# 金蝶工业互联网标识解析

# 本地API接口使用规范说明

## 1 版本

v0.3

## 2 修改记录

|  |  |  |
| --- | --- | --- |
| 版本号 | 修改内容 | 日期 |
| v0.1 | 初版 | 2024.05.07 |
| v0.2 | API后端代码重新设计后 | 2024.06.19 |
| V0.3 | 讲解进一步细化 | 2024.06.27 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## 3 本地API配置全过程

出于信息安全的考虑，企业通常需要配置本地API来验证前来查询数据的请求是否合法，如是否拥有权限、权限有效时间是否过期等问题。通过配置本地API + token访问模式可以有效解决此问题。

### 3.1 API数据源基本信息配置

本地API的配置中会有一部分通用的配置，这些通用的内容便是「API数据源」配置，可以理解为API数据源与本地API的关系为一对多的关系，换句话理解也可以是多个API可能会沿用同一套API数据源的配置。故此处将API数据源配置提炼出来作为通用资源，避免过多重复操作。

以下是数据源配置基本信息概况及其UI分布：

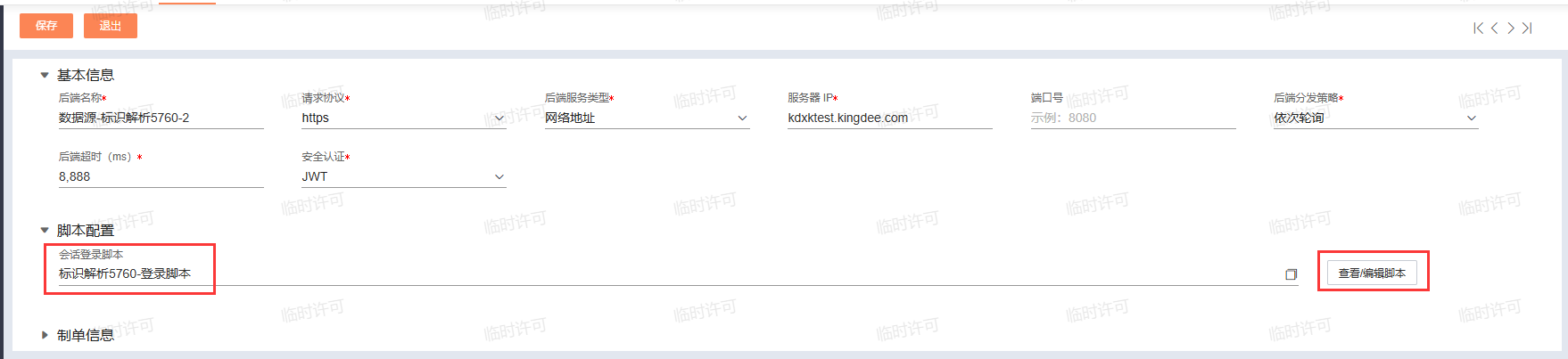


|  |  |  |
| --- | --- | --- |
| 字段名称 | 必填 | 备注 |
| 后端名称 | 是 | 自定义数据源名称。如：“主导-\*\*\*\*\*\*电气股份有限公司” |
| 请求协议 | 是 | http或https |
| 后端服务类型 | 是 | 默认为“网络地址” |
| 服务器IP | 是 | 请填写存放数据的主机IP。如：“\*\*\*\*\*.kingdee.com” |
| 端口号 | 否 | 如有特定端口号请填写；没有则为默认端口号 |
| 后端分发策略 | 是 | 默认为“依次轮询” |
| 后端超时 | 是 | 单位为毫秒 |
| 安全认证 | 是 | 默认为JWT，也可以选择“无认证”。请注意选择“无认证”后将不执行会话登录脚本 |

