

目录

前言	1.1
Postman简介	1.2
Postman下载	1.3
Postman功能: Request	1.4
新建Request	1.4.1
JSON语法检查	1.4.2
GET的Request的多参数	1.4.3
给接口添加描述	1.4.4
Postman功能: Response	1.5
Response数据显示模式	1.5.1
Response其他功能	1.5.2
保存多个Example	1.5.3
Postman功能: 其他工具和功能	1.6
分组Collection	1.6.1
历史记录History	1.6.2
用环境变量实现多服务器版本	1.6.3
代码生成工具	1.6.4
测试接口	1.6.5
Mock Server	1.6.6
Postman功能: 界面和配置	1.7
多Tab分页	1.7.1
界面查看模式	1.7.2
多颜色主题	1.7.3
Postman生成API文档	1.8
预览和发布API文档	1.8.1
附录	1.9
参考资料	1.9.1

API开发利器：Postman

简介

在涉及HTTP方面的后台REST API开发时，往往需要调试API接口。这方面有很多工具，其中最好用的算是接下来要介绍的Postman了。

源码+浏览+下载

本书的各种源码、在线浏览地址、多种格式文件下载如下：

Gitbook源码

- [crifan/api_tool_postman: API开发利器：Postman](#)

在线浏览

- [API开发利器：Postman book.crifan.com](#)
- [API开发利器：Postman crifan.github.io](#)

离线下载阅读

- [API开发利器：Postman PDF](#)
- [API开发利器：Postman ePub](#)
- [API开发利器：Postman Mobi](#)

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间：2017-12-27 17:36:26

Postman简介

我们在后台开发期间，尤其是HTTP的RESTful API时，往往需要在实现了后台的接口代码后，找合适的工具去调试以验证和确保自己的API接口是可以正常工作的。

这时候，找到合适的API调试工具，就显得很重要。因为合适的工具可以极大地提高工作效率，减少生命的浪费。

之前用过一些工具，最终发现Postman是个非常好用的API调试工具，所以在此推荐之。

关于Postman Postman，其官网有言简意赅的介绍：

Postman helps you develop APIs faster

让（作为后台开发人员的）你开发API接口时更方便和快捷

-» 利用Postman：

- 可以方便的调试API接口
- 还可以把各个API内容发布为文档，方便其他（比如移动端等）人员查看接口详情

下面就来详细的解释Postman的各种功能的用法。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间：2017-12-29 11:59:21

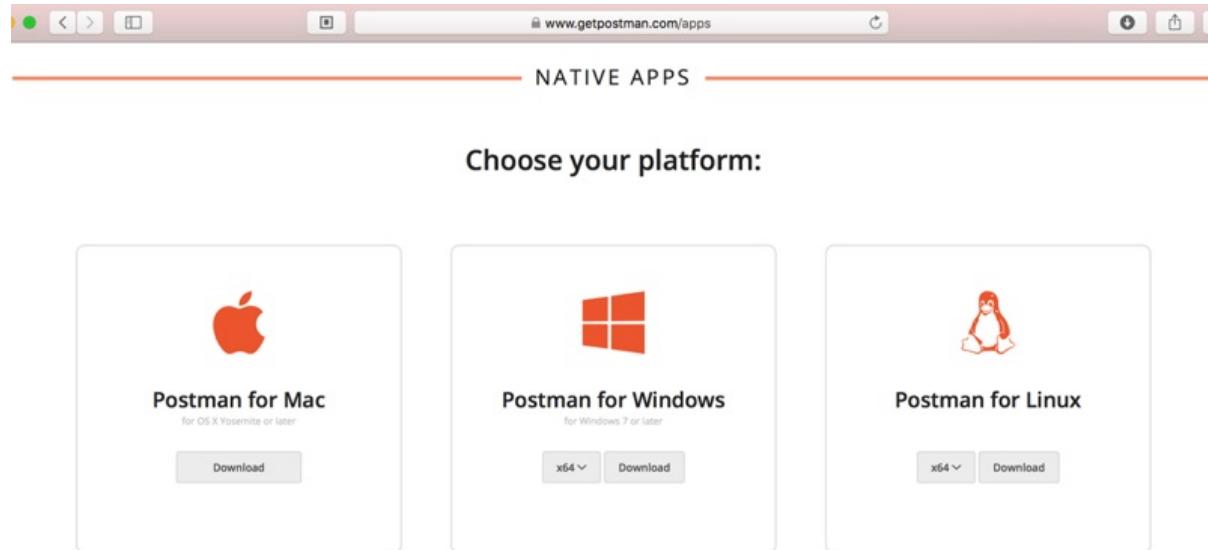
Postman下载

Postman有两种形式：

- **Chrome App:** Postman for Chrome
 - 由于Chrome本身快要废弃Chrome App模式了，所以Postman的Chrome App模式也已废弃逐渐不用了
- **app软件:** 建议下载（不同平台的）独立的软件去使用

下面主要介绍下载安装独立版本app软件的Postman的过程：

去主页[Postman | Supercharge your API workflow](#)找到：[Postman | Apps](#)



去下载自己平台的版本：

- Mac
- Windows (x86/x64)
- Linux (x86/x64) 即可。

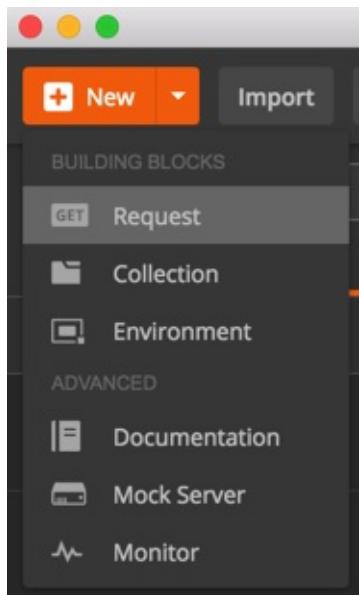
crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间：2017-12-29 13:50:27

Postman功能：Request

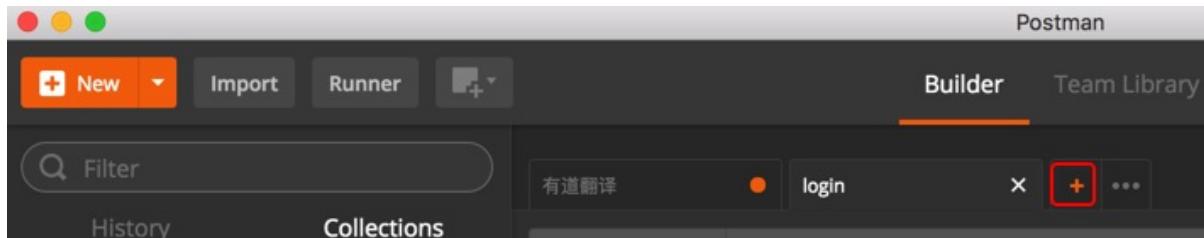
crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2017-12-27 17:37:01

新建Request

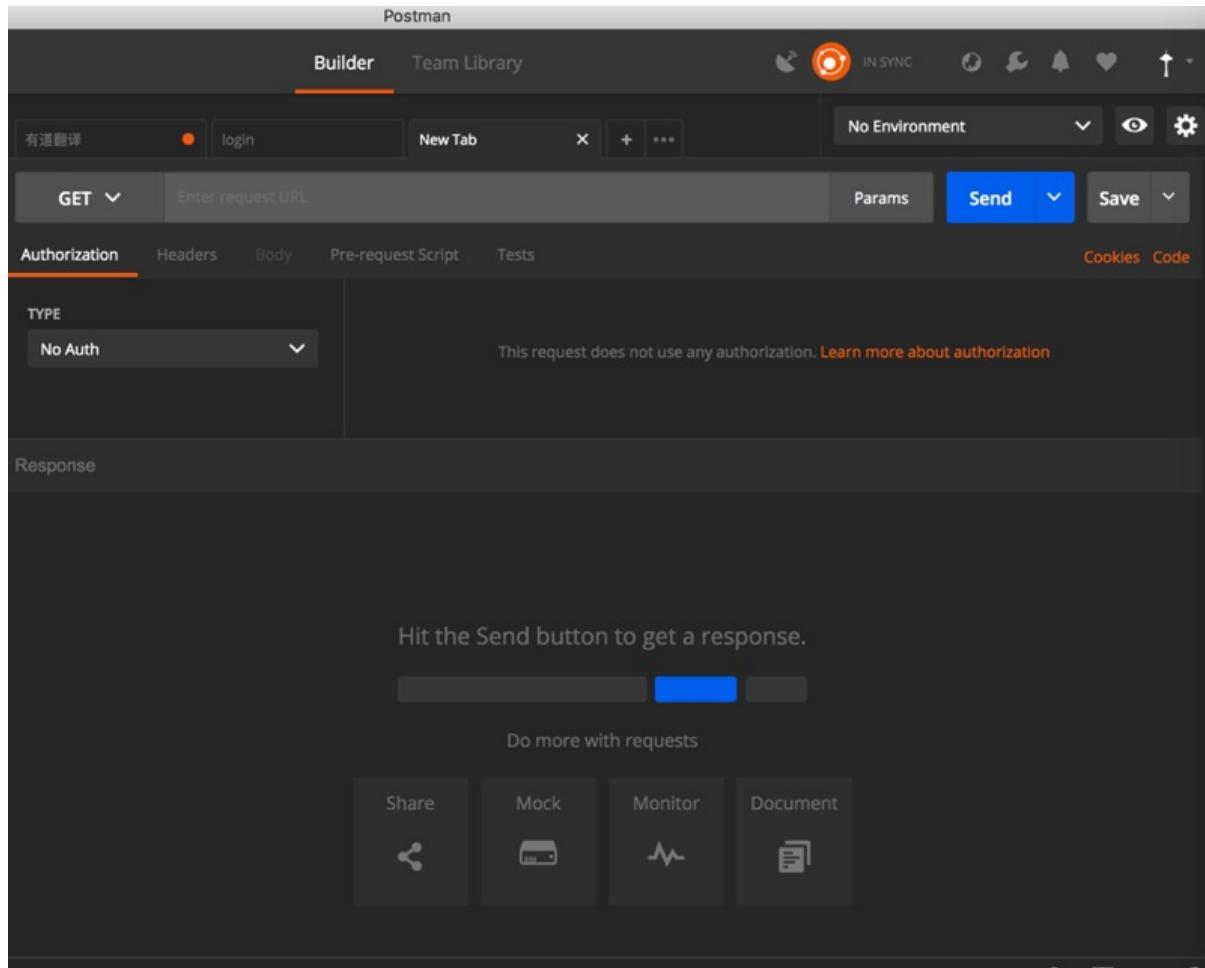
去新建接口，即对应的**Request**: New -> Request



或，在右边的Tab页面中点击加号+：

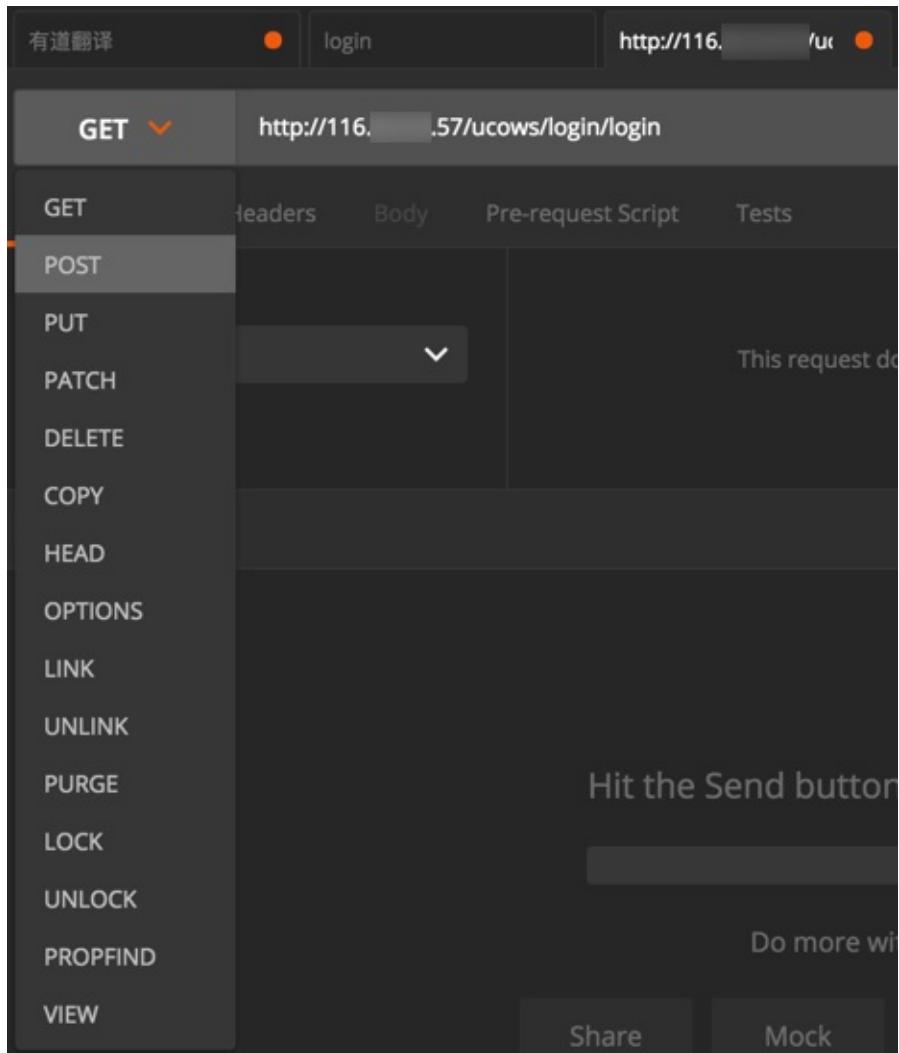


即可看到新建的Tab页：



然后：

- 设置HTTP的Method方法和输入api的地址



- 设置相关头信息

The screenshot shows the Postman Headers tab with the following configuration:

Key	Value
content	Content-MD5 Content-Length Content-Transfer-Encoding Content-Type

The "Content-Type" header is currently selected in the list.

The screenshot shows the Postman interface with a POST request to `http://116.57/ucows/login/login`. The 'Headers' tab is selected, showing a single header `Content-Type` set to `application/json`. A dropdown menu is open over the value field, listing various options: application/atom+xml, application/ecmascript, application/json (selected), application/javascript, application/octet-stream, application/ogg, and application/pdf.

- 设置相关GET或POST等的参数

The screenshot shows the Postman interface with a POST request to `http://116.57/ucows/login/login`. The 'Body' tab is selected, showing the raw JSON content:

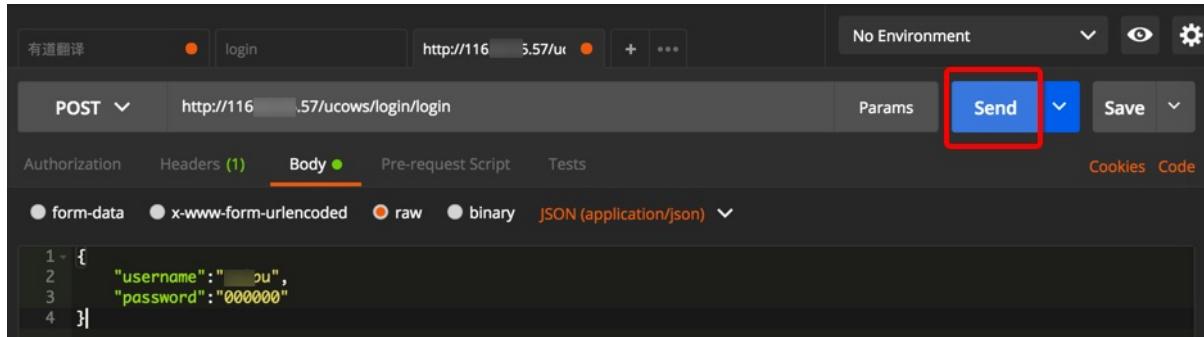
```

1 {
2   "username": "████pu",
3   "password": "000000"
4 }

```

A dropdown menu is open next to the `JSON (application/json)` label, listing other options: Text, Text (text/plain), JSON (application/json) (selected), Javascript (application/javascript), XML (application/xml), XML (text/xml), and HTML (text/html).

都填写好之后，点击Send去发送请求Request：



即可看到返回的响应Response的信息了：

The screenshot shows the Postman interface after sending the request. The response status is `200 OK`, time `107 ms`, and size `480 B`. The response body is displayed in JSON format:

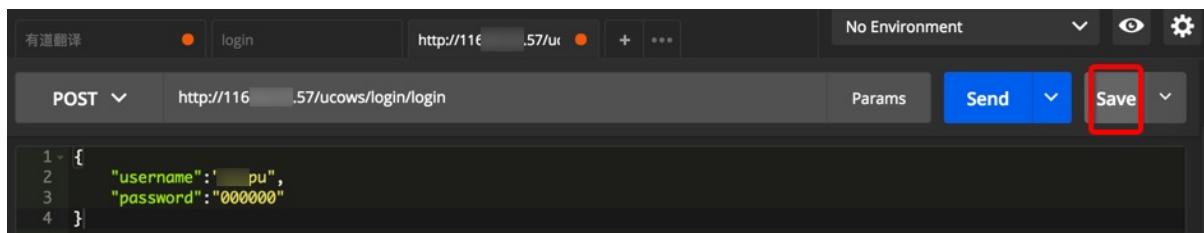
```

1 - {
2   "code": 200,
3   "message": "ok",
4   "data": {
5     "tokenid": "d4a638d",
6     "success": true,
7     "userId": "e5c7bf21",
8     "loginUserType": "Auto",
9     "cowfarmList": [
10       {
11         "name": "乳业",
12         "id": "1",
13         "farm_address": "廊坊市",
14         "email": "sina.com",
15         "link_man": "甫"
16       }
17     ]
18   }
19 }

