API适配-Body映射如何使用?

1、Body映射概述

通过Body映射可以对输入输出的HTTP BODY进行自定义, 通过内置的标签可以插入原生Js语法以及变量，使您的输入输出完全可以自定义。

2、 示例

```
`{`
`    "error": "<%= $response.statusCode %>",    `
`    "message": "<%= $response.body.error_msg %>",    `
`    "ua": "<%= $context.userAgent %>"`
`}`
```

3、标签说明

Body映射函数可以使用 `<%= ... %>` 插入变量, 也可以用 `<% ... %>` 执行任意的 JavaScript 代码。 如果您希望插入一个值, 并让其进行HTML转义, 请使用 `<%- ... %>`

4、内置对象和方法

(1) $context

变量	说明
`$context.httpMethod`	客户端请求方法
`$context.resourcePath`	请求路径
`$context.sourceIP`	客户端IP
`$context.userAgent`	客户端UserAgent

(2) $request

变量	说明
`$request.rawBody`	base64编码的请求body
`$request.body`	解析为对象的body(前提请求body是json或者xml, 否则为null)
`$request.referer`	浏览器referer
`$request.userAgent`	客户端UserAgent
`$request.params`	请求参数(path或者querystring或者被定义过的body)例如: `$request.params["参数名"]`
`$request.headers`	请求头, 例如: `$request.headers["Content-Type"][0]`
`$request.uri`	带querystring的url
`$request.url`	不带querystring的url
`$request.cookie(x)`	获取某个cookie值
`$request.query(x)`	获取某个querystring或者post表单的值
`$request.header(x)`	获取某个header的值

(3) $response

变量	说明
`$response.rawBody`	base64编码的响应body
`$response.body`	解析为对象的body(前提请求body是json或者xml, 否则为null)
`$response.headers`	响应头, 例如: `$response.headers["Content-Type"][0]`
`$response.statusCode`	响应码
`$response.header(x)`	获取某个header的值

(4) $util

变量	说明
`$util.base64Encode(str)`	base64编码, 返回字符串
`$util.base64Decode(str)`	base64解码, 返回字符串
`$util.jsonEncode(obj)`	将对象编码成为json字符串
`$util.jsonDecode(str)`	将json字符串解码成为对象
`$util.escape(str)`	编码HTML
`$util.unescape(str)`	解码HTML
`$util.each(obj, callbackFunction)`	遍历数组或对象, callbackFunction 格式: `function(value, [key or Index])`
`$util.has(object, key)`	对象是否包含给定的key吗？返回值为bool类型
`$util.keys(object)`	检索object拥有的所有可枚举属性的名称
`$util.values(object)`	返回object对象所有的属性值
`$util.isEqual(object, other)`	执行两个对象之间的优化深度比较，确定他们是否应被视为相等
`$util.isEmpty(object)`	如果object 不包含任何值(没有可枚举的属性)，返回true。 对于字符串和类数组（array-like）对象，如果length属性为0，那么_.isEmpty检查返回true
`$util.isArray(object)`	如果object是一个数组，返回true
`$util.isObject(object)`	如果object是一个对象，返回true。需要注意的是JavaScript数组和函数是对象，字符串和数字不是
`$util.isFunction(object)`	如果object是一个函数（Function），返回true
`$util.isString(object)`	如果object是一个字符串，返回true
`$util.isNumber(object)`	如果object是一个数值，返回true (包括 NaN)
`$util.isBoolean(object)`	如果object是一个布尔值，返回true，否则返回false
`$util.isRegExp(object)`	如果object是一个正则表达式，返回true
`$util.isNaN(object)`	这和原生的isNaN 函数不一样，如果变量是undefined，原生的isNaN 函数也会返回 true
`$util.isNull(object)`	如果object的值是 null，返回true
`$util.isUndefined(value)`	如果value是undefined，返回true
`$util.random(min, max)`	返回一个min 和 max之间的随机整数。如果你只传递一个参数，那么将返回0和这个参数之间的整数
`$util.pairs(object)`	把一个对象转变为一个 [key, value] 形式的数组
`$util.first(array)`	返回array（数组）的第一个元素
`$util.last(array)`	返回array（数组）的最后一个元素
`$util.map(listOrObject, callback)`	通过回调函数批量操作数组或者对象的内容, 例如: `$util.map({one: 1, two: 2}, function(num, key){return num * 3;});`

(5) 更多实例

```
<%
    var contextJson = $util.jsonEncode($context);
    var requestHeaders = {};
    $util.each($request.headers, function(value, key) {
            requestHeaders[key] = value[0];
    });
%>
{
    "context": "<%= contextJson %>",
    "request_headers": "<%= $util.jsonEncode(requestHeaders) %>",
    "response_body": "<%= $util.jsonEncode($response.body) %>",
    "response_raw_body": "<%= $response.rawBody %>"
}
```

最终输出

{ "context": { "httpMethod": "GET", "resourcePath": "/bucket/imgx-test", "sourceIP": "127.0.0.1", "userAgent": "Mozilla/5.0 (Macintosh;