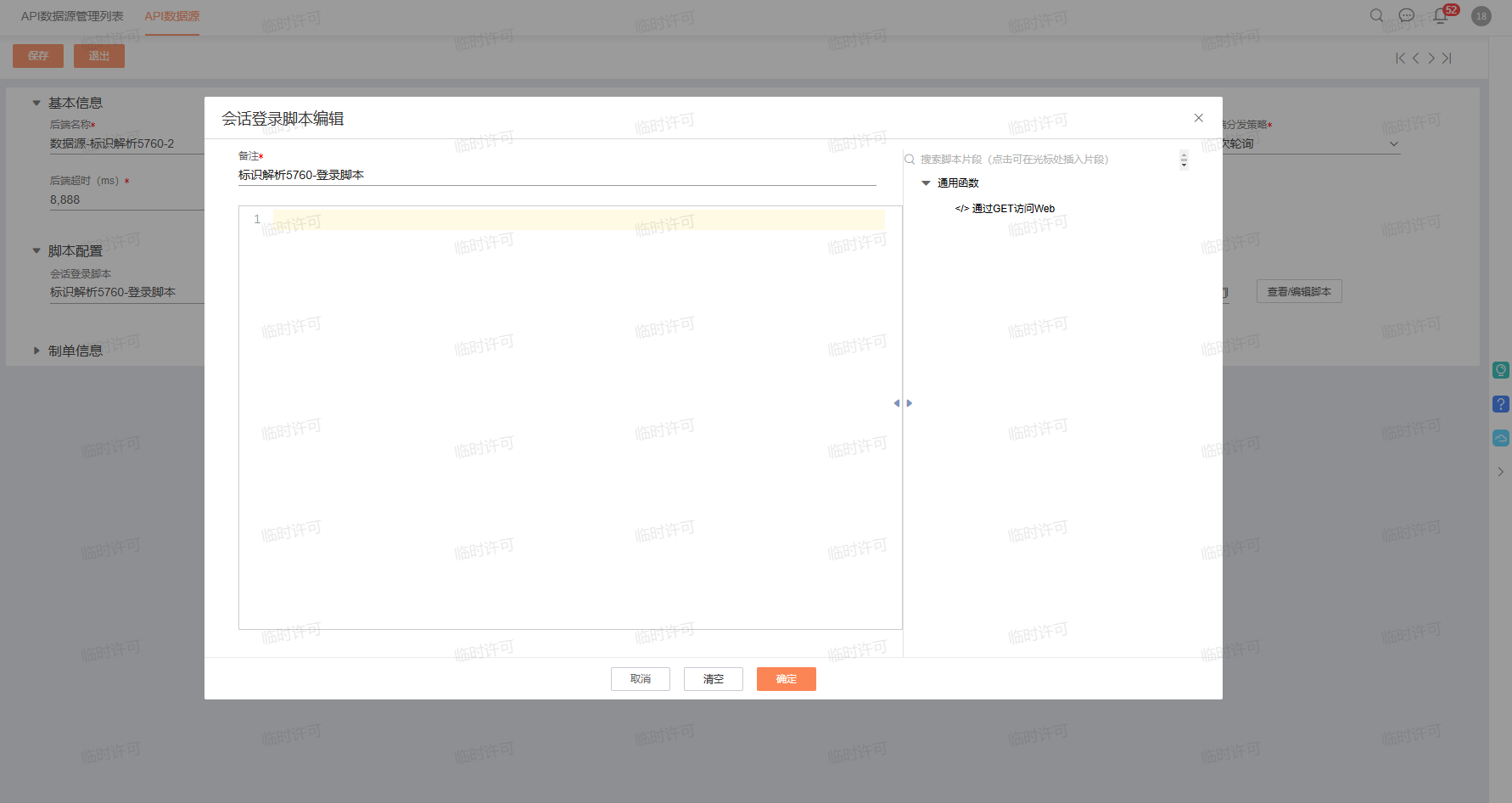
### 3.2 会话登录脚本

如果标识解析访问贵司数据需要认证，则需配置「会话登录脚本」。执行会话登录脚本就是获取认证的过程；不需认证则不需要配置脚本。

左侧为会话登录脚本的备注名，或者说是标题，右侧按钮则可以编辑脚本内容。



点击右侧编辑脚本内容后会弹出一个动态表单。在左侧空白处即可编辑脚本。右侧可以使用系统预留的脚本片段。



以下是一个可以成功调用的获取token的「会话登录脚本」示例，可进行参考。

// 第一部分：获取AppToken

var url = "https://\*\*\*\*\*\*\*.kingdee.com/\*\*\*\*\*\*\*/api/getAppToken.do";