```

然后可以重复上述修改Request的参数，点击Send去发送请求的过程，以便调试到API接口正常工作为止。

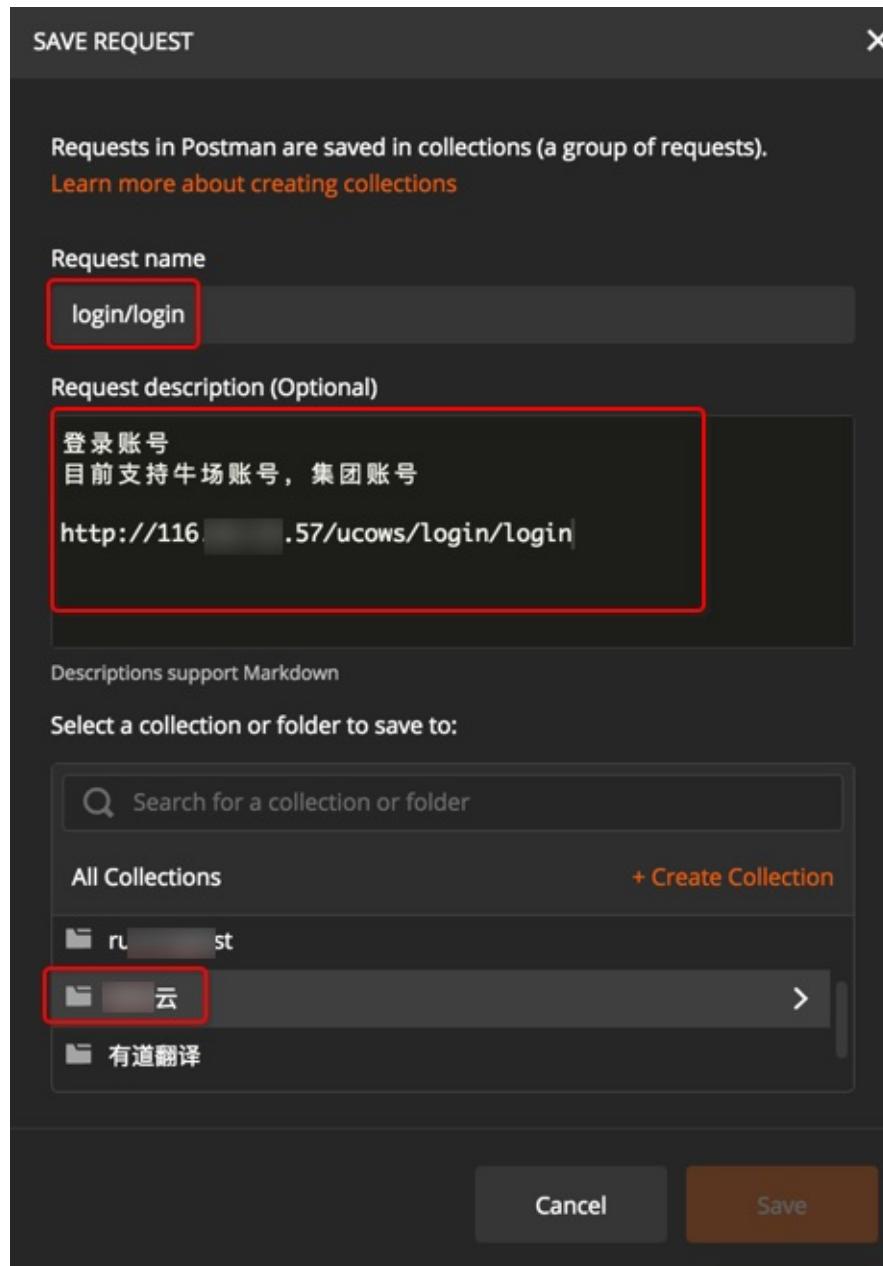
待整个接口都调试完毕后，记得点击Save去保存接口信息：



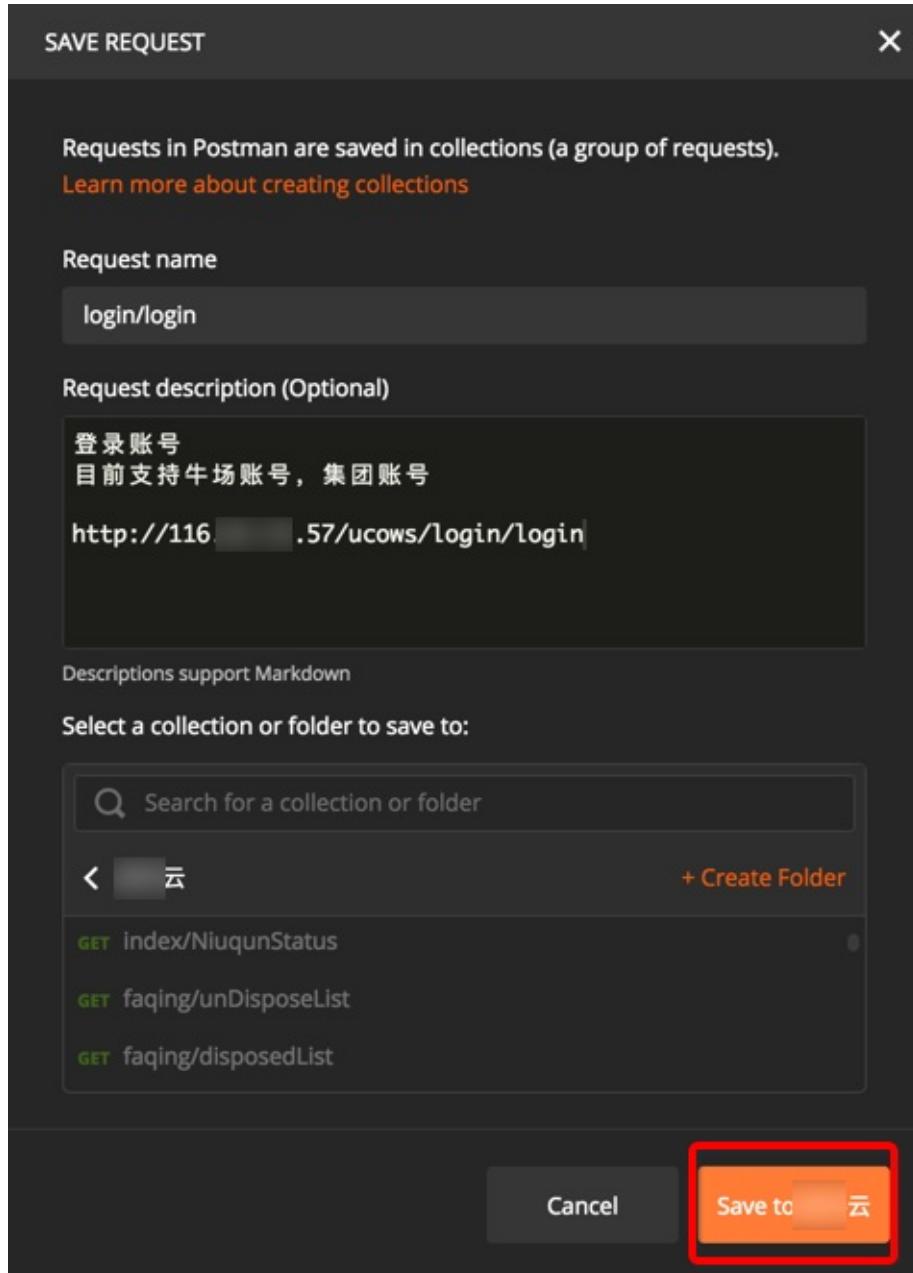
去保存当前API接口，然后需要填写相关的接口信息：

- Request Name: 请求的名字
 - 我一般习惯用保存为 接口的最后的字段名，比如 `http://{{server_address}}/ucows/login/login` 中的 `/login/login`
- Request Description: 接口的描述

- 可选 最好写上该接口的要实现的基本功能和相关注意事项
- 支持Markdown语法
- Select a collection or folder to save: 选择要保存到哪个分组（或文件夹）
 - 往往保存到某个API接口到所属的该项目名的分组

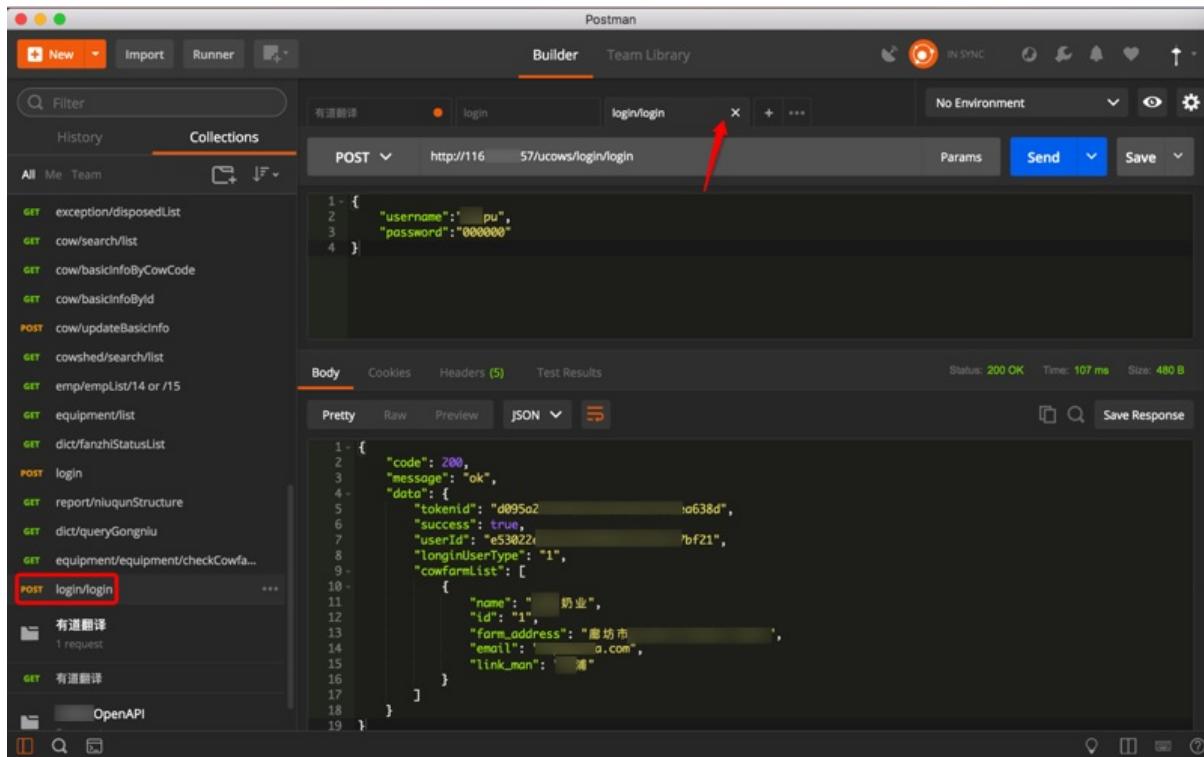


填写好内容，选择好分组，再点击保存：



此时，Tab的右上角的黄色点（表示没有保存）消失了，表示已保存。

且对应的分组中可以看到对应的接口了：



[warning] 默认不保存返回的Response数据

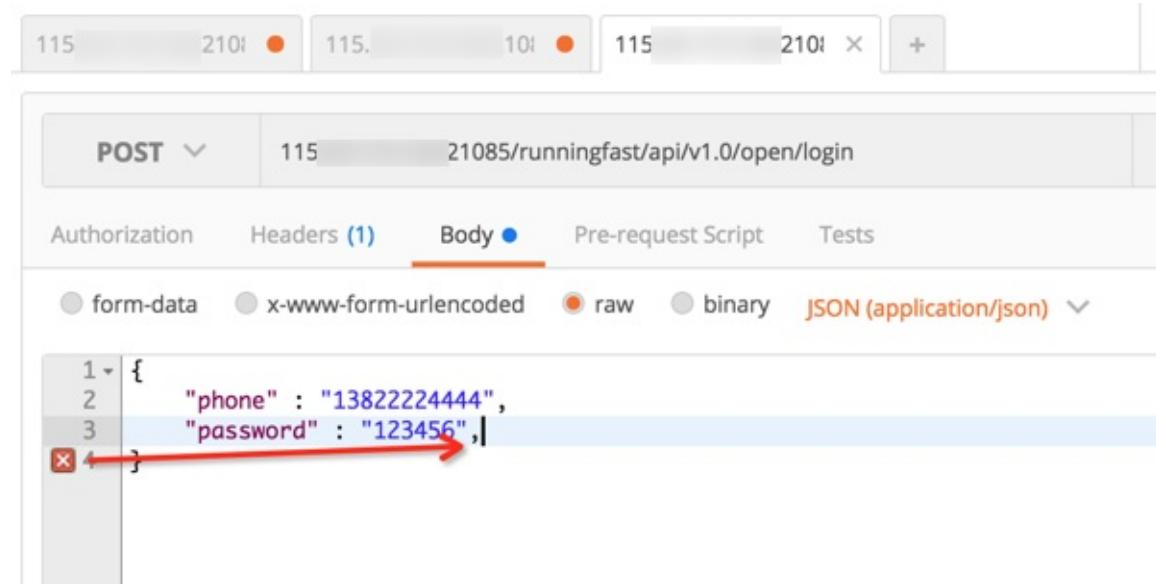
- 直接点击Save去保存，只能保存API本身（的Request请求），不会保存Response的数据
- 想要保存Response数据，需要用后面要介绍的 [多个Example](#)

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2017-12-29 15:20:23

JSON语法检查

在写POST时的Body中的JSON参数时，如果语法出错会智能提示。

比如json的值的行末多余逗号：



crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间： 2017-12-29 20:09:11

GET的Request的多参数

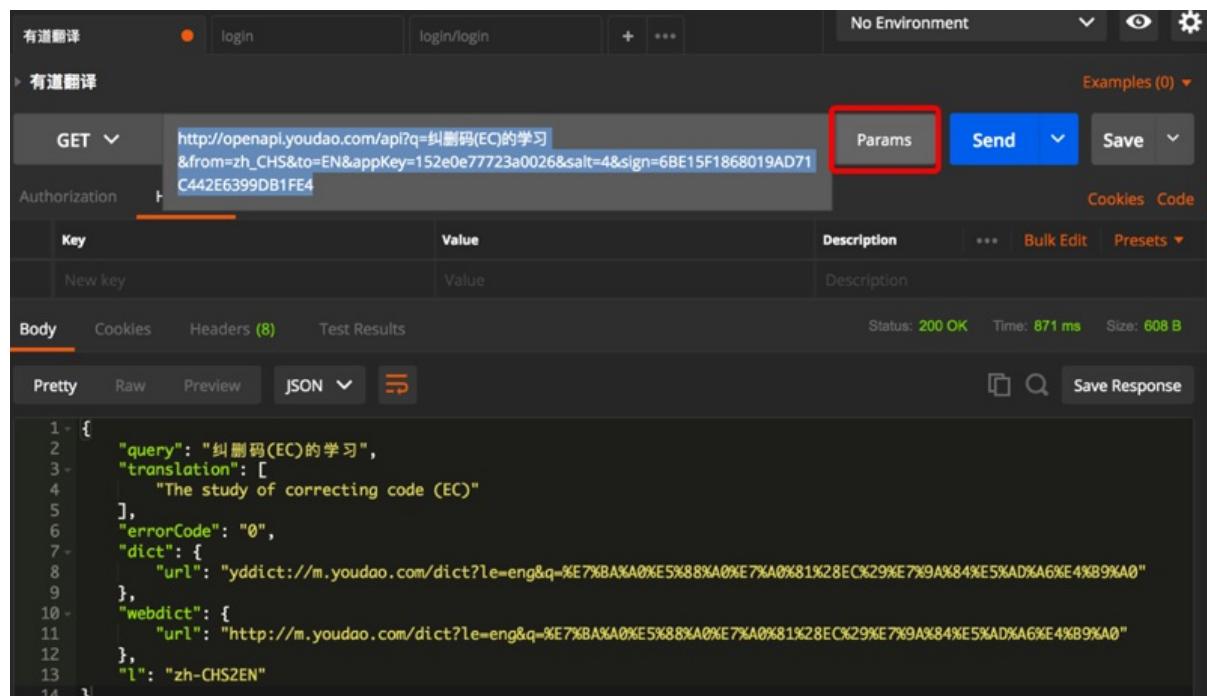
在GET请求中有多个参数需要处理的时候，Postman中有很多方便的手段去操作：

自动解析多个参数Params

比如，对于一个GET的请求的url是：
`http://openapi.youdao.com/api?q=纠删码(EC)的学习&from=zh_CHS&to=EN&appKey=152e0e77723a0026&salt=4&sign=6BE15F1868019AD71C442E6399DB1FE4`

对应着其实是 `?key=value` 形式中包含多个Http的GET的query string=query parameters

Postman可以自动帮我们解析出对应参数，可以点击Params：



The screenshot shows the Postman interface for a GET request to `http://openapi.youdao.com/api`. The URL bar contains the full URL with query parameters. The 'Params' button in the top right is highlighted with a red box. Below the URL bar, there's a table for managing parameters with columns for Key, Value, and Description. The 'Body' tab is selected, showing a JSON response structure. The response body is a JSON object with various fields like 'query', 'translation', 'errorCode', 'dict', 'url', 'webdict', and 'l'. The status bar at the bottom indicates a successful 200 OK response.

```

1 - {
2   "query": "纠删码(EC)的学习",
3   "translation": [
4     "The study of correcting code (EC)"
5   ],
6   "errorCode": "0",
7   "dict": {
8     "url": "yddict://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%A0%81%28EC%29%E7%9A%84%E5%AD%A6%E4%B9%A0"
9   },
10  "webdict": {
11    "url": "http://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%A0%81%28EC%29%E7%9A%84%E5%AD%A6%E4%B9%A0"
12  },
13  "l": "zh-CHS2EN"
14 }

```

看到展开的多个参数：

The screenshot shows the Postman interface with a red box highlighting the 'Params' table. The table contains the following data:

Key	Value	Description
<input checked="" type="checkbox"/> q	纠删码(EC)的学习	
<input checked="" type="checkbox"/> from	zh_CHS	
<input checked="" type="checkbox"/> to	EN	
<input checked="" type="checkbox"/> appKey	152e0e77723a0026	
<input checked="" type="checkbox"/> salt	4	
<input checked="" type="checkbox"/> sign	6BE15F1868019AD71C442E6399DB1FE4	

Below the table, the 'Body' tab is selected, showing the JSON response:

```

1 - {
2   "query": "纠删码(EC)的学习",
3   "translation": [
4     "The study of correcting code (EC)"
5   ],
6   "errorCode": "0",

```

如此就可以很方便的修改，增删对应的参数了。

不勾选某些参数达到临时禁用的效果

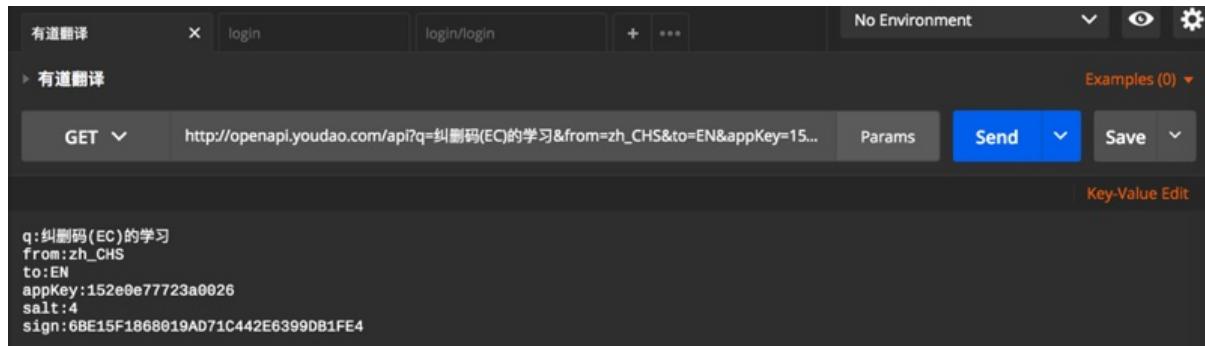
且还支持，在不删除某参数的情况下，如果想要暂时不传参数，可以方便的通过不勾选的方式去实现：

The screenshot shows the Postman interface with the 'Params' table. The table contains the following data:

Key	Value	Description
<input type="checkbox"/> q	纠删码(EC)的学习	
<input checked="" type="checkbox"/> from	zh_CHS	
<input type="checkbox"/> to	EN	
<input checked="" type="checkbox"/> appKey	152e0e77723a0026	
<input checked="" type="checkbox"/> salt	4	
<input checked="" type="checkbox"/> sign	6BE15F1868019AD71C442E6399DB1FE4	

