内容出错" + err.Error())

    }

    fmt.Println("服务端返回: " + string(body))
}
```

示例 2

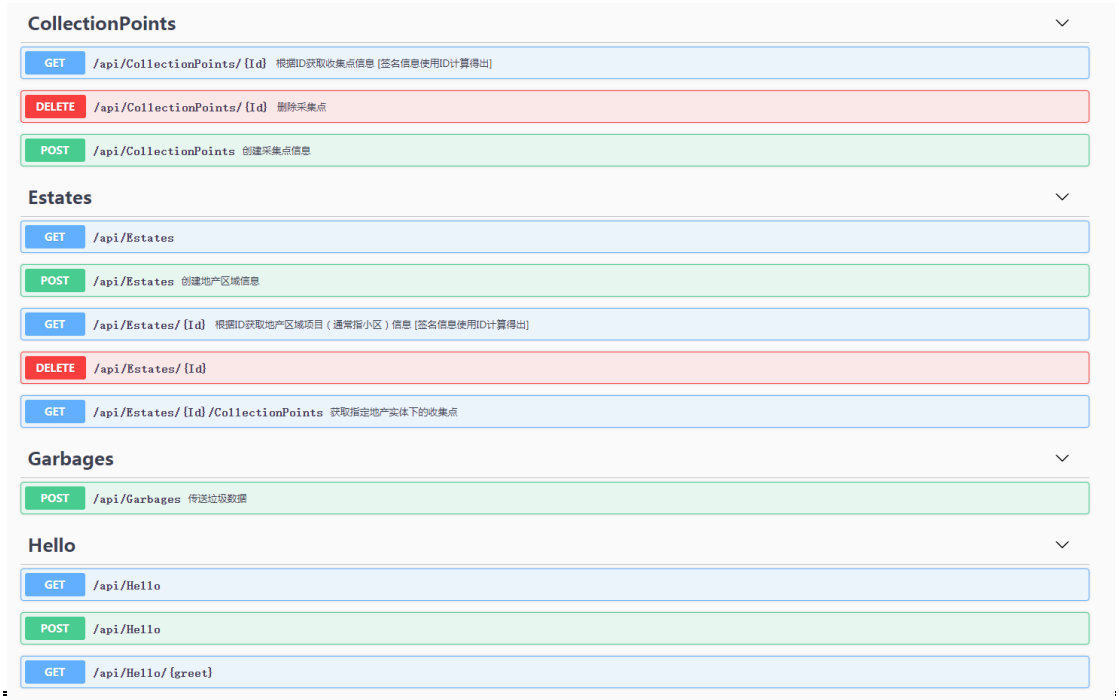
```
local M = {}
M.UserId = "8dd8c6e3-3634-4bc0-b523-9a0e29c7c40e";
M.ApiSecret = "b1yUWkXZgrT1Of7t";
M.ApiSecretHash = "88ee325fab0791d7";


--- 获取字符串格式的 UTC Unix 时间戳
function M.GetUnixTimeStampStr()
    return tostring(os.time(os.date("!*t")));
end

function M.GetSignature(...)
    local paramsStr = "";
    for i , v in ipairs({...}) do
        paramsStr = paramsStr .. tostring(v);
    end
```

```
    local secretParams = paramsStr .. M.ApiSecret;
    return CS.TinaX.EncryUtil.GetMD5(secretParams, true,
true);
end

function M.GetHelloWorld(greet)
    if greet == nil or type(greet) ~= "string" then
        error("greet is invalid.")
    end
    local md5 = M.GetSignature(greet);
    local req = XCore.Http.MakeRequest({
        uri = "https://bright.ljflytjl.cn:9101/api/Hello/" ..
greet,
        header = {
            { key = "Accept", value = "text/plain" },
            { key = "secret", value = M.ApiSecretHash },
            { key = "nonce", value = 10 },
            { key = "time", value = M.GetUnixTimeStampStr()},
            { key = "sign", value = M.GetSignature(greet)},
            { key = "user", value = M.UserId}
        }
    });
    XCore.Http.GetAsync(req, function(resp, err)
        -- onFinish callback
        if err ~= nil then
            error(err.Message)
        end
        print(resp:ReadAsString());
    end)
end