var header = {

"Content-Type":"application/json; charset=utf-8"

};

var params = {String.FormatJson(

{"appId":"\*\*\*\*\*\*",

"appSecret":"\*\*\*\*\*\*"

})};

sessions = HttpPost(url, params, "utf-8", null, header).result;

var session = String.ParseJson(sessions);

var apptoken = session.data.app\_token;

// 第二部分：获取AccessToken

var url = "https://\*\*\*\*\*\*\*.kingdee.com/\*\*\*\*\*\*\*/api/login.do";

var header = {

"Content-Type":"application/json; charset=utf-8"

};

var params = {

String.FormatJson({

"user":"185\*\*\*\*\*\*\*\*",

"apptoken":apptoken})

};

sessions = HttpPost(url, params, "utf-8", null, header).result;

var session = String.ParseJson(sessions);

var accesstoken = session.data.access\_token;

// 第三部分：构建包含header的请求头

String.FormatJson({

"header":{

"accesstoken":accesstoken

}

});

以上脚本示例整体上分为三大部分。

第一部分是获取AppToken（应用token）。其中第一部分的url变量可以对应获取贵司AppToken的URL。

header为可选项（也可以不用写），"Content-Type":"application/json; charset=utf-8"为建议的内容。

params变量包含了请求体的参数，比如appId（应用ID）、appSecret（秘钥）等。

String.FormatJson函数将Map/List组成的对象格式化为JSON字符串。

将params组装完毕后，就可以发起POST请求了。在以上HttpPost方法中，分别输入了五个参数，分别是：目标URL、请求体参数params、编码类型（全球广泛泛用的UTF-8）、访问的cookies、请求头header。

调用HttpPost方法就可以获取AppToken了。获取后返回，后置的“.result”意思表示取返回结果中result字段的值，赋值给session。

第二部分获取AccessToken（进入系统内部的token）。这里的url变量需要替换为企业对应的获取AccessToken的URL。header为可选项。

这次调用POST请求就可以带上user（用户名）和AppToken，获取AccessToken。并赋值给sessions变量，sessions变量经过String.ParseJson方法解析后成为Map/List对象，再赋值给session。“session.data.access\_token”表示取session变量的data字段下的access\_token字段的值。最终赋给accesstoken变量。