批量编辑GET的多个参数

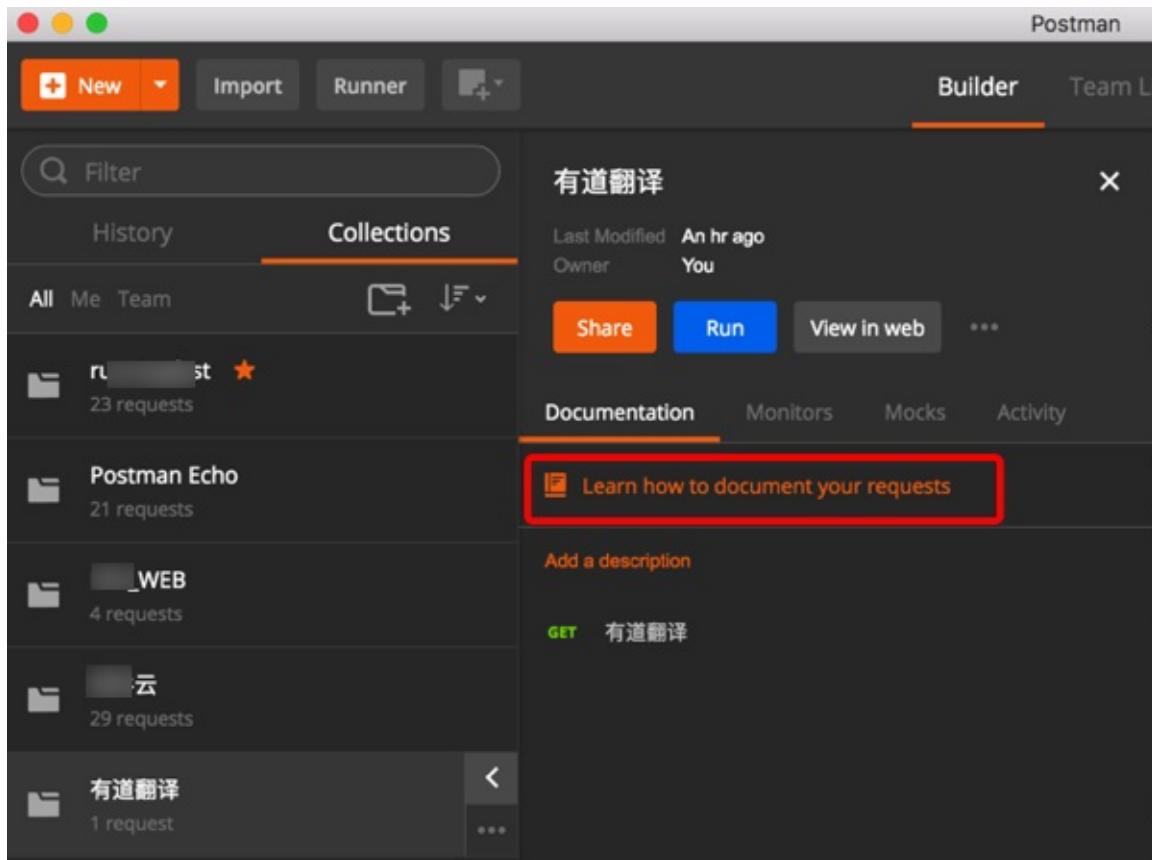
当然，如果想要批量的编辑参数，可以点击右上角的**Bulk Edit**，去实现批量编辑。



crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2017-12-29 20:19:54

给接口添加描述

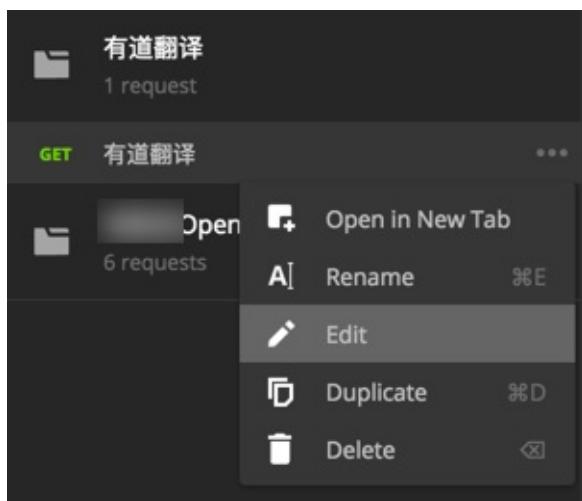
通过看到：



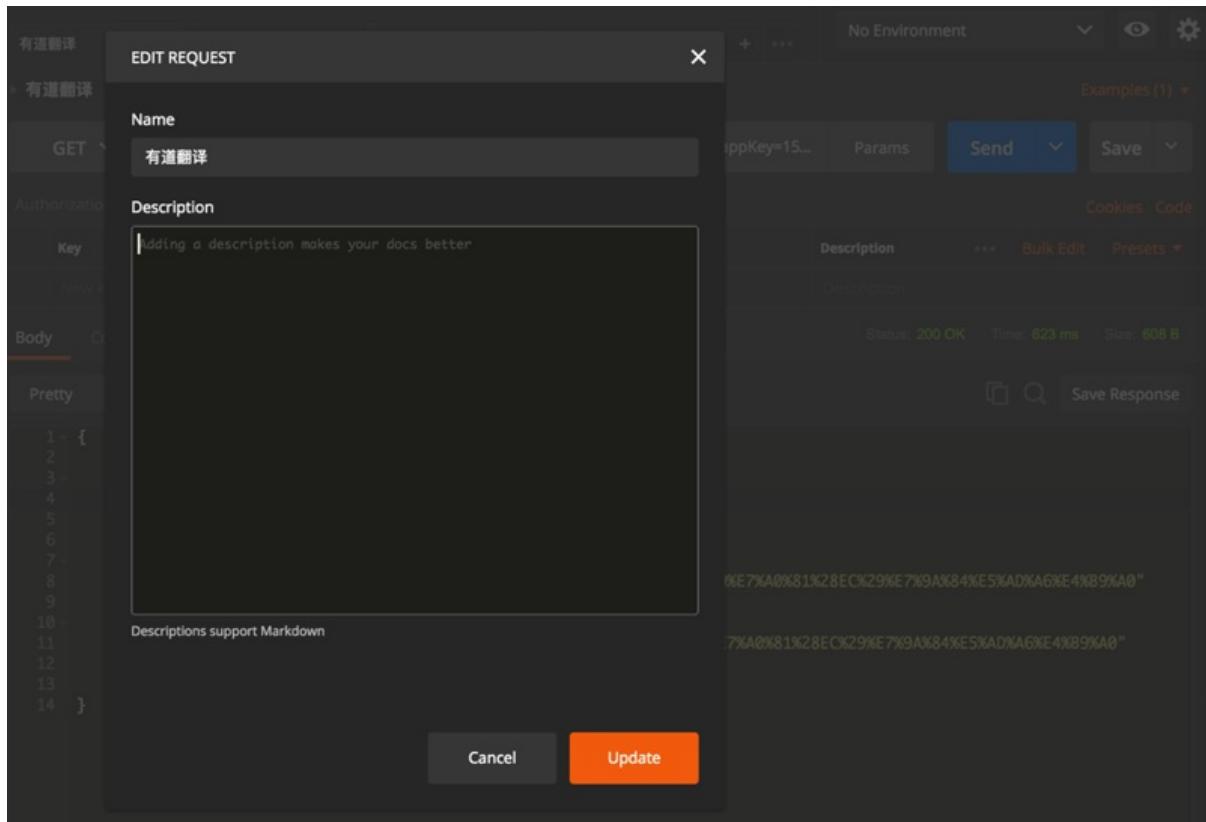
[Intro to API documentation](#)

得知，API的描述中，也支持Markdown的。

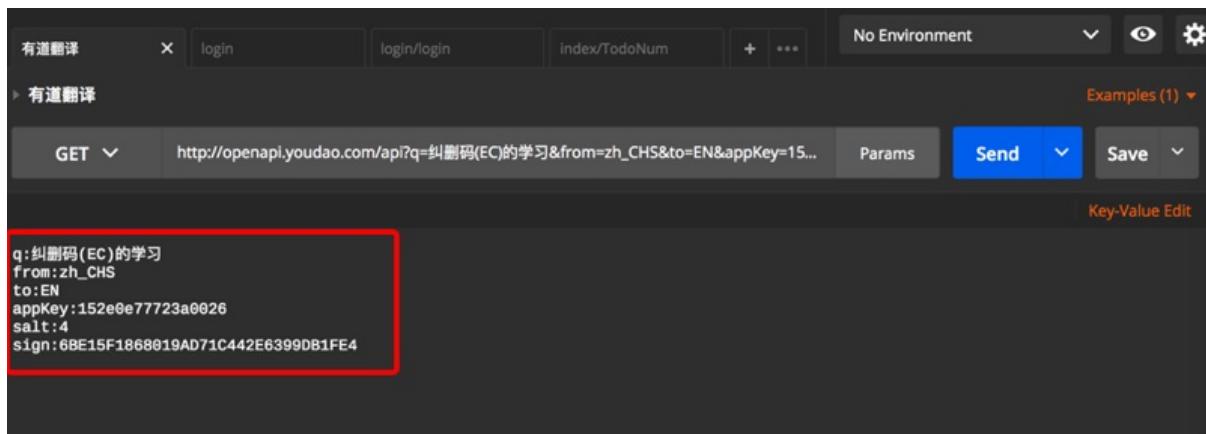
所以，可以很方便的添加有条理的接口描述，尤其是参数解释了：



可以看到 Descriptions support Markdown



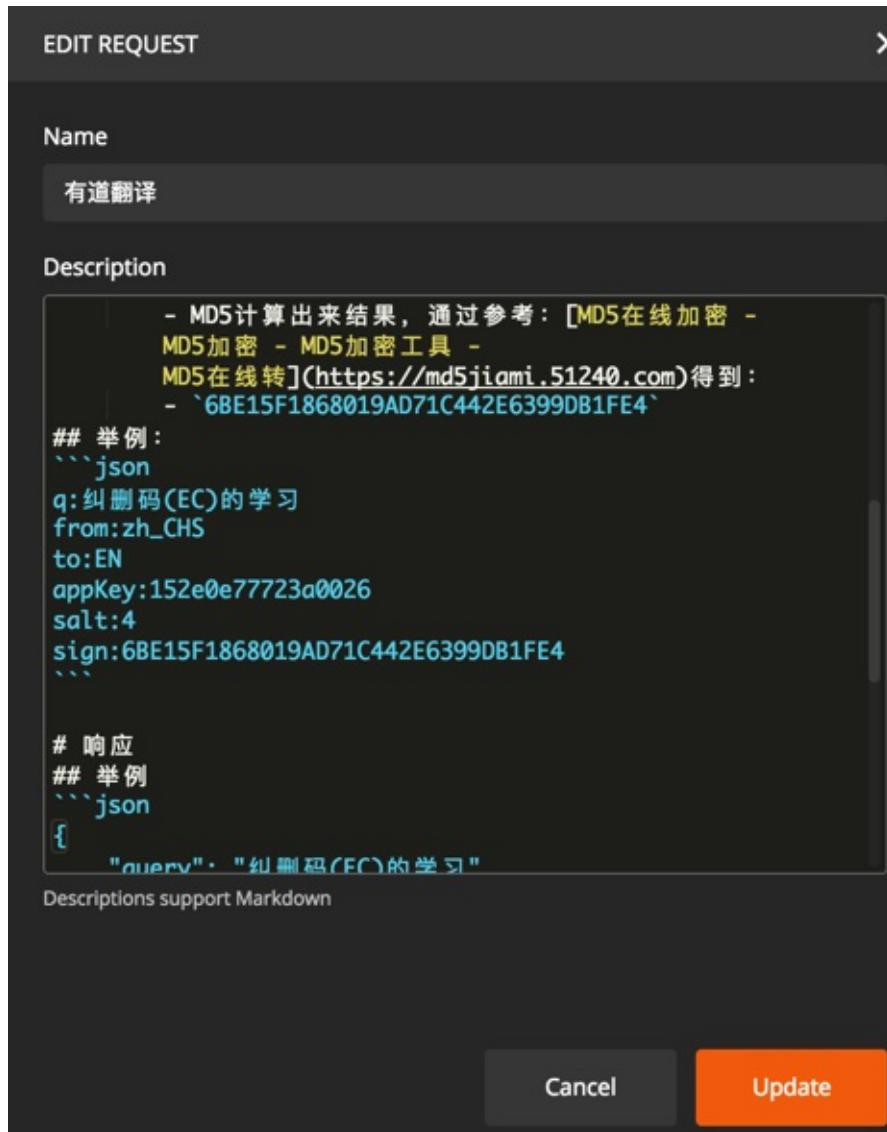
而对于要解释的参数，可以通过之前的 Param -> Bulk Edit 的内容：



拷贝过来，再继续去编辑：



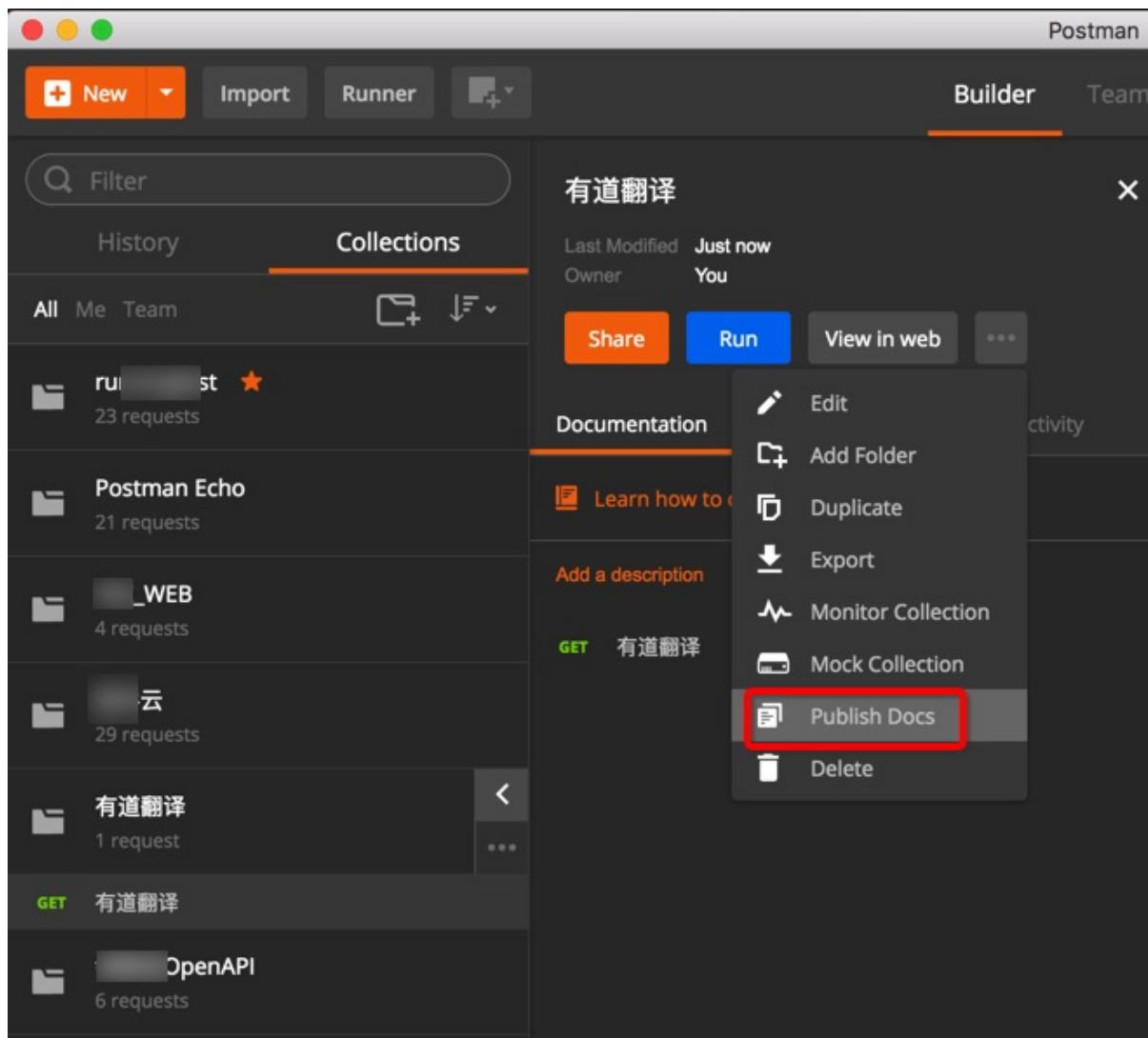
以及添加更多解释信息：



点击Update后，即可保存。

发布后带Markdown描述的API的效果

去发布后：



对应的效果: [有道翻译](#)

The screenshot shows the POSTMAN application interface. At the top, there are several tabs: Mock resp, 奶牛云, Supercharge..., collection-ru..., 奶牛云, Documenter, Intro to API d..., 有道翻译, Documenter, and 有道翻译. The main window title is "documenter.getpostman.com/view/669382/collection/77fd4ek". The top right corner shows "Public", "Run in Postman", "No environment", and a dropdown for "李茂 (cifan)".
The left sidebar has a "POSTMAN" logo and sections for "有道翻译", "Introduction", and "GET 有道翻译".
The main content area has a title "有道翻译" and a subtitle "GET 有道翻译". Below it is a sample request URL:
`http://openapi.youdao.com/api?q=纠删码(EC)的学习&from=zh_CHS&to=EN&appKey=152e0e77723a0026&salt=4&sign=6BE15F1868019AD71C442E6399DB1FE4`
A red box highlights the "请求参数" section, which contains the following text:

调用 有道 的API实现将 中文 翻译为 英文

请求 **markdow**
n的描述

请求参数

- **q** : query string, 查询字符串
- **from** : 从什么语言, 原始语言
- **to** : 翻译为, 目标语言
- **appKey** :自己 有道智云 账号中创建的应用的 应用ID
- **salt** :给MD5算法加的 盐 , 一般用任意随机数即可
- **sign** :MD5加密后的值=md5(appKey+q+salt+密钥)
 - 经过验证, 好像大小写均可
 - 举例:
 - appKey=应用ID=152e0e77723a0026
 - q=纠删码(EC)的学习

The right side shows a "Sample Request" section with a curl command and a "Sample Response" section displaying a JSON object:

```
curl --request GET \
--url "http://openapi.youdao.com/api?q=%E7%BA%A0%E5%88%A0%E7%A0%81&from=zh_CHS&to=EN&appKey=152e0e77723a0026&salt=4&sign=6BE15F1868019AD71C442E6399DB1FE4"
```

{
 "query": "纠删码(EC)的学习",
 "translation": [
 "The study of correcting code (EC)"
],
 "errorCode": "0",
 "dict": {
 "url": "yddict://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%A0%81&from=zh_CHS&to=EN&appKey=152e0e77723a0026&salt=4&sign=6BE15F1868019AD71C442E6399DB1FE4"
 },
 "sign": "6BE15F1868019AD71C442E6399DB1FE4"
}

example
示例

有道翻译

Introduction

GET 有道翻译

```
q:纠删码(EC)的学习
from:zh_CHS
to:EN
appKey:152e0e77723a0026
salt:4
sign:6BE15F1868019AD71C442E6399DB1FE4
```

响应

举例

```
{
    "query": "纠删码(EC)的学习",
    "translation": [
        "The study of correcting code (EC)"
    ],
    "errorCode": "0",
    "dict": {
        "url": "yddict://m.youdao.com/dict?le=eng&q=%E7%AE%A9%E6%8B%9D"
    },
    "webdict": {}Click to Expand
```

PARAMS

q	纠删码(EC)的学习
from	zh_CHS
to	EN
appKey	152e0e77723a0026
salt	4
sign	6BE15F1868019AD71C442E6399DB1FE4

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 11:01:53

Postman功能：Response

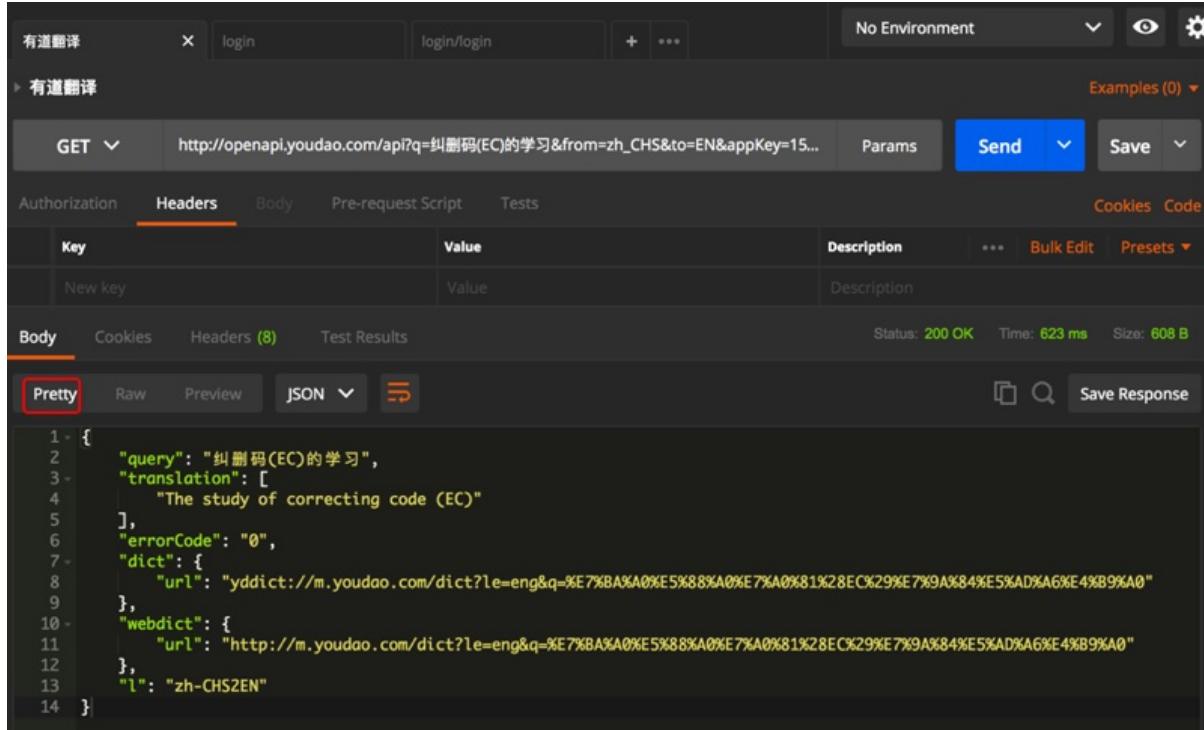
Postman中对于Response响应，也有很多方便好用的功能。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2017-12-29 20:20:38

Response数据显示模式

Postman对于返回的Response数据，支持三种显示模式：

默认 格式化后的Pretty模式



The screenshot shows the Postman interface with a request to `http://openapi.youdao.com/api?q=纠删码(EC)的学习&from=zh_CHS&to=EN&appKey=15...`. The response body is displayed in Pretty mode, showing a JSON object with various fields like query, translation, errorCode, dict, and webdict. The JSON is formatted with indentation and line breaks for readability.