return M;
```

Swagger

由于 API 文档的完善工作量较大，为了让各开发者能够尽快入手开发，本系统临时开放 Swagger 页面.

Swagger 页面地址为: **https://bright.ljflytjl.cn:9101/swagger**



相关事项说明如下：

1. 随着 API 文档的完善，Swagger 页面随时关闭
2. 由于 Swagger 是第三方工具，且默认不支持本平台内部设计的 Dto 描述格式，因此 Swagger 自动生成的内容可能与平台实际运作的 API 接口冲突，包括如下已知问题：
    - 部分接口不会出现在 Swagger 页面上
    - Swagger 页面的调试功能完全不可用

- Swagger 页面所描述的 **Media type** 与实际支持类型有出入
- Swagger 页面所显示的 Dto 在部分数据格式下与实际返回内容不符，请以实际返回内容为准。

3. Swagger 仅供参考，实际请以 API 文档说明为准。
4. Swagger 自带的调试功能在本平台无效，请使用开发者各自擅长的 Web API 调试工具进行调试。
5. 使用 Protocol buffer 格式接入时，按照 swagger 所呈现的 dto 格式编写**.proto** 文件即可，版本为 **syntax = "proto3";**

测试案例：Hello World API

本平台提供 Hello World API 用于开发者的接入测试。

[GET] /api/Hello

- 验证方式：账户信息验证
- 验证传递形式：**Header**
- 参数: 无
- 返回类型: **Text**

[GET] /api/Hello/{greet}

- 验证方式：账户信息验证
- 验证传递形式：**Header**

请求参数

key	类型	可空	签名验证	说明	备注
greet	**Text**	**false**	**1**	-	

返回类型: **Text**

[POST] /api/Hello

- 验证方式：账户信息验证
- 验证传递形式：**Body(DTO)**

请求参数

key	类型	可空	签名验证	说明	备注

greet	**Text**	**true**	**1**	**-**	

返回内容

key	类型	索引顺序	说明	备注
msg	**Text**	**0**	**-**	返回文本
code	**Number**	**1**	**-**	固定返回 **1**

地产区域数据 API

[GET] /api/Estates

- 验证方式：账户信息验证
- 验证传递形式：**Header**

请求参数

请求参数

key	类型	可空	签名验证	说明	备注
page	Number	true	-	页索引	Query Text Parameters
size	Number	true	-	每页数据量	Query Text Parameters
search	Number	true	-	搜索文本	Query Text Parameters

返回内容：

key	类型	索引顺序	说明	备注
id	UUID	0	地产区域ID	-
code	Text	1	地产区域编码	-
addr	Text	2	地产区域所在地址	-
city	Text	3	城市编码	例：320500000000（苏州）
area	Text	4	区域编码	例：320507000000（相城）
street	Text	5	街道编码	例：320507105000（渭塘镇）

[POST] /api/Estates

添加地产区域信息.

- 验证方式：账户信息验证
- 验证传递形式：**Body(DTO)**

请求参数

key	类型	可空	签名验证	说明	备注
code	Text	false	1	地产区域编码	开发者指定，不可重复, 长度限制 2 到 32
name	Text	false	2	名称	长度限制 2 到 32
addr	Text	false	3	地址	长度限制 4 到 128
city	Text	false	4	城市编码	例：320500000000（苏州）
area	Text	false	5	区域编码	例：320507000000（相城）
street	Text	false	6	街道编码	例：320507105000（渭塘镇）

返回内容：

key	类型	索引顺序	说明	备注
id	UUID	0	地产区域ID	-
code	Text	1	地产区域编码	-
addr	Text	2	地产区域所在地址	-
city	Text	3	城市编码	例：320500000000（苏州）
area	Text	4	区域编码	例：320507000000（相城）
street	Text	5	街道编码	例：320507105000（渭塘镇）

[GET] /api/Estates/{Id}

查询地产区域信息.

- 验证方式：账户信息验证
- 验证传递形式：**Header**

请求参数

key	类型	可空	签名验证	说明	备注
Id	UUID	false	1	Id	Query Text Parameters

返回内容：

key	类型	索引顺序	说明	备注
id	UUID	0	地产区域ID	-
code	Text	1	地产区域编码	-
addr	Text	2	地产区域所在地址	-
city	Text	3	城市编码	例：320500000000（苏州）
area	Text	4	区域编码	例：320507000000（相城）
street	Text	5	街道编码	例：320507105000（渭塘镇）

推送垃圾采集数据 API

[POST] api/Garbages

向服务端推送测试用垃圾采集数据.

> 该接口仅供测试，请自行备份保存相关数据。

- 验证方式：设备信息验证
- 验证传递形式：**Body(DTO)**

请求参数

key	类型	可空	签名验证	说明	备注
weight	Number	false	1	垃圾称重数据	64位浮点进度，单位为千克
trash	Text	false	2	垃圾桶编码	-
type	Number	false	3	垃圾类型	-
scanningTime	Number	false	4	数据扫描时间	UNIX时间戳
d_status	Number	false	5	设备状态	-

返回内容：HTTP 状态码

垃圾类型说明

固定枚举类型：

- 缺省类型 ：**0**
- 厨余垃圾 ：**1**
- 可回收物 ：**2**
- 有害垃圾 ：**3**
- 其他垃圾 ：**4**

尽量请勿传递缺省类型

设备状态说明

- 使用中 ：**0**
- 异常 ：**1**
- 检修 ：**2**
- 检修结束 ：**3**
- 启用 ：**4**
- 未知 ：**5**