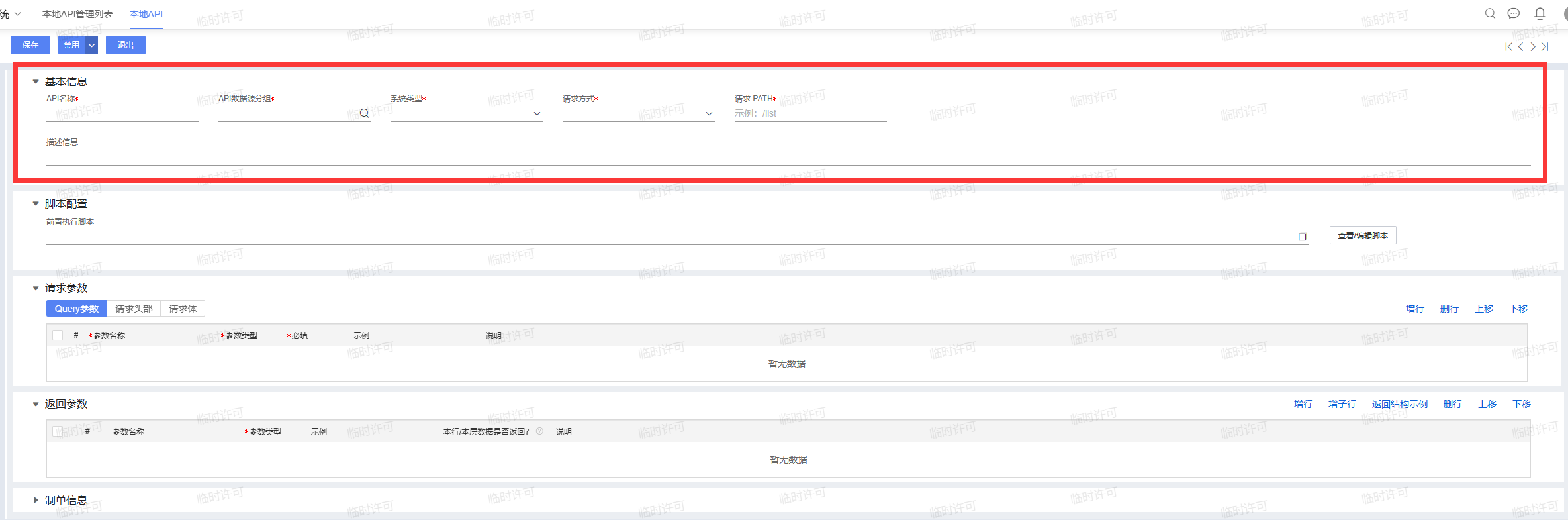
第三部分中，将第二部分中的accesstoken变量赋值给“accesstoken”字段，组装进一个新的header请求体中，并解析为字符串。全部脚本执行完以后，会拿到类似于这样的字符串：

{"header":{"accesstoken":"1737726643826328576\_cNkU1D2qR2ktmjbg14jJ8VqTf8pA13UxNstnW3MJ6c4WU5Tm3HzGiVX1DfRIITLawRRhfx7FK0QKdf44xbcszxlzHh71sYGTMop8"}}