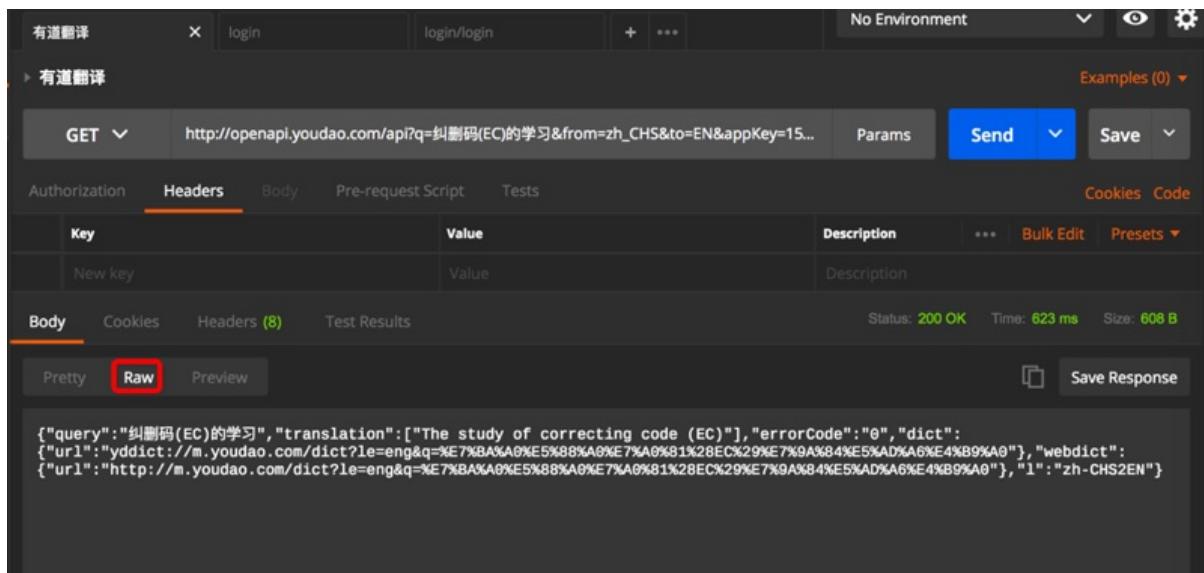
```

1 - {
2   "query": "纠删码(EC)的学习",
3   "translation": [
4     "The study of correcting code (EC)"
5   ],
6   "errorCode": "0",
7   "dict": {
8     "url": "yddict://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%A0%81%28EC%29%E7%9A%84%E5%AD%A6%E4%B9%A0"
9   },
10  "webdict": {
11    "url": "http://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%A0%81%28EC%29%E7%9A%84%E5%AD%A6%E4%B9%A0"
12  },
13  "l": "zh-CHS2EN"
14 }

```

Raw原始模式

点击Raw，可以查看到返回的没有格式化之前的原始数据：

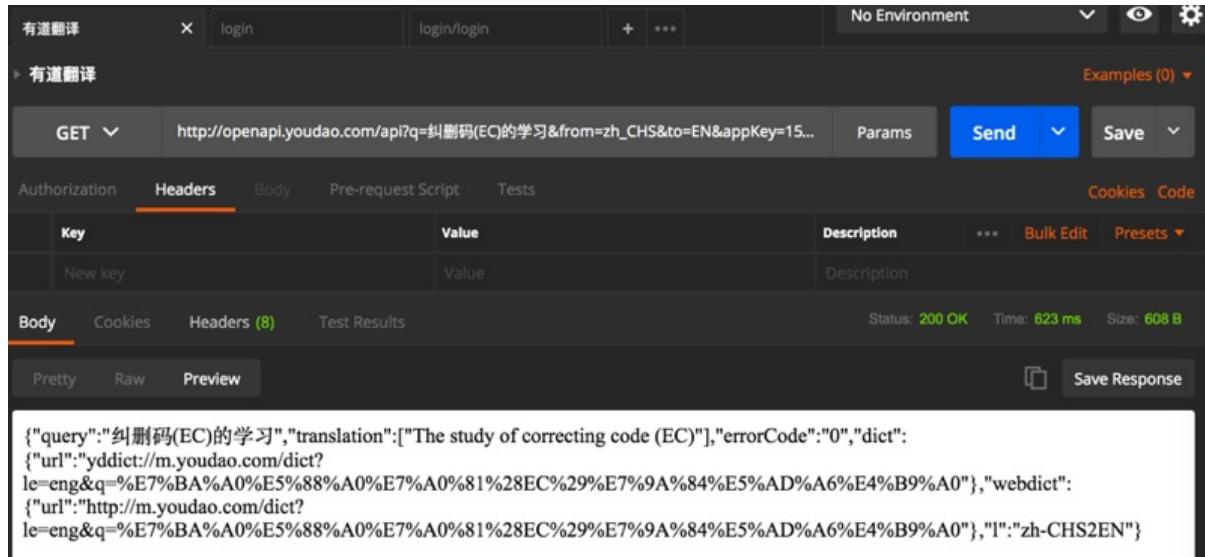


The screenshot shows the Postman interface with the same request and response as the previous screenshot, but the response body is displayed in Raw mode. The JSON output is shown as a single, unindented block of text,失去了可读性。

```
{"query":"纠删码(EC)的学习","translation":["The study of correcting code (EC)"],"errorCode":"0","dict":{"url":"yddict://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%A0%81%28EC%29%E7%9A%84%E5%AD%A6%E4%B9%A0"},"webdict":{"url":"http://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%A0%81%28EC%29%E7%9A%84%E5%AD%A6%E4%B9%A0"},"l":"zh-CHS2EN"}
```

Preview预览模式

以及Preview，是对应Raw原始格式的预览模式：



The screenshot shows the Postman interface with a request to `http://openapi.youdao.com/api?q=纠删码(EC)的学习&from=zh_CHS&to=EN&appKey=15...`. The Headers tab is selected, and the Body tab is selected under the Preview section. The response body is displayed as:

```
{"query":"纠删码(EC)的学习","translation":["The study of correcting code (EC)"],"errorCode":"0","dict":{"url":"yddict./m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%A0%81%28EC%29%E7%9A%84%E5%AD%A6%E4%B9%A0"},"webdict":{"url":"http://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%A0%81%28EC%29%E7%9A%84%E5%AD%A6%E4%B9%A0"},"l":"zh-CHS2EN"}
```

Preview这种模式的显示效果，好像是对于返回的是html页面这类，才比较有效果。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间： 2017-12-29 20:26:35

Response其他功能

Response的Cookies

很多时候普通的API调用，倒是没有Cookie的：

The screenshot shows the Postman application interface. At the top, there's a header bar with tabs for 'login' and 'login/login'. Below the header, the URL is set to `http://openapi.youdao.com/api?q=纠删码(EC)的学习&from=zh_CHS&to=EN&appKey=15...`. The main workspace has tabs for 'Body', 'Headers (8)', and 'Test Results', with 'Headers' currently selected. In the 'Headers' tab, there's a table with one row labeled 'New key' under the 'Key' column and 'Value' under the 'Value' column. To the right of the table, there are buttons for 'Description', '...', 'Bulk Edit', and 'Presets'. Below the table, the status bar shows 'Status: 200 OK', 'Time: 623 ms', and 'Size: 608 B'. At the bottom of the workspace, there's a large image of a cookie and the text 'No cookie for you'. A note below the image says 'No cookies were returned by the server'.

Response的Headers头信息

举例，此处返回的是有Headers头信息的：

The screenshot shows a Postman interface with the following details:

- Request URL:** http://openapi.youdao.com/api?q=纠删码(EC)的学习&from=zh_CHS&to=EN&appKey=15...
- Method:** GET
- Headers:** (8)
- Body:** (raw) JSON
- Test Results:** Status: 200 OK, Time: 623 ms, Size: 608 B

The Headers section lists the following responses:

Key	Value	Description
Connection	keep-alive	
Content-Encoding	gzip	
Content-Type	application/json;charset=UTF-8	
Date	Thu, 02 Nov 2017 02:23:06 GMT	
Server	nginx	
Transfer-Encoding	chunked	
Vary	Accept-Encoding	
X-Application-Context	application:8686	

可以从中看到服务器是Nginx的。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2017-12-29 20:30:27

保存多个Example

之前想要实现，让导出的API文档中能看到接口返回的Response数据。后来发现是Example这个功能去实现此效果的。

如何添加Example

The screenshot shows the Postman application interface. In the top right corner, there is a red box around the 'Examples (0)' button. On the right side of the screen, a sidebar titled 'Examples (0)' is open, containing the message 'No examples added' and a link 'Save responses and associated requests as Examples. Learn More'. Below this, there is an 'Add Example' button, which is also highlighted with a red box. The main workspace shows a GET request to 'http://openapi.youdao.com/api?q=纠删码(EC)的学习&from=zh_CHS&to=EN&appKey=15...'. The 'Body' tab is selected, displaying a JSON response:

```
1 - {
2   "query": "纠删码(EC)的学习",
3   "translation": [
4     "The study of correcting code (EC)"
5   ],
6   "errorCode": "0",
7   "dict": {
8     "url": "yddict://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%A0%81%28EC%29%E7%9A%84%E5%AD%A6%E4%B9%A0"
9   },
10  "webdict": {
11    "url": "http://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%A0%81%28EC%29%E7%9A%84%E5%AD%A6%E4%B9%A0"
12  },
13  "l": "zh-CHS2EN"
14 }
```

继续点击Save Example：

The screenshot shows the Postman interface with a saved example. The top right corner has a red box around the "Save Example" button. Below it, the "EXAMPLE REQUEST" section shows a GET request to [http://openapi.youdao.com/api?q=纠删码\(EC\)的学习&from=zh_CHS&to=EN&appKey=152e0e77723a0026&salt=4&sig=...](http://openapi.youdao.com/api?q=纠删码(EC)的学习&from=zh_CHS&to=EN&appKey=152e0e77723a0026&salt=4&sig=...). The "Headers" tab is selected, showing a single header "New key" with value "Value". The "EXAMPLE RESPONSE" section shows a 200 OK status with a JSON response body. The response body is a JSON object with fields like "query", "translation", "errorCode", "dict", and "webdict". A red arrow points to the "Examples (1)" dropdown in the top right.

```

1 - {
2 -   "query": "纠删码(EC)的学习",
3 -   "translation": [
4 -     "The study of correcting code (EC)"
5 -   ],
6 -   "errorCode": "0",
7 -   "dict": {
8 -     "url": "yddict://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%AA%81%28EC%29%E7%9A%84%E5%AD%A6%E4%89%A0"
9 -   },
10 -  "webdict": {
11 -    "url": "http://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%AA%81%28EC%29%E7%9A%84%E5%AD%A6%E4%89%A0"
12 -  },
13 -  "l": "zh-CHS2EN"
14 - }

```

保存后，就能看到Example(1)了：

The screenshot shows the Postman interface with the "Examples (1)" dropdown open. A red arrow points to the dropdown. The "Body" tab is selected, showing the same JSON response as before. The "Headers" tab is also visible. The "Status" bar at the bottom shows 200 OK, 623 ms, and 608 B.

```

1 - {
2 -   "query": "纠删码(EC)的学习",
3 -   "translation": [
4 -     "The study of correcting code (EC)"
5 -   ],
6 -   "errorCode": "0",
7 -   "dict": {
8 -     "url": "yddict://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%AA%81%28EC%29%E7%9A%84%E5%AD%A6%E4%89%A0"
9 -   },
10 -  "webdict": {
11 -    "url": "http://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%AA%81%28EC%29%E7%9A%84%E5%AD%A6%E4%89%A0"
12 -  },
13 -  "l": "zh-CHS2EN"
14 - }

```

单个Example在导出的API文档中的效果

然后再去导出文档，导出文档中的确能看到返回数据的例子：

The screenshot shows the Youdao Translation API documentation on the left and its export to Swagger UI on the right.

Youdao Translation API Documentation:

- Introduction:** A brief introduction to the API.
- GET 有道翻译:** A sample request for the translation endpoint.
- 请求 (Request):** A section detailing the required parameters:
 - 请求参数 (Request Parameters):**
 - q: query string, 查询字符串
 - from: 从什么语言, 原始语言
 - to: 翻译为, 目标语言
 - appKey: 自己 有道智云 账号中创建的应用的应用ID
 - salt: 给MD5算法加的 盐 , 一般用任意随机数即可
 - sign: MD5加密后的值=md5(appKey+q+salt+密钥)
 - 经过验证, 好像大小写均可
 - 举例:
 - appKey=应用ID=152e0e77723a0026
 - q=纠正码(EC)的学习
 - salt=4
 - 密钥=应用密钥
 - =sYmnnOaisQgZzrlrBFozWAtsaRyyJg4N

Swagger UI Export:

- Sample Request:** Shows the Node.js code for making a GET request to the Youdao API.
- Sample Response:** Shows the JSON response from the API, including the query, translation, error code, and URL.

多个Example在导出的API文档中的效果

The screenshot shows the NiuNiuCloud API documentation on the left and its export to Swagger UI on the right.

NiuNiuCloud API Documentation:

- Introduction:** A brief introduction to the API.
- POST login/login:** A sample request for the login endpoint.
- HEADERS:** Shows the Content-Type header set to application/json.
- BODY:** Shows the JSON body for the login request.

Swagger UI Export:

- Sample Request:** Shows the Node.js code for making a POST request to the login endpoint.
- Sample Response:** Shows the JSON response from the API, including the code, message, data (with a token ID), success status, user ID, and login user type.

The screenshot shows the Postman application interface. On the left, there's a sidebar titled "Introduction" with a list of API endpoints. The main area shows a "POST login/login" request. The "Sample Request" tab displays a Node.js script to make a POST request to "http://116.57/uows/login/login". The "Sample Response" tab shows a JSON object with fields like "code", "message", "data", and "tokenid". A red box highlights the "tokenid" field in the response.

奶牛云

Introduction

POST login/login

GET index/CurMonthTodoNum

GET index/NiuqunStatus

GET faqing/unDisposeList

GET faqing/disposedList

GET peizhong/unDisposeList

GET peizhong/disposedList

GET peizhong/basicInfo

POST peizhong/update

GET yunjian/chujianUnDisposeList

GET yunjian/fujianUnDisposeList

GET yunjian/disposedList

GET yunjian/chujianBasicInfo

GET yunjian/fujianBasicInfo

GET exception/unDisposeList

GET exception/disposedList

GET cow/search/list

POST login/login

登录账号 目前支持牛场账号，集团账号
http://116.57/uows/login/login

HEADERS

Content-Type application/json

BODY

```
{  
    "username": "testman",  
    "password": "000000"  
}
```

Sample Request

```
I ihui
```

```
var http = require("http");  
  
var options = {  
    "method": "POST",  
    "hostname": [  
        "{{server_address}}"  
    ],  
    "path": [  
        "uows",  
        "login",  
        "login"  
    ]  
};
```

Copy to Clipboard

Sample Response

```
{  
    "code": 200,  
    "message": "ok",  
    "data": {  
        "tokenid": "40a0bf1caf8247ebbac16ddc7942cbeb",  
        "success": true,  
        "userId": "B61cf863d454471b7e75af69e018794",  
        "loginUserType": "1"  
    }  
}
```

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2017-12-29 21:38:10

Postman功能：其他工具和功能

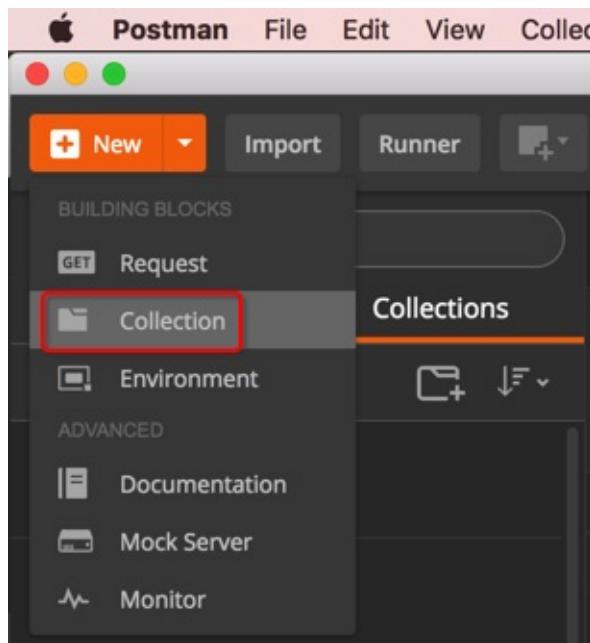
此处接着介绍Postman的其他一些功能或有用的工具。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2017-12-29 13:51:38

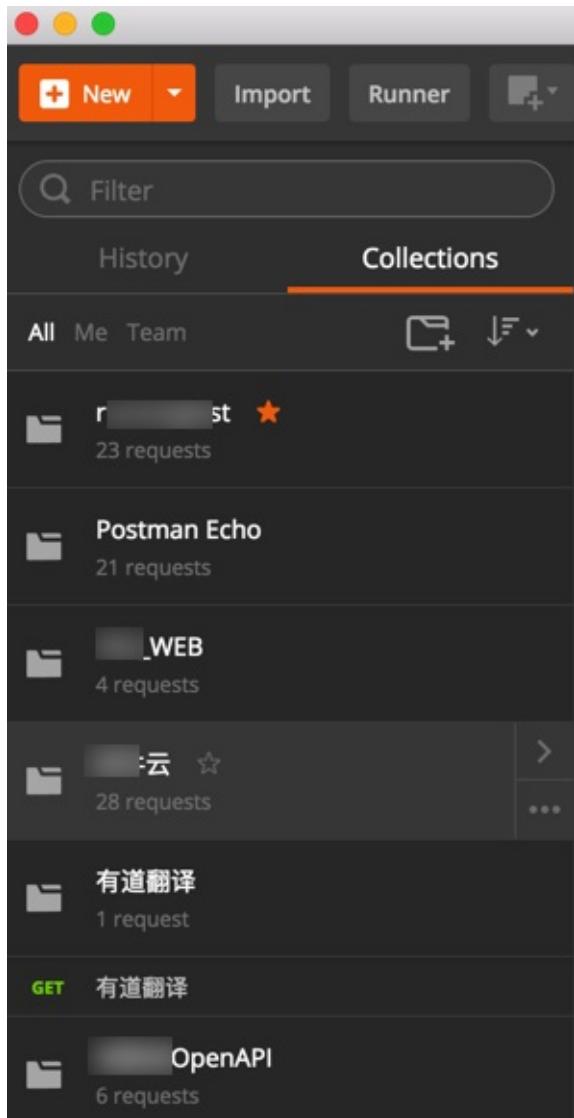
分组Collection

在刚开始一个项目时，为了后续便于组织和管理，把同属该项目的多个API，放在一组里

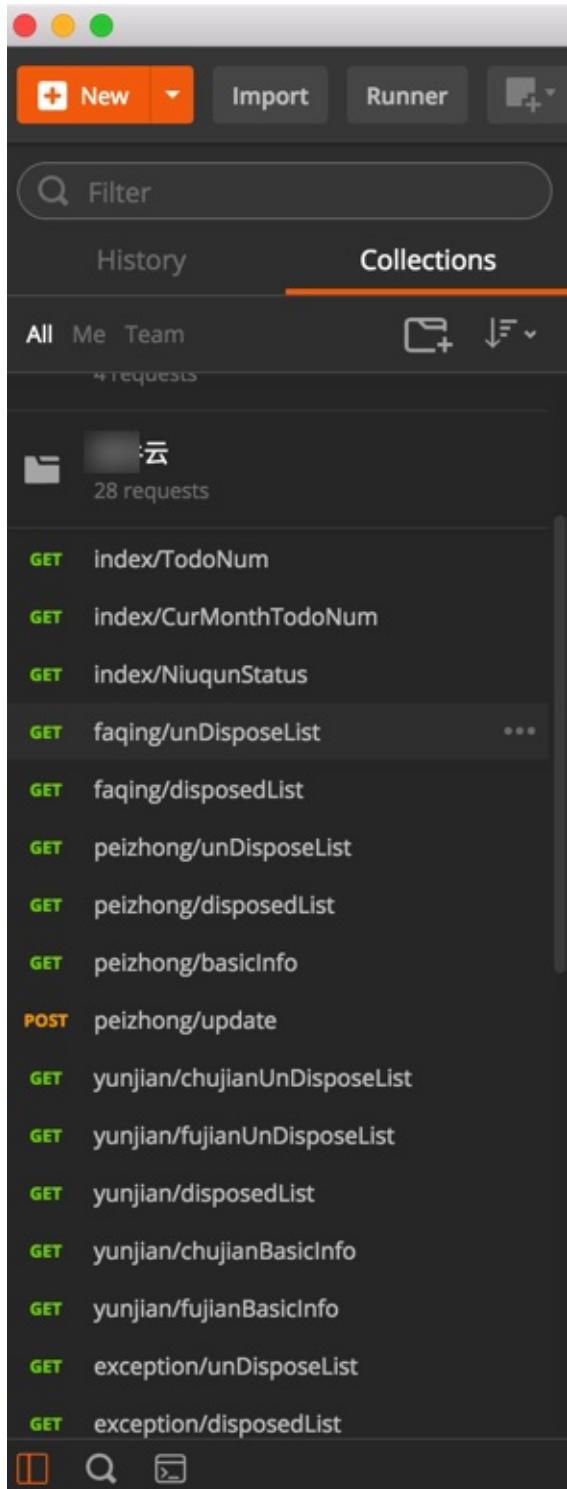
所以要先去新建一个Collection: New -> Collection



使用了段时间后，建了多个分组的效果：



单个分组展开后的效果：



crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2017-12-29 13:56:46

历史记录History

Postman支持history历史记录，显示出最近使用过的API：

The screenshot shows the Postman interface with the 'History' tab selected in the left sidebar. A red arrow points to the 'History' tab. The main workspace displays a recent API request for a POST method to '21084/runningfast/api/v1.0/open/register'. The 'Body' tab is selected, showing JSON input:

```
1 - {  
2   "phone": "13811118888",  
3   "smsCode": "505033",  
4   "email": "register",  
5   "firstName": "crifan",  
6   "lastName": "Li",  
7   "password": "123456",  
8   "facebookUserId": "11 56"  
9 }
```

Below the body, the response status is 200 OK and the time is 926 ms. The response body is:

```
1 - {  
2   "code": 10303,  
3   "message": "sms code is wrong"  
4 }
```

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间： 2017-12-29 15:22:24

用环境变量实现多服务器版本

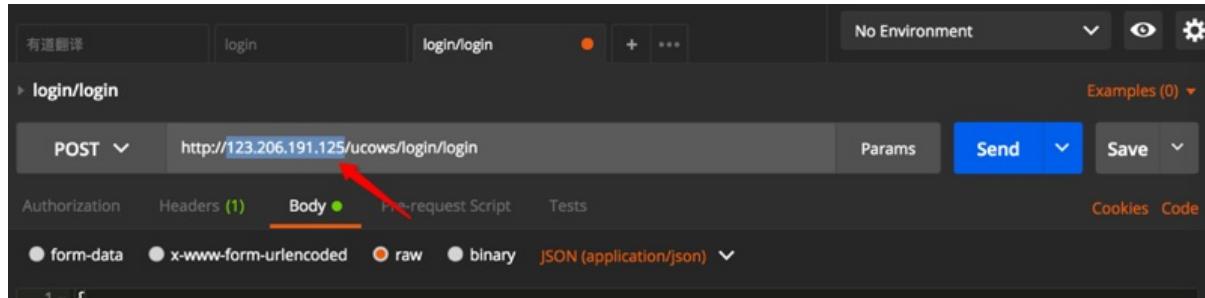
现存问题

在测试API期间，往往存在多种环境，对应IP地址（或域名也不同）

比如：

- **Prod:** `http://116.62.25.57/ucows`
 - 用于开发完成发布到生产环境
- **Dev:** `http://123.206.191.125/ucows`
 - 用于开发期间的线上的Development的测试环境
- **LocalTest:** `http://192.168.0.140:80/ucows`
 - 用于开发期间配合后台开发人员的本地局域网内的本地环境，用于联合调试API接口

而在测试API期间，往往需要手动去修改API的地址：



效率比较低，且地址更换后之前地址就没法保留了。

另外，且根据不同IP地址（或者域名）也不容易识别是哪套环境。

解决办法

小幺鸡的线上环境和本机环境的切换

之前得知[小幺鸡，简单好用的接口文档管理工具 -》发送JSON-演示项目](#) 中有个好用的功能：

支持不同环境：

- 线上环境
- 本地环境

等，当时以为Postman不支持呢

Postman支持用Environment环境变量去实现多服务器版本

后来发现Postman中，有Environment和Global Variable，用于解决这个问题，实现不同环境的管理：

The screenshot shows the Postman interface with a request to `http://openapi.youdao.com/api?q=纠删码(EC)的学习&from=zh_CHS&to=en`. The 'Environment' dropdown is set to 'No Environment'. A red box highlights the 'Environment' button in the top right. The 'Headers' tab is selected, and a red box highlights the 'Globals' button in the top right of the header section. The 'Body' tab is selected, showing a JSON response with a red box highlighting the 'Pretty' button.

```

1 {
2   "query": "纠删码(EC)的学习",
3   "translation": [
4     "The study of correcting code (EC)"
5   ],
6   "errorCode": "0",
7   "dict": {
8     "url": "yddict://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%A0%81%28ECK29%E7%9A%84%E5%AD%A6%E4%B9%A0"
9   },
10  "webdict": {
11    "url": "http://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%A0%81%28ECK29%E7%9A%84%E5%AD%A6%E4%B9%A0"
12  },
13  "l": "zh-CHS2EN"
14 }

```

-》很明显，就可以用来实现不用手动修改url中的服务器地址，从而动态的实现，支持不同服务器环境：

- Production 生产环境
- Development 开发环境
- Local 本地局域网环境

如何使用Environment实现多服务器版本

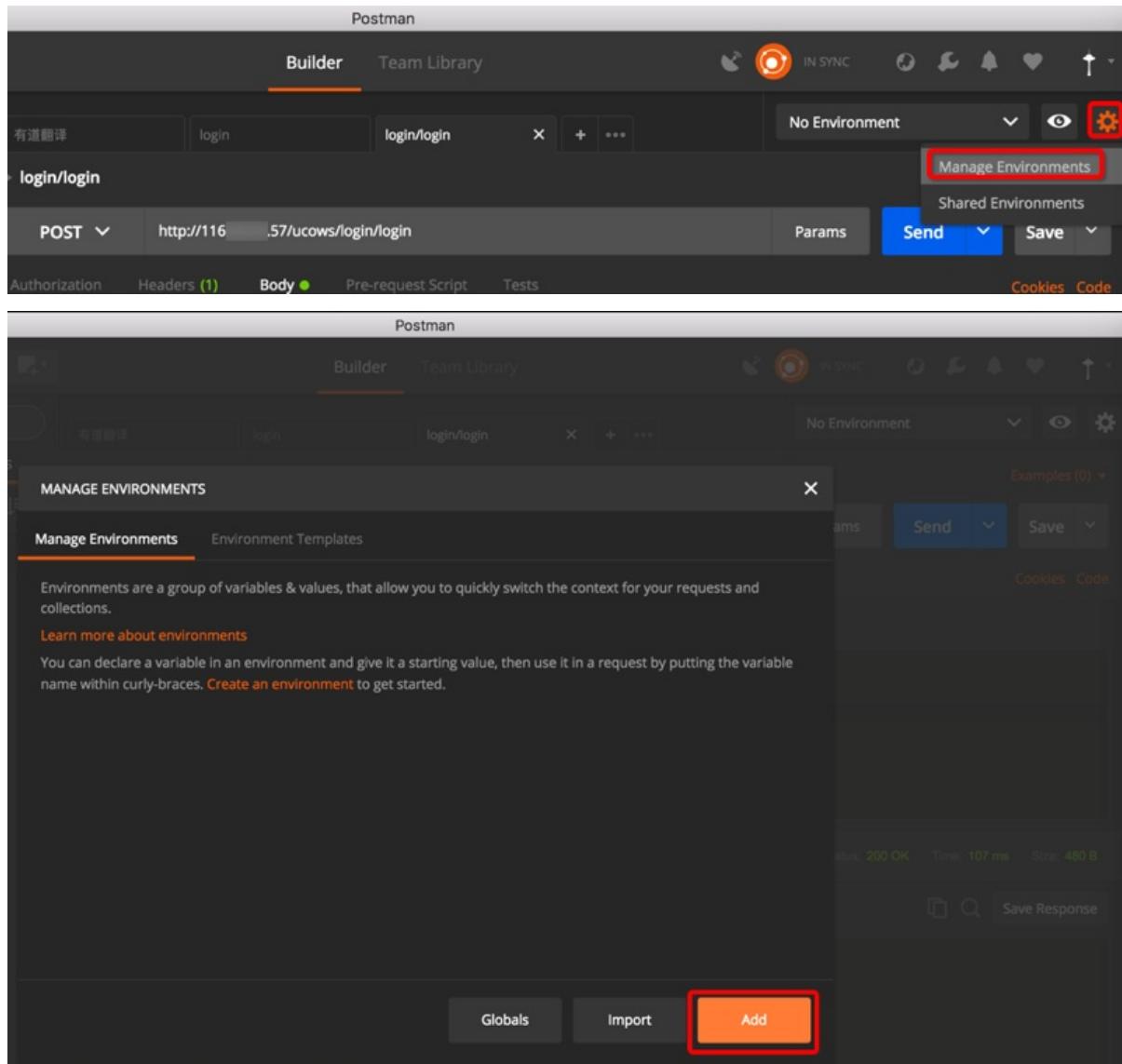
The screenshot shows the Postman interface with a POST request to `http://116.57.ucows/login/login`. The 'Environment' dropdown is set to 'No Environment'. A red box highlights the 'Environment' button in the top right. The 'Body' tab is selected, showing a JSON body with a red box highlighting the 'JSON (application/json)' button. The 'Body' tab has a green dot indicating it's selected.

```

1 {
2   "username": "ipu",
3   "password": "000000"
4 }

```

或者：



Environments are a group of variables & values, that allow you to quickly switch the context for your requests and collections.

[Learn more about environments](#)

You can declare a variable in an environment and give it a starting value, then use it in a request by putting the variable name within curly-braces. [Create an environment](#) to get started.

输入Key和value：

MANAGE ENVIRONMENTS

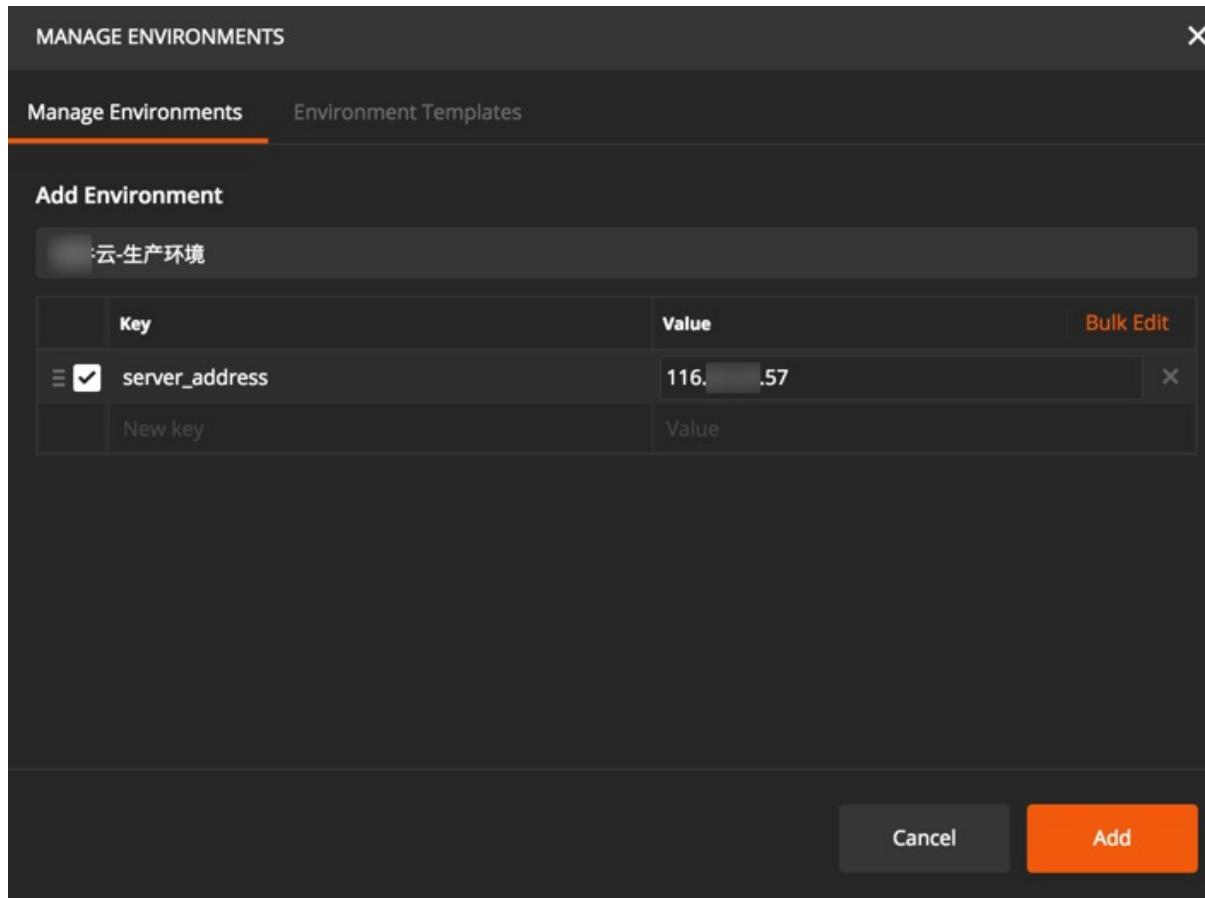
Manage Environments Environment Templates

Add Environment

云-生产环境

Key	Value	Bulk Edit
<input checked="" type="checkbox"/> server_address	116. .57	X
New key	Value	

Cancel Add



点击Add后：

MANAGE ENVIRONMENTS

Manage Environments Environment Templates

Environments are a group of variables & values, that allow you to quickly switch the context for your requests and collections.

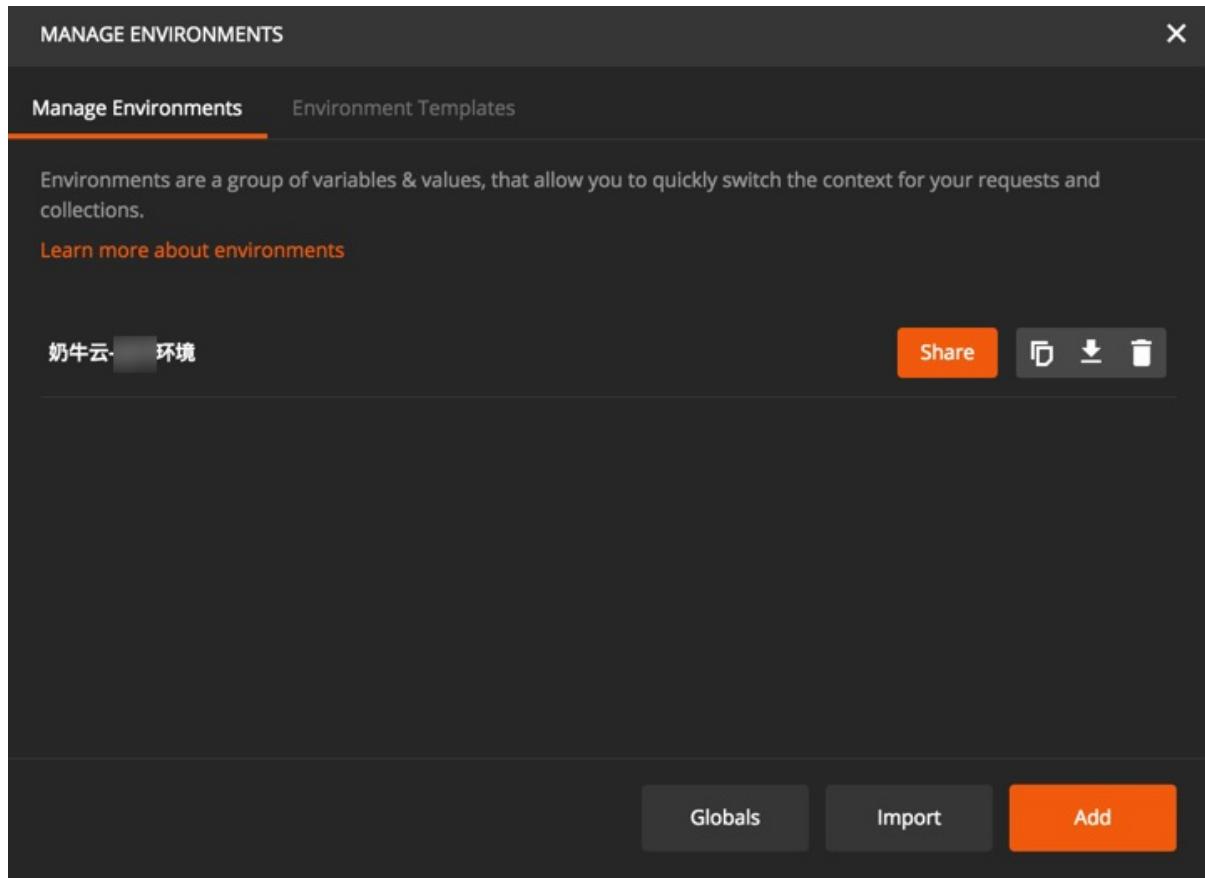
Learn more about environments

奶牛云- 环境

Share

Import

Add



[info] 环境变量可以使用的地方

- URL
- URL params
- Header values
- form-data/url-encoded values
- Raw body content
- Helper fields
- 写 test 测试脚本中
 - 通过 postman 的接口，获取或设置环境变量的值。

此处把之前的在 url 中的 IP 地址（或域名）换成环境变量：

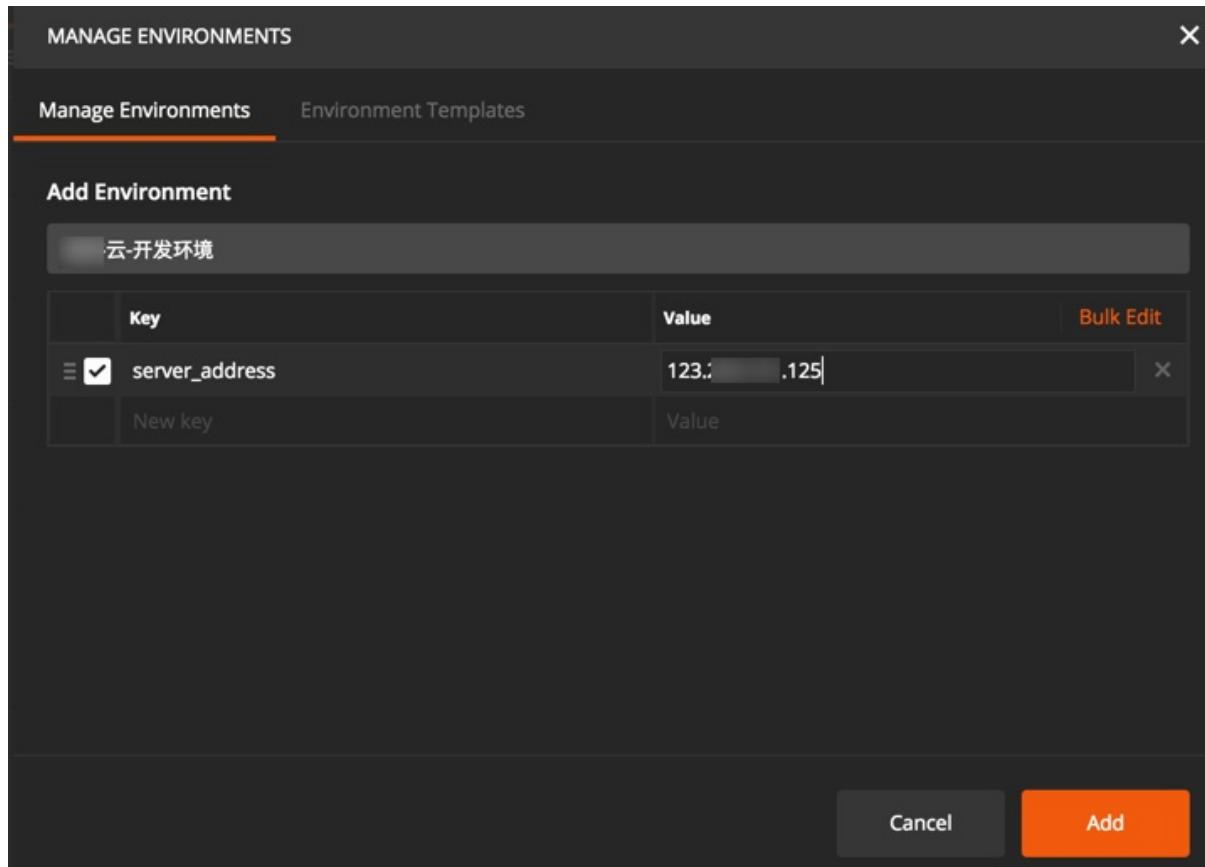
The screenshot shows the Postman Builder interface. In the top right, there's a dropdown menu for environments, with one entry labeled "云-生产环境" (Cloud - Production Environment) highlighted with a red box. Below this, in the main workspace, a POST request is defined with the URL "http://{{server_address}}/ucows/login/login". The URL field is also highlighted with a red box. The "Body" tab is selected, showing a JSON payload:

```
1 {  
2   "username": "pu",  
3   "password": "000000"  
4 }
```