则说明脚本执行成功。

### 3.3 本地API基本信息配置

本地API基本信息分布如下图。

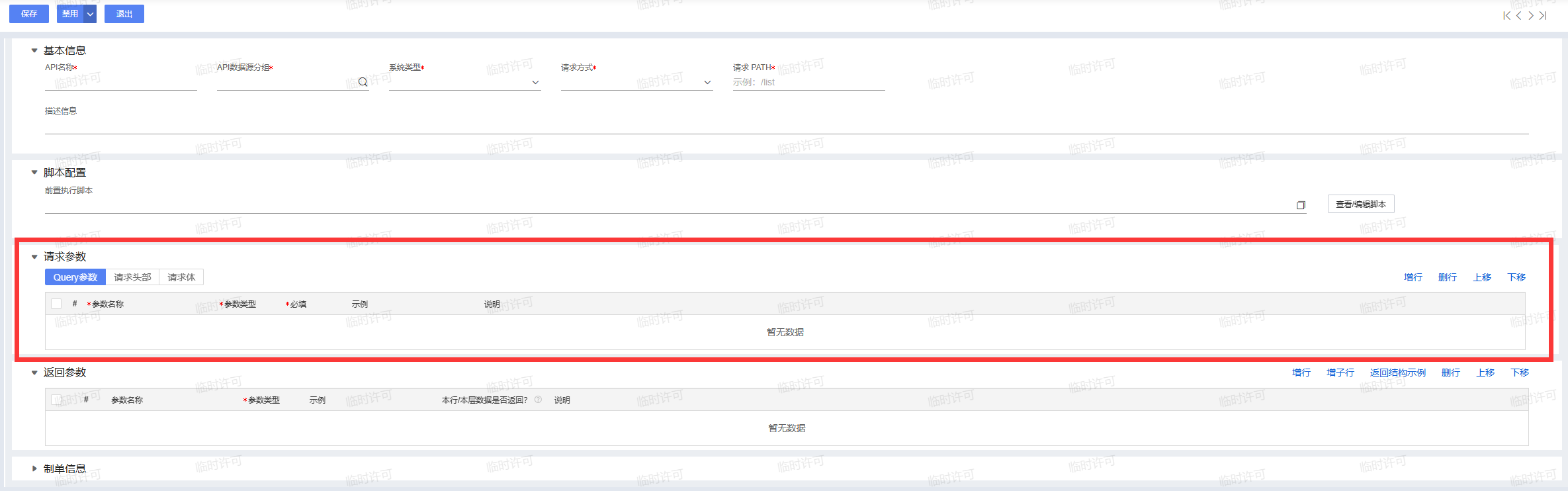


字段描述如下表格。

|  |  |  |
| --- | --- | --- |
| 字段名称 | 是否必填 | 备注 |
| API名称 | 是 | 自定义API名称。如：“主导-\*\*\*\*电气股份有限公司” |
| API数据源分组 | 是 | 正是前面配置好的数据源，可多对一复用 |
| 系统类型 | 是 | 根据实际情况选择对应产品 |
| 请求方式 | 是 | GET与POST |
| 请求PATH | 是 | 如：/\*\*\*\*\*/\*\*\*/\*\*\*/\*\*\*\*/\*\*\*  请注意斜线要正确书写，类似于示例这样 |
| 描述信息 | 否 |  |