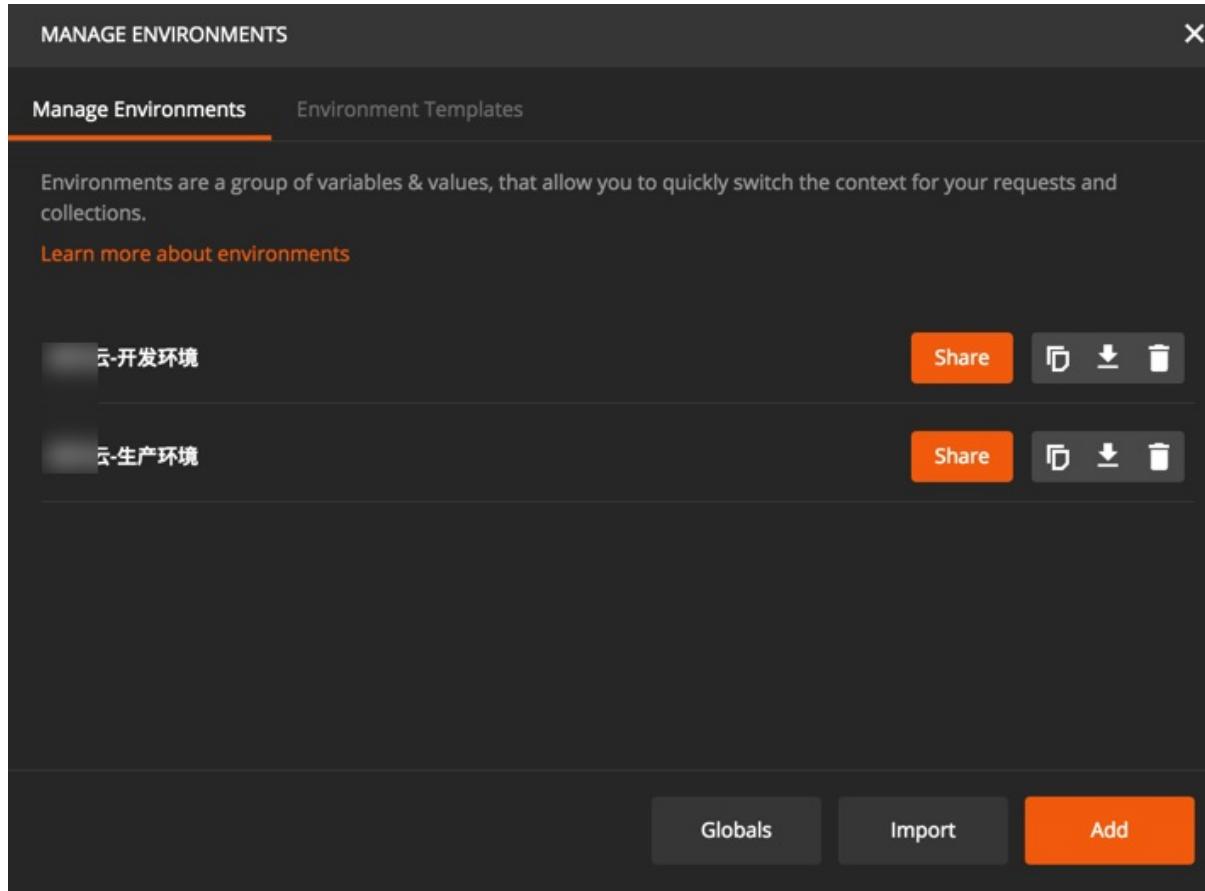
鼠标移动到环境变量上，可以动态显示出具体的值：

This screenshot is similar to the previous one, but it includes a tooltip for the environment variable "server_address". The tooltip is a white box with a red border, containing the text "E server_address" and "Value 116 .57". It also shows "Scope Environment". The rest of the interface is identical to the first screenshot.

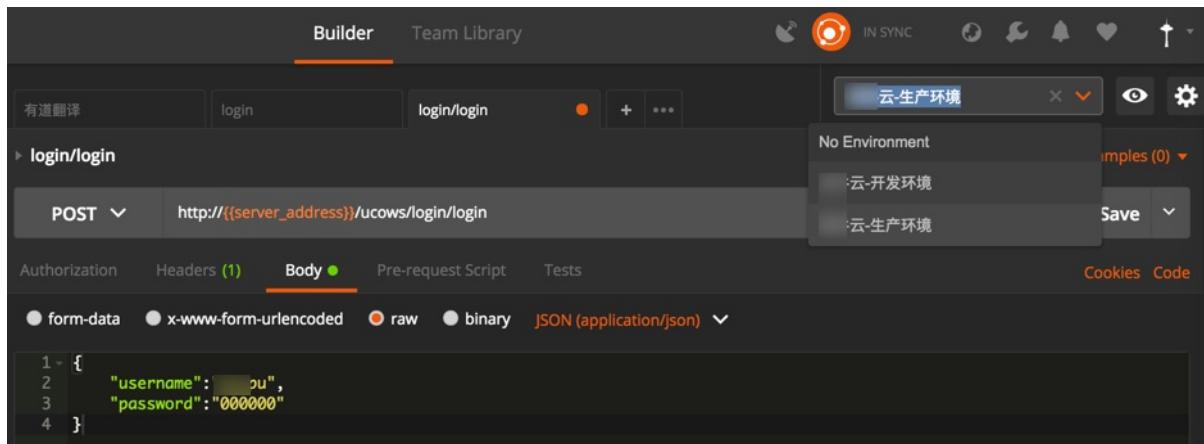
再去添加另外一个开发环境：



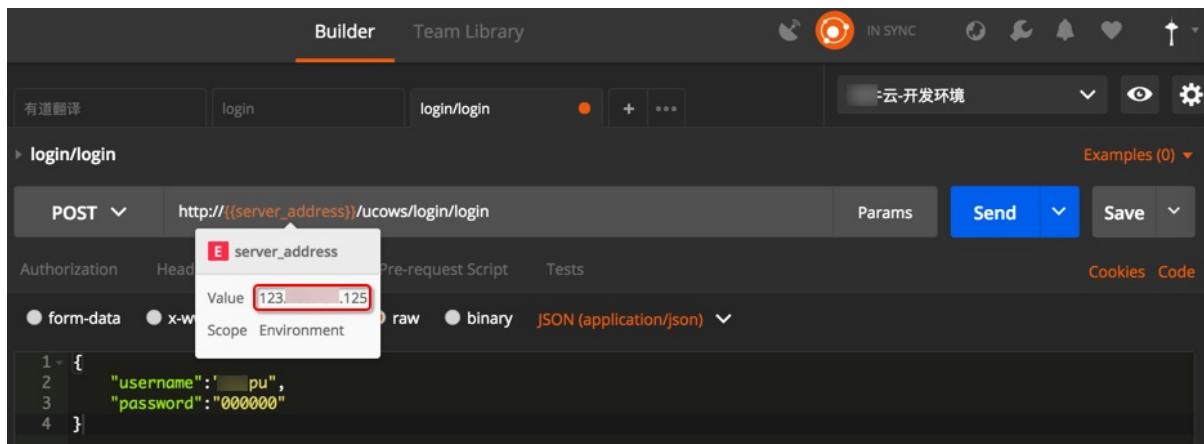
则可添加完2个环境变量，表示两个服务器地址，两个版本：



然后就可以切换不同服务器环境了：



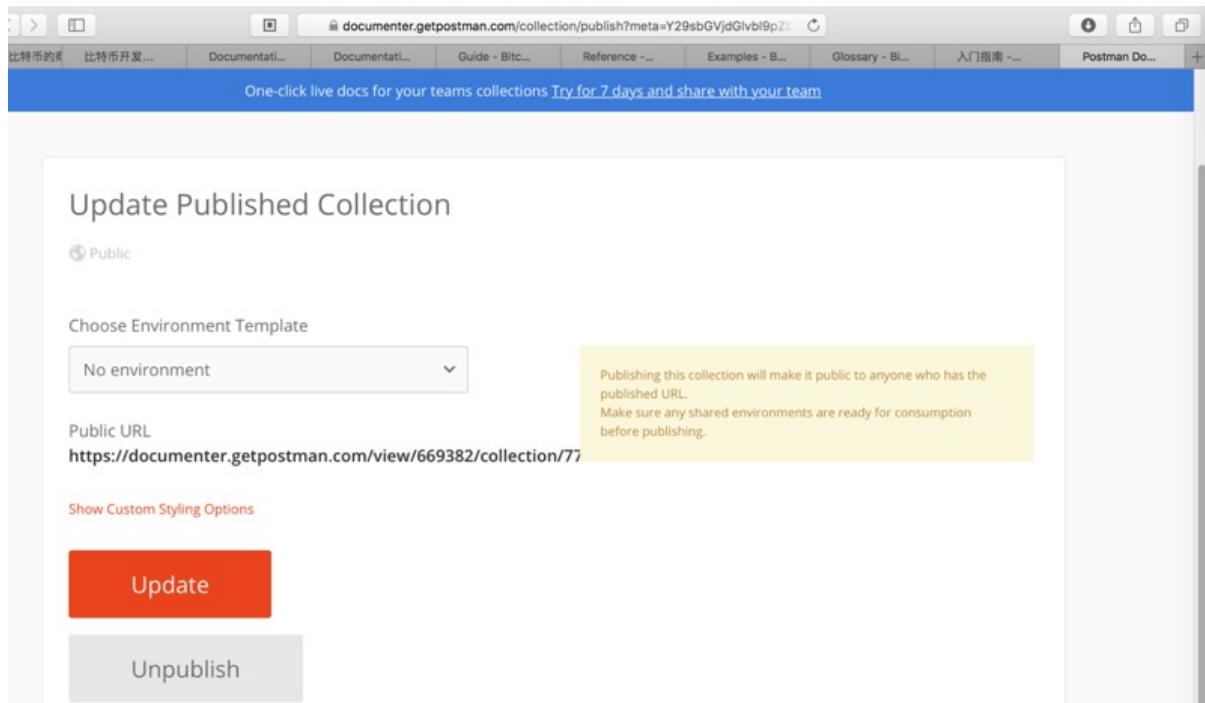
可以看到，同样的变量server_address，在切换后对应IP地址就变成希望的开发环境的IP了：



Postman导出API文档中多个环境变量的效果

顺带也去看看，导出为API文档后，带了这种Environment的变量的接口，文档长什么样子：

发现是在发布之前，需要选择对应的环境的：



Update Published Collection

Public

Choose Environment Template

No environment

No environment

一线 开发环境

2/c

云-开发环境

云-生产环境

云-英文版

Unpublish

Update Published Collection

Public

Choose Environment Template

云-开发环境

Public URL

<https://documenter.getpostman.com/view/669382/collection/77>

Show Custom Styling Options

Update

Unpublish

发布后的文档，可以看到所选环境和对应服务器的IP的：

The screenshot shows the Postman application window. At the top, there's a navigation bar with links like '比特币开发', 'Documentati...', 'Documentati...', 'Guide - Bitc...', 'Reference - ...', 'Examples - B...', 'Glossary - Bi...', '入门指南 - ...', 'Postman Do...', and '奶牛云'. Below the navigation bar, the title '奶牛云' is displayed. On the left side, there's a sidebar with a list of API endpoints. In the main area, there's a request card for a POST request to 'login/login'. The URL field contains 'http://118.89.104.149/ucows/login/login'. To the right of the URL field, there's a red box highlighting the 'Run in Postman' button. Further down, there's a 'Sample Request' section with a code block for curl:

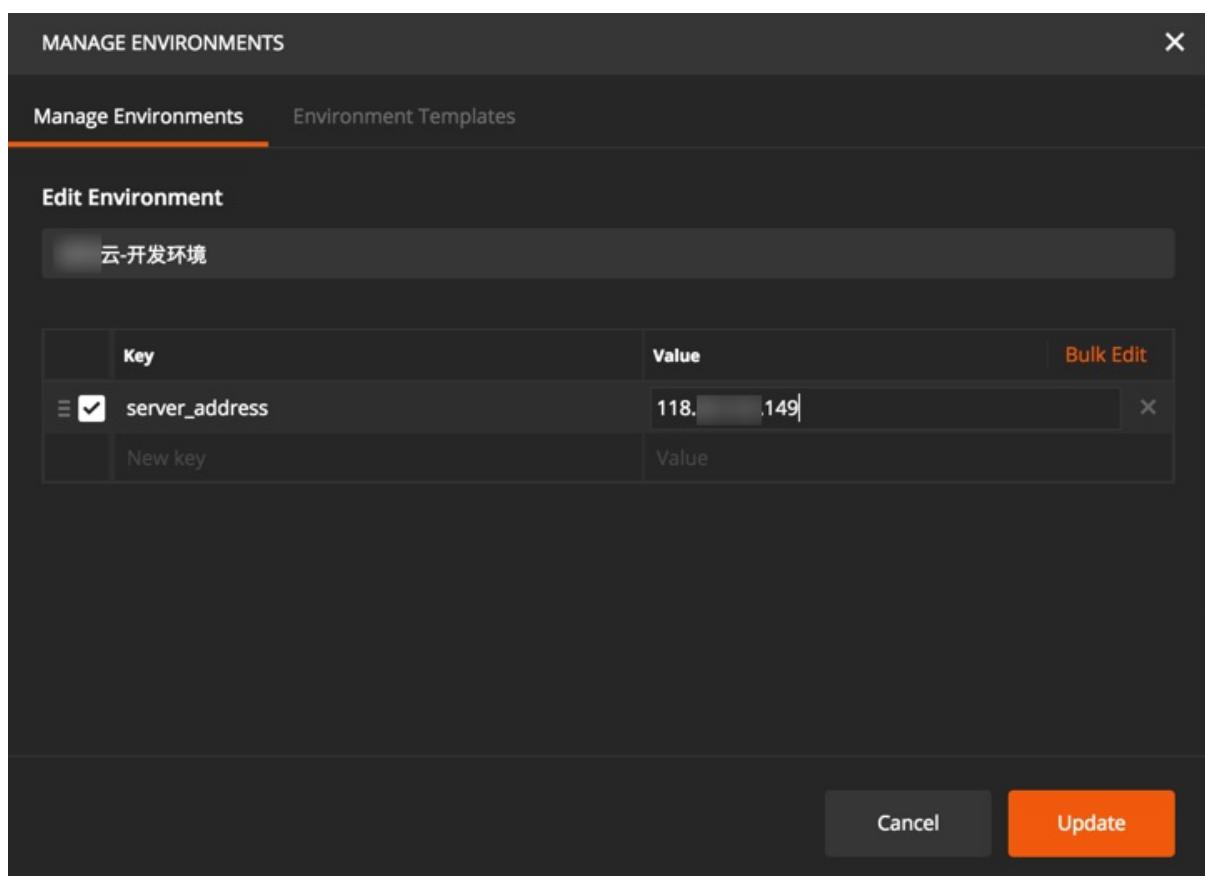
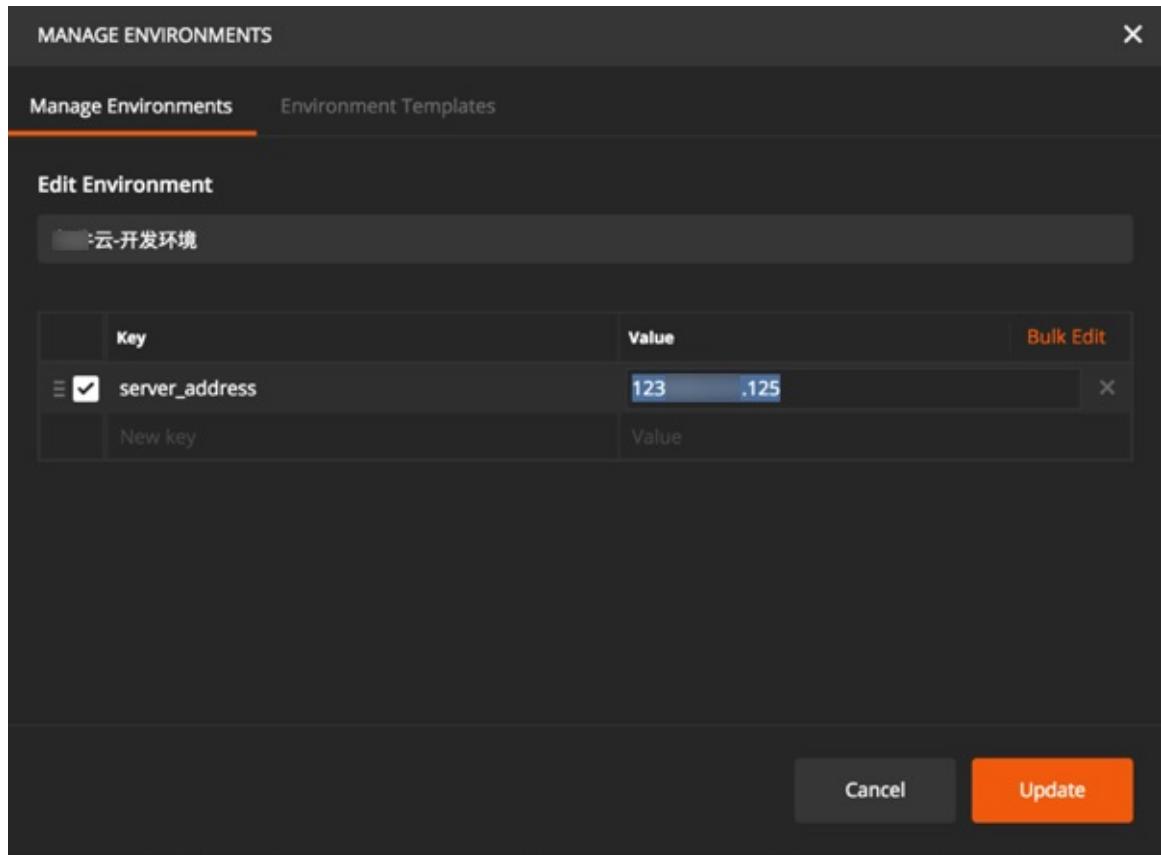
```
curl --request POST \
--url 'http://118.89.104.149/ucows/login/login' \
--header 'content-type: application/json' \
--data '{' \
"username": "weipu", \
"password": "000000" \
}'
```

当然发布文档后，也可以实时切换环境：

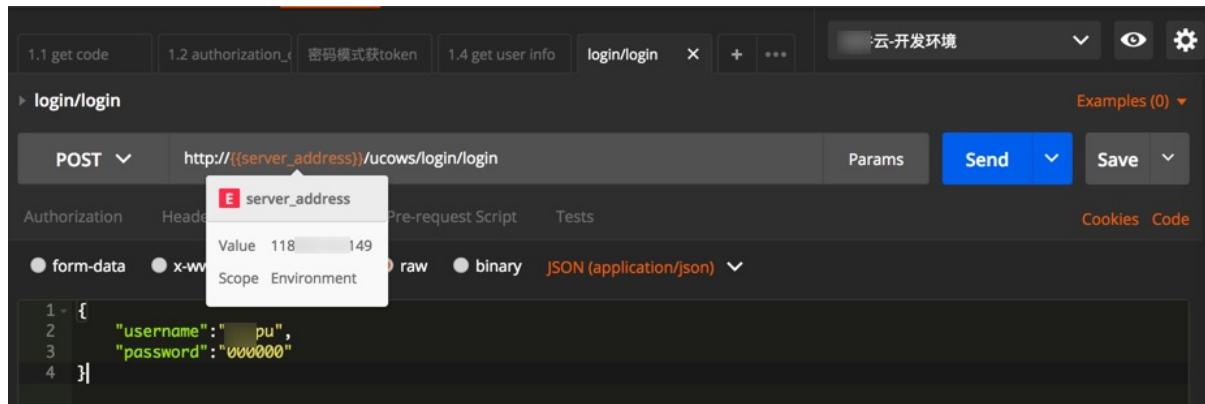
The screenshot shows the Postman interface. On the left, there's a sidebar with 'POST login/login' and a list of other API endpoints. The main area shows a 'Sample Request' for 'login/login' with a curl command. Above the curl command, there's a dropdown menu for 'Environment'. The menu is open, showing several environments: 'No environment', 'SHARED TEMPLATES', 'No shared environments', 'PRIVATE ENVIRONMENTS', and four local environments: ':云-开发环境', ':一线 开发环境', ':云-生产环境', and ':云-英文版'. The ':云-生产环境' option is highlighted. At the top right, there's a dropdown for '李茂 (crifan)' and another for '开发环境' which is currently set to ':云-开发环境'. Below the main interface, there's a detailed view of the 'login/login' endpoint, showing its URL and headers.

环境变量的好处

当更换服务器时，直接修改变量的IP地址：



即可实时更新，当鼠标移动到变量上即可看到效果：

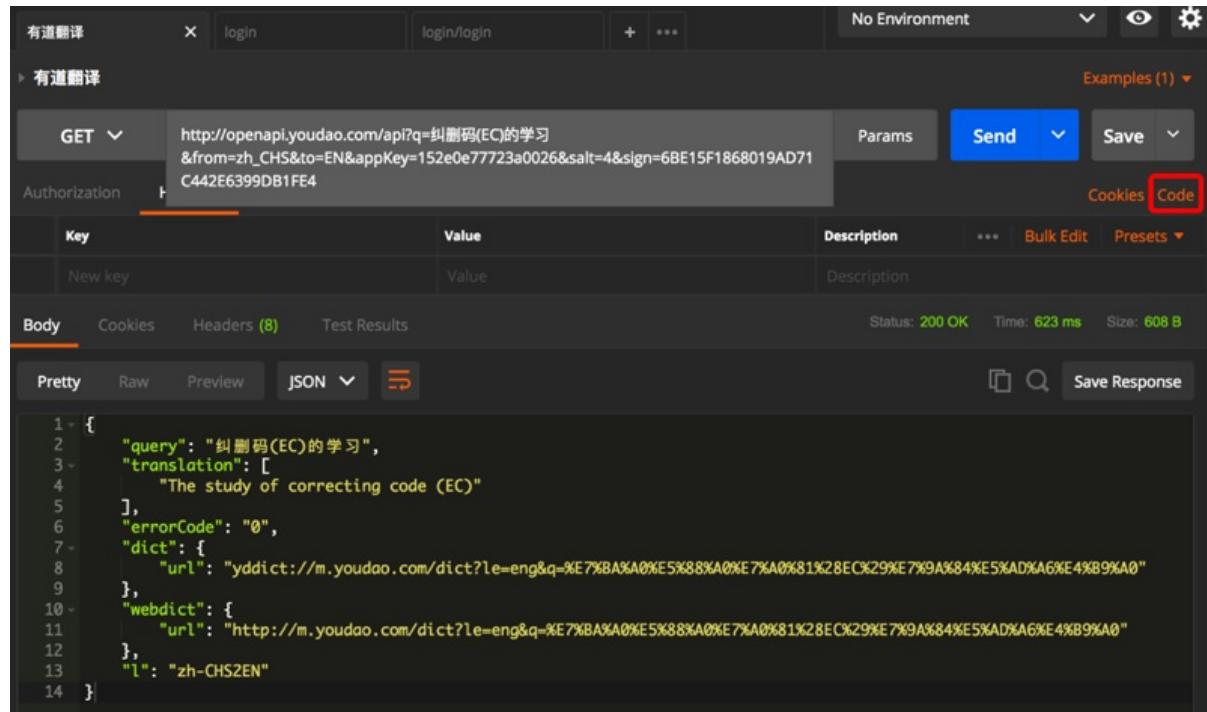


crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 11:45:28

代码生成工具

查看当前请求的HTTP原始内容

对于当前的请求，还可以通过点击Code



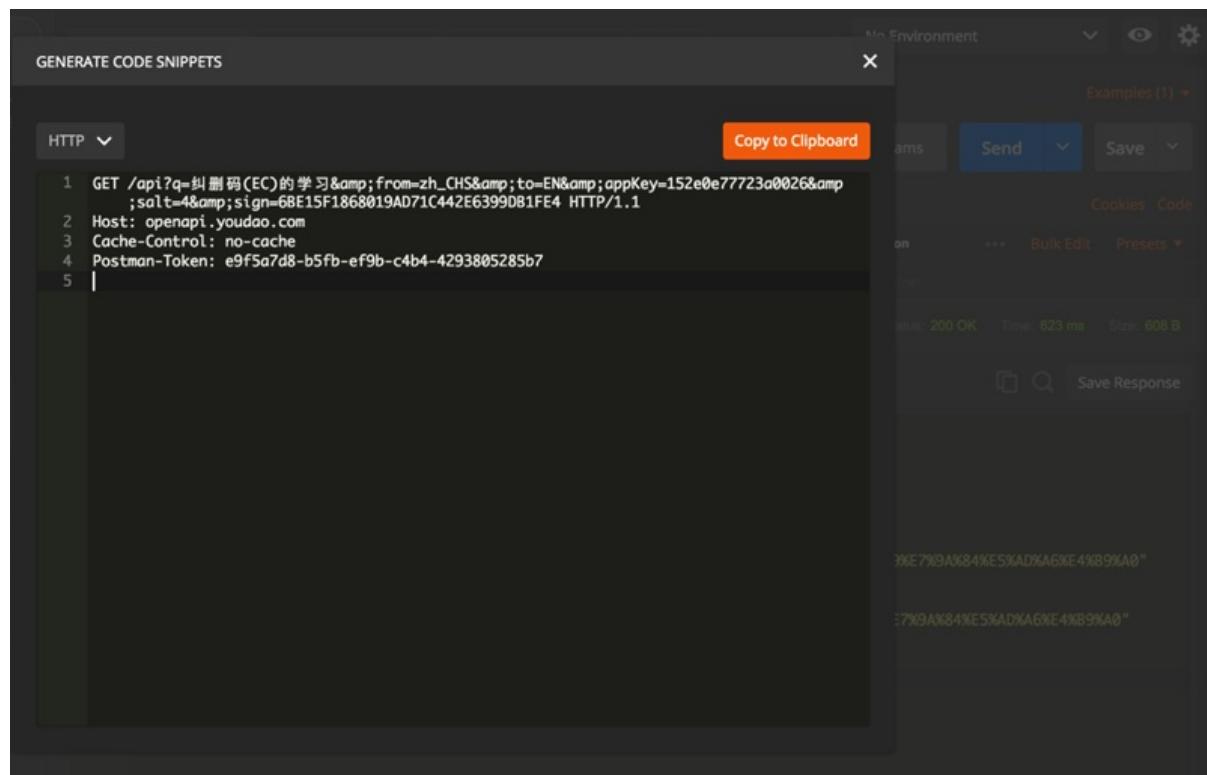
The screenshot shows a Postman request for the Youdao Translate API. The URL is `http://openapi.youdao.com/api?q=纠删码(EC)的学习&from=zh_CHS&to=EN&appKey=152e0e77723a0026&salt=4&sign=6BE15F1868019AD71C442E6399DB1FE4`. The 'Body' tab is selected, showing a JSON response with the following content:

```

1 {
2   "query": "纠删码(EC)的学习",
3   "translation": [
4     "The study of correcting code (EC)"
5   ],
6   "errorCode": "0",
7   "dict": {
8     "url": "yddict://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%A0%81%28EC%29%E7%9A%84%E5%AD%A6%E4%B9%A0"
9   },
10  "webdict": {
11    "url": "http://m.youdao.com/dict?le=eng&q=%E7%BA%A0%E5%88%A0%E7%A0%81%28EC%29%E7%9A%84%E5%AD%A6%E4%B9%A0"
12  },
13  "l": "zh-CHS2EN"
14 }

```

去查看对应的符合HTTP协议的原始的内容：



The screenshot shows the raw HTTP code snippets for the same request. The code is as follows:

```

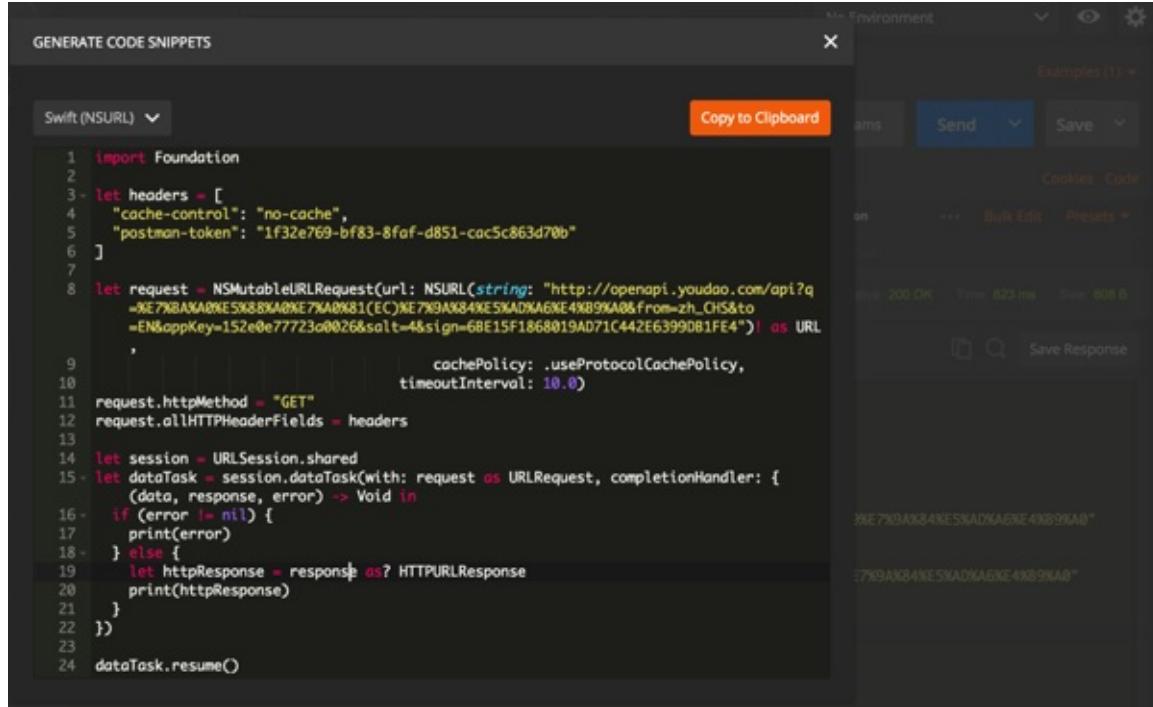
1 GET /api?q=纠删码(EC)的学习&from=zh_CHS&to=EN&appKey=152e0e77723a0026&salt=4&sign=6BE15F1868019AD71C442E6399DB1FE4 HTTP/1.1
2 Host: openapi.youdao.com
3 Cache-Control: no-cache
4 Postman-Token: e9f5a7d8-b5fb-ef9b-c4b4-4293805285b7
5

```

各种语言的示例代码 Code Generation Tools

比如：

Swift语言



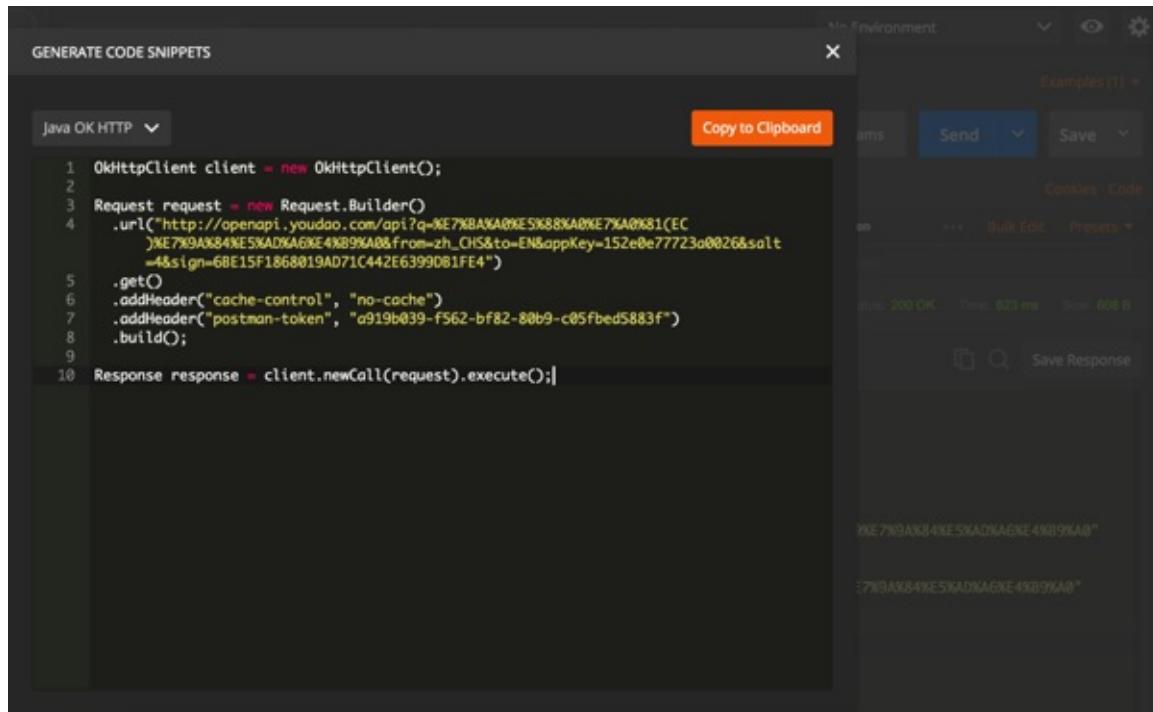
The screenshot shows the Postman interface with the "GENERATE CODE SNIPPETS" tab selected. The language dropdown is set to "Swift (NSURL)". The code generated is:

```

1 import Foundation
2
3 let headers = [
4     "cache-control": "no-cache",
5     "postman-token": "1f32e769-bf83-8faf-d851-cac5c863d70b"
6 ]
7
8 let request = NSMutableURLRequest(url: NSURL(string: "http://openapi.youdao.com/api?q=%E7%8BAK&key=%E5%88%A0&%E7%AA%81(EC)%E7%9A%84%E5%AD%6X&4X89%A0&from=zh_CHS&to=EN&appKey=152e0e77723a0026&salt=4&sign=68E15F1868019AD71C442E63990B1FE4")) as URLRequest
9
10 request.cachePolicy = .useProtocolCachePolicy,
11         timeoutInterval: 10.0
12 request.httpMethod = "GET"
13 request.allHTTPHeaderFields = headers
14
15 let session = URLSession.shared
16 let dataTask = session.dataTask(with: request as URLRequest, completionHandler: {
17     (data, response, error) -> Void in
18     if (error != nil) {
19         print(error)
20     } else {
21         let httpResponse = response as? HTTPURLResponse
22         print(httpResponse)
23     }
24 })
25 dataTask.resume()

```

Java语言



The screenshot shows the Postman interface with the "GENERATE CODE SNIPPETS" tab selected. The language dropdown is set to "Java OK HTTP". The code generated is:

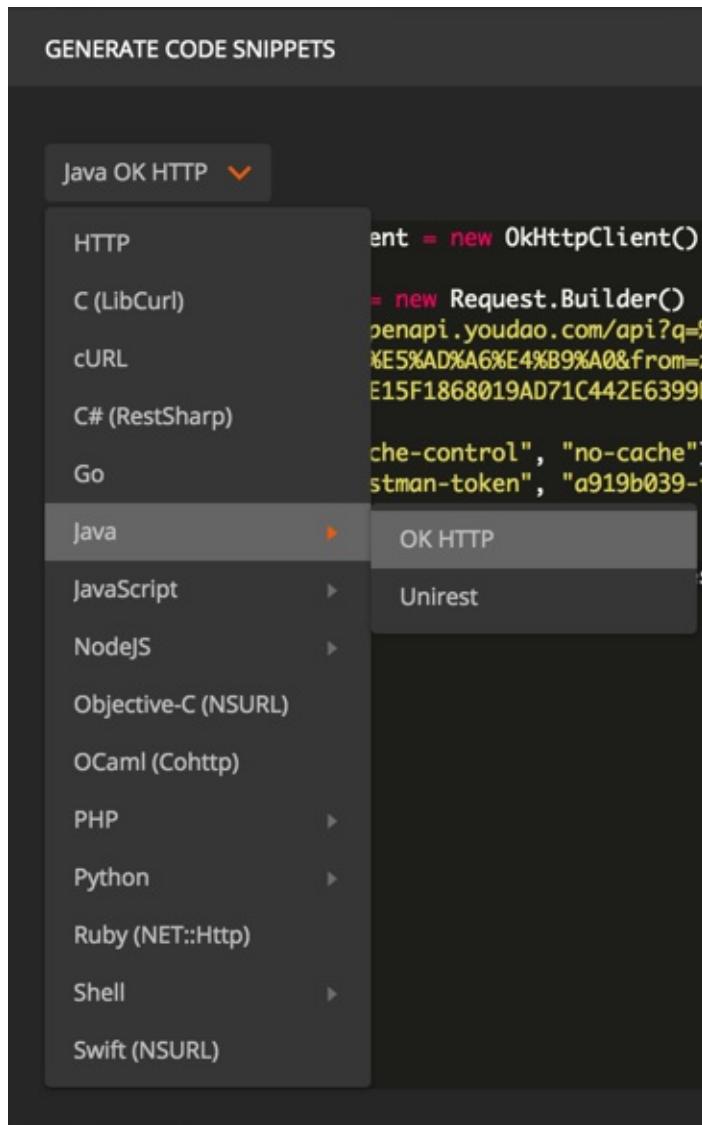
```

1 OkHttpClient client = new OkHttpClient();
2
3 Request request = new Request.Builder()
4     .url("http://openapi.youdao.com/api?q=%E7%8BAK&key=%E5%88%A0&%E7%AA%81(EC)%E7%9A%84%E5%AD%6X&4X89%A0&from=zh_CHS&to=EN&appKey=152e0e77723a0026&salt=4&sign=68E15F1868019AD71C442E63990B1FE4")
5     .get()
6     .addHeader("cache-control", "no-cache")
7     .addHeader("postman-token", "a919b039-f562-bf82-80b9-c05fbed5883f")
8     .build();
9
10 Response response = client.newCall(request).execute();

```

其他各种语言

还支持其他各种语言：



目前支持的语言有：

- HTTP
- C (LibCurl)
- cURL
- C#(RestSharp)
- Go
- Java
 - OK HTTP
 - Unirest
- Javascript
- NodeJS
- Objective-C(NSURL)
- OCaml(Cohttp)
- PHP
- Python
- Ruby(NET::Http)
- Shell