### 3.4 前置执行脚本与请求参数

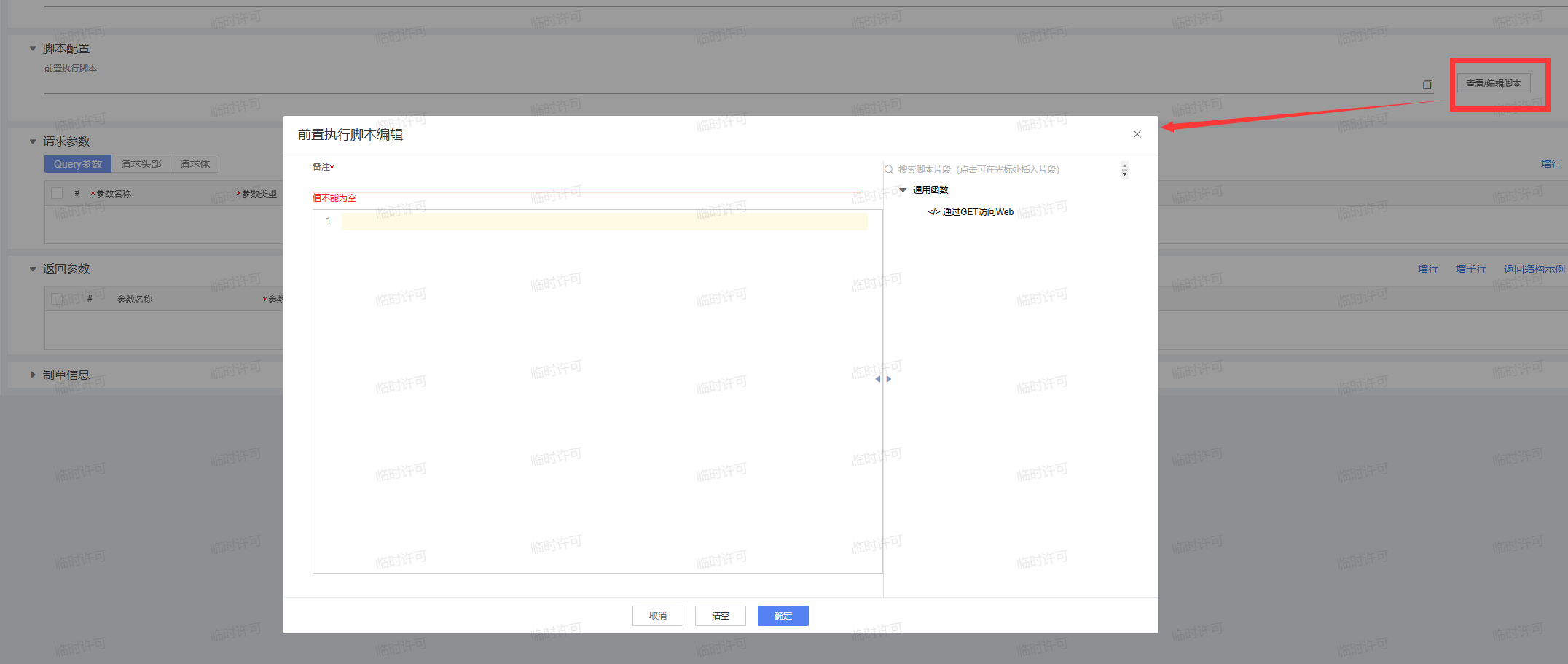
在会话登录部分配置完成以后，就可以开始构建请求参数的内容了。



注意事项如下：

* 如果是选择了GET方法，那么可以在本地API请求参数面板中定义Query参数与请求头；如果是POST方法，那么还可以额外定义请求体。
* 发起请求的顺序是：Web请求会先在本地API面板上取请求头、请求体、Query参数的值，然后再执行前置脚本。其中脚本可以对拿到的面板上的值进行修改（但不会修改源页面上本身的值），并最终以脚本上修改后的为准，并发起请求。
* 当然，如果在面板中配置好的话，前置脚本也可以不写。但是如果脚本中就将请求体、请求头、Query参数配置好的话，那么面板也可以不用写。
* Query参数支持中文，采用的是全球广泛使用的UTF-8进行编码。

如果要使用「前置执行脚本」，请按照已经定义好的变量进行脚本代码的编写。



|  |  |
| --- | --- |
| 上下文变量名称 | 备注 |
| $header | 请求头 |
| $query | Query参数 |
| $body | 请求体 |
| $protocol | 请求协议（如https、http） |
| $ip | 主机地址 |
| $port | 端口号 |
| $url | 获取数据接口URL |
| $req\_type | 请求方法（如GET、POST） |
| $expires\_in | 会话超时时间（秒） |
| $request\_context | 请求上下文 |
| $backend\_timeout | 后端响应超时时间（毫秒） |
| $sys\_type | 系统类型 |
| $backend\_server\_type | 后端服务类型 |
| $backend\_distribute\_strategy | 后端分发策略 |
|  |  |
|  |  |

以上「已经定义好」的、美元符号“$”前缀的这些变量，可以直接在脚本中获取，同事也可以在脚本中查询内容或修改。

前置脚本参考：

var $header = {

"Content-type":"application/json"

};

$query.chinese1 = "脚本修改中文";

$body.package.bizType = "business\_is\_business";

$body.package.id = 12345678;

$body.body2 = "modified\_body2";

### 3.5 返回参数

在这里定义返回参数名称，支持多层子行定义。

倘若要定义多层结构（比如需要增子行），请将本行定义为Object或Array类型。如果定义了多行结构，而父级节点却不是Object或Array类型，那么本数据包将无法予以正常返回。

在多层结构中，可以一键开启所有数据都返回。也可以关闭其中个别行数据的返回。



请注意返回参数这里的参数名称、参数类型，一定要和请求目标URL配置的返回数据包的参数名称、参数类型保持一致。否则将无法正常返回数据。

## 4 调用测试

请求JSON示例如下。

如果是主导企业则通过企业ID来调用，deid为企业ID。

如果是协同企业则通过本地API的ID来调用，localApi为本地API的ID。

{

"localApi":"1962580016319435776"

}

返回内容参考如下：

{

"data":"{\"bs2\":\"bs2\",\"id1\":\"id1\",\"id3\":\"id3\"}",

"errorCode":"0",

"message":null,

"status":true

}