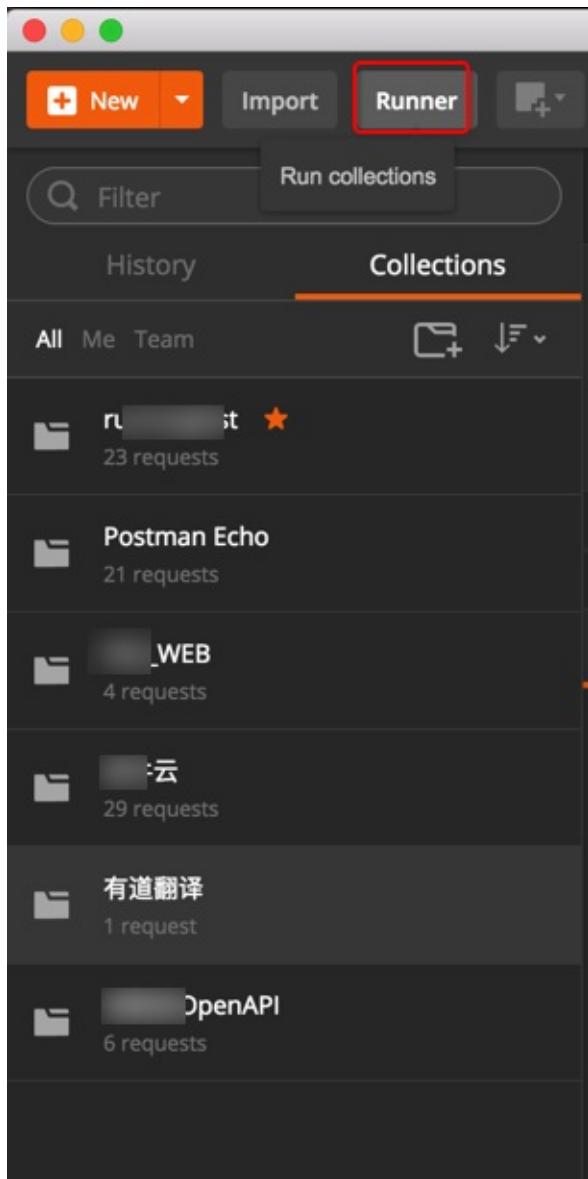
- Swift(NSURL)

代码生成工具的好处是：在写调用此API的代码时，就可以参考对应代码，甚至拷贝粘贴对应代码，即可。

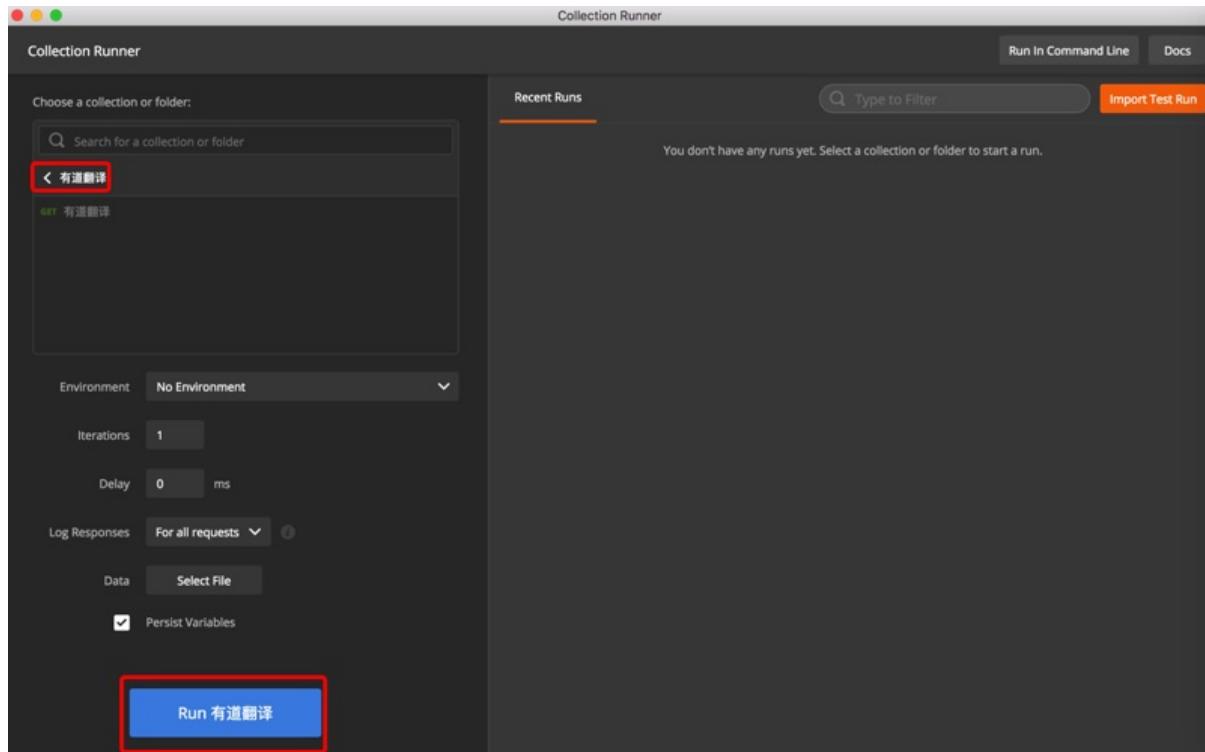
crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间：2017-12-29 20:46:55

测试接口

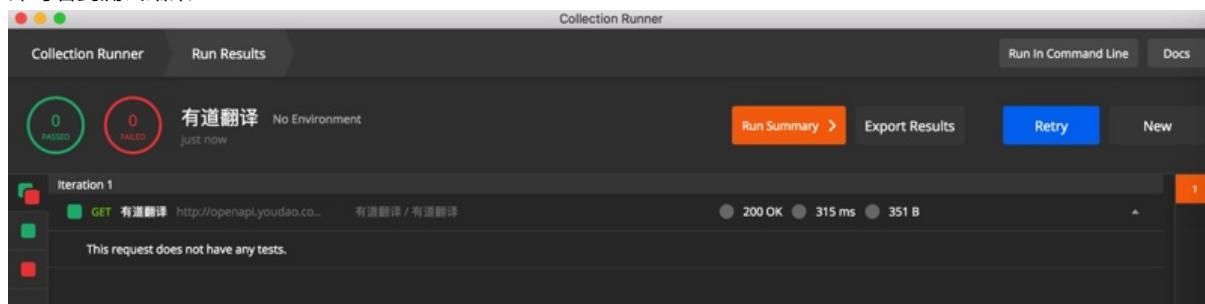
选中某个分组后，点击Runner



选中某个分组后点击Run



即可看到测试结果：



-» 好像是需要自己预先去添加test，然后才能测试的。

关于此功能的介绍可参考[Postman官网的git图](#)

[info] TODO

待后续有空继续完善此处Postman测试接口的内容。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间： 2018-01-02 11:09:57

Mock Server

TODO:

关于Mock功能，抽空去参考：

[What is Postman Pro](#)

-»

[Mock responses in Postman by using Examples – Postman Blog](#)

试试。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间： 2018-01-02 13:52:49

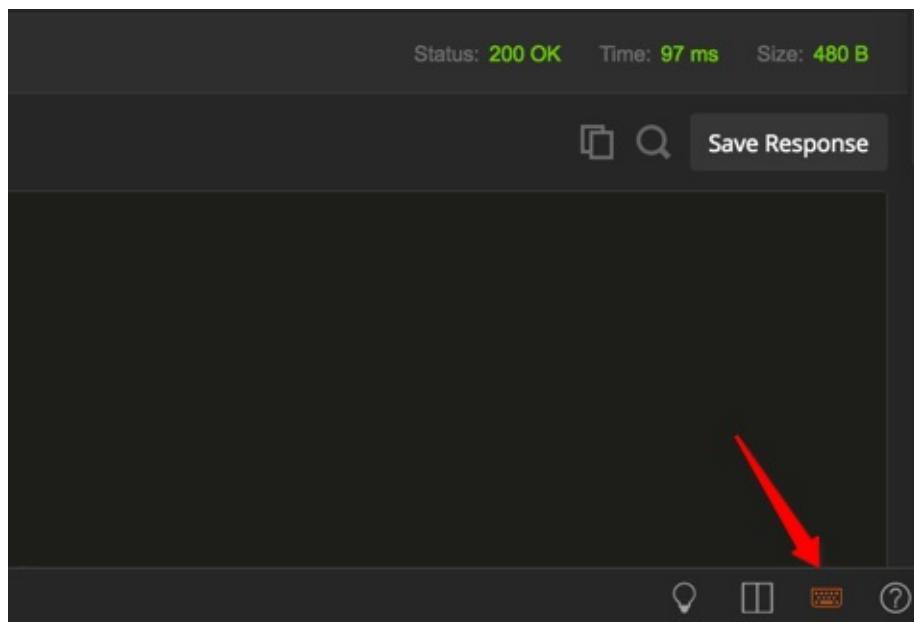
Postman功能：界面和配置

打开Postman的设置有两种方式：

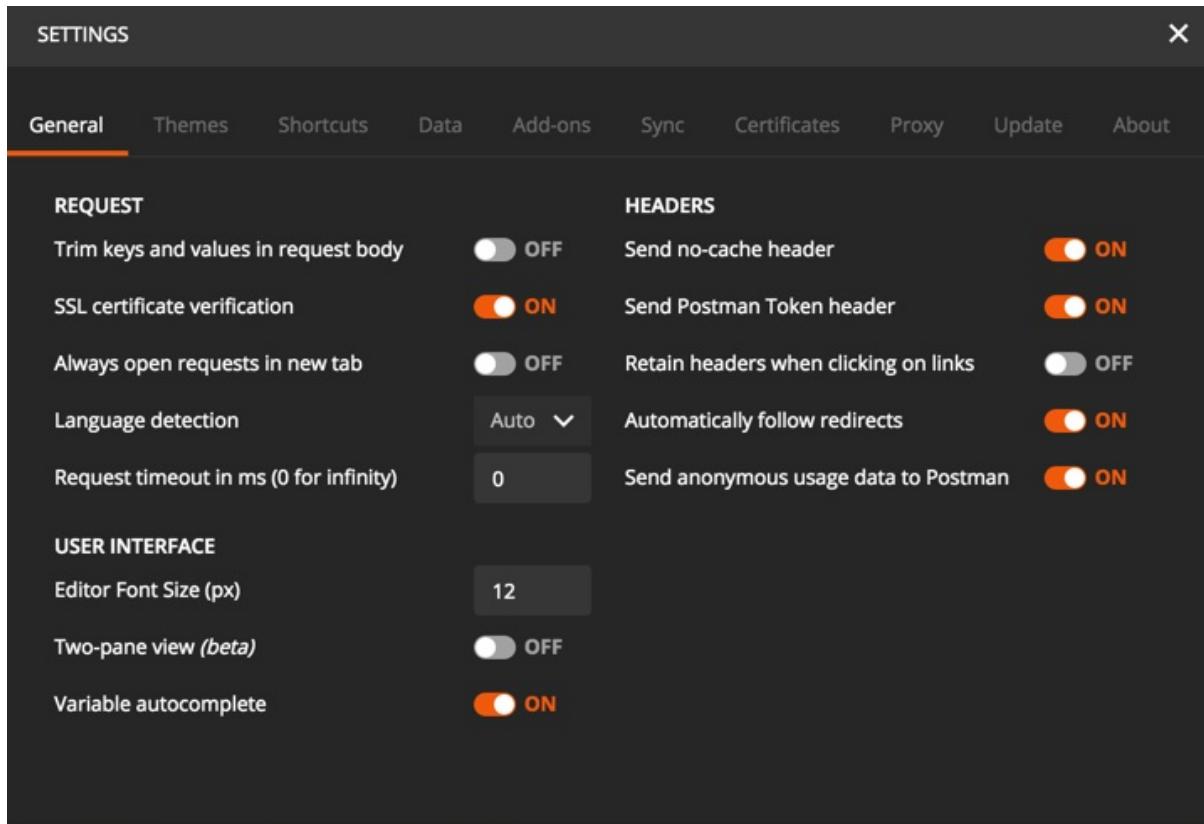
Postman -> Preferences



点击右下角的键盘按钮



都可以打开配置界面：



接下来介绍一些常见界面和功能的设置。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 11:49:45

多Tab分页

Postman支持多tab页，于此对比之前有些API调试工具就不支持多Tab页，比如 Advanced Rest Client

多tab的好处：

方便在一个tab中测试，得到结果后，复制粘贴到另外的tab中，继续测试其它接口

比如此处tab1中，测试了获取验证码接口后，拷贝手机号和验证码，粘贴到tab2中，继续测试注册的接口

The screenshot displays two tabs in the Postman interface. The left tab (tab 1) shows a POST request to `/open/smscode` with a JSON body containing `"phone": "13811118888"` and `"type": "register"`. The right tab (tab 2) shows a POST request to `/open/register` with a JSON body containing `"phone": "13811118888"`, `"smsCode": "585033"`, `"email": "register"`, `"firstName": "crifan"`, `"lastName": "Li"`, `"password": "123456"`, and `"facebookUserId": "113156"`. Both tabs indicate a successful response with a status of 200 OK.

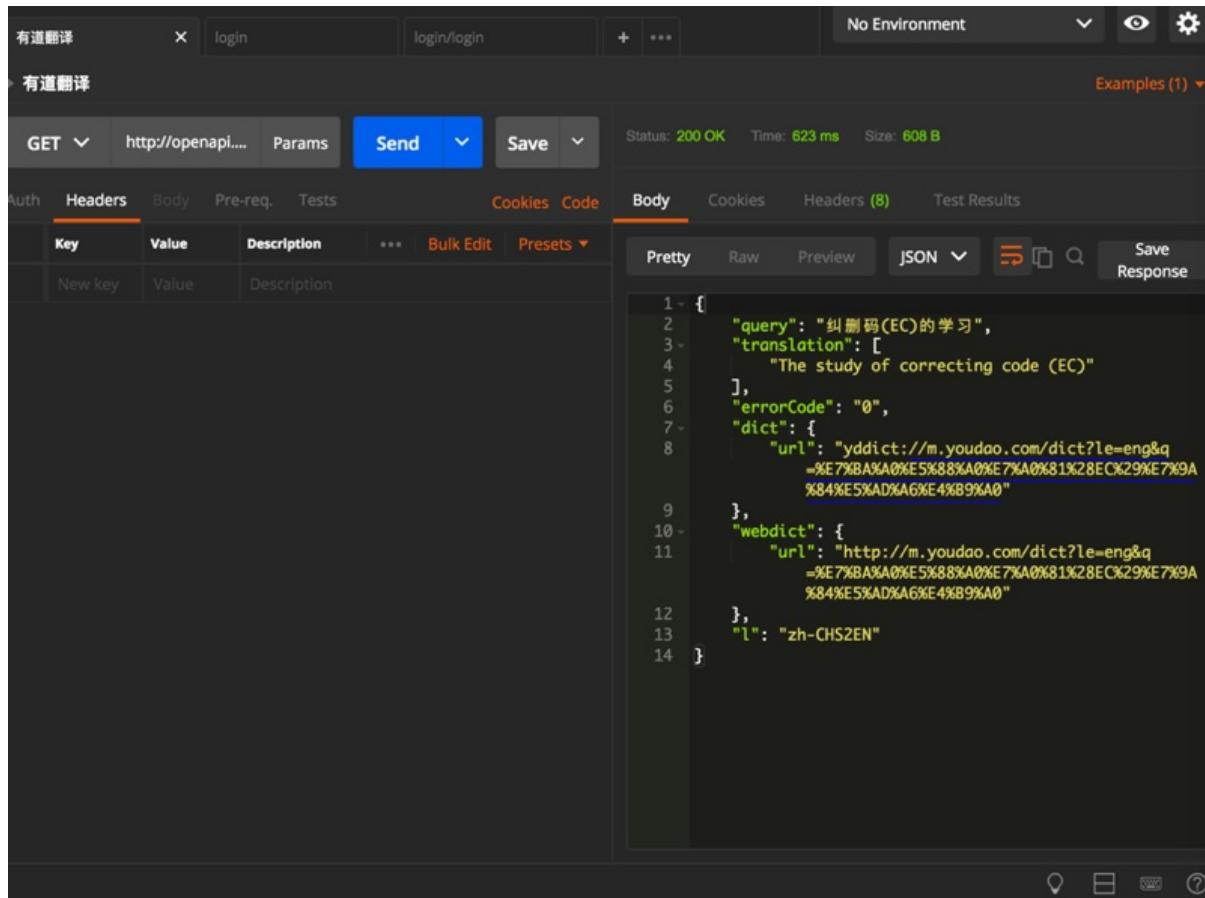
12-29 15:29:47

界面查看模式

Postman的默认的Request和Response是上下布局：

The screenshot shows the Postman interface in its default mode. At the top, there's a navigation bar with tabs for 'Builder' (which is selected) and 'Team Library'. Below the navigation bar, there's a search bar containing 'login' and a dropdown menu showing 'login/login'. To the right of the search bar are several icons: a gear, a sync icon with 'IN SYNC', and other standard application icons. The main workspace is divided into two sections: 'Request' (top) and 'Response' (bottom). The Request section contains a 'GET' method, a URL 'http://openapi.youdao.com/api?q=纠删码(EC)的学习&from=zh_CHS&to=EN&appKey=15...', and a table for 'Headers'. The Response section displays a JSON response with code highlighting. A status bar at the bottom indicates 'Status: 200 OK', 'Time: 623 ms', and 'Size: 608 B'. In the bottom right corner of the response pane, there's a button labeled 'Two pane view (⌘V)' with a red box drawn around it.

此处点击右下角的 Two pane view , 就变成左右的了：



[info] 左右布局的用途

对于数据量很大，又想要同时看到请求和返回的数据的时候，应该比较有用。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2017-12-29 20:35:54

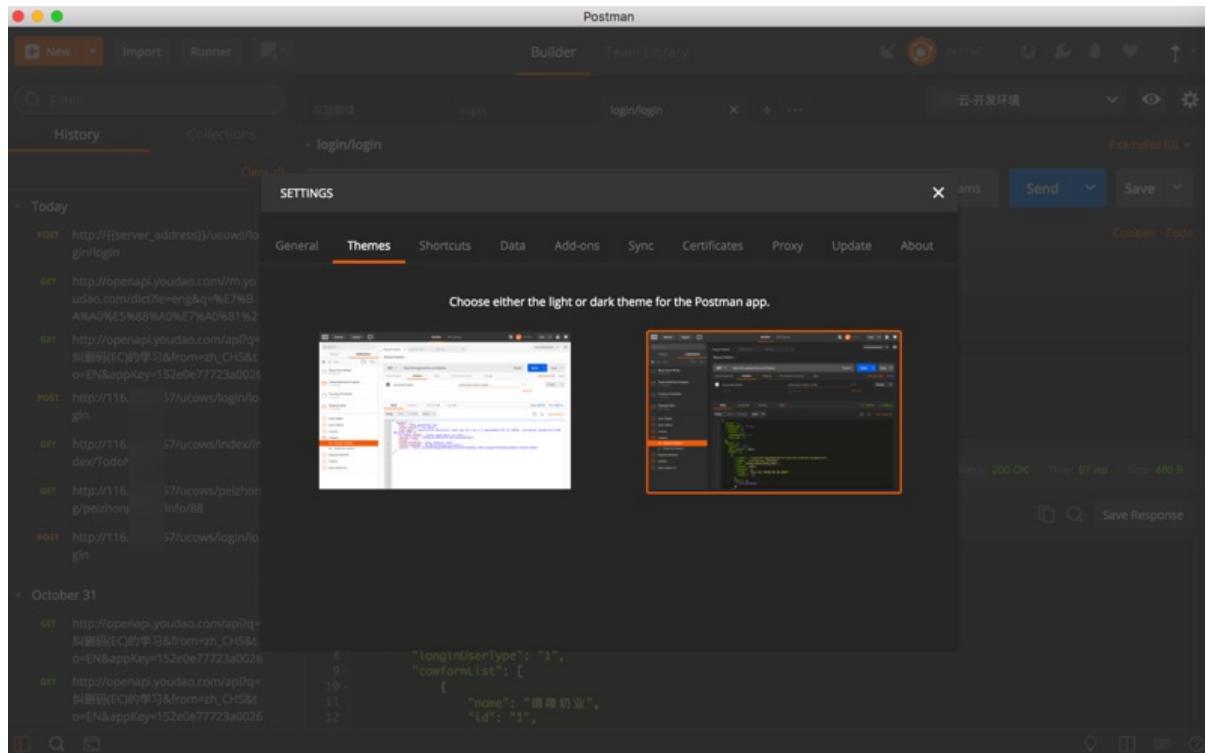
多颜色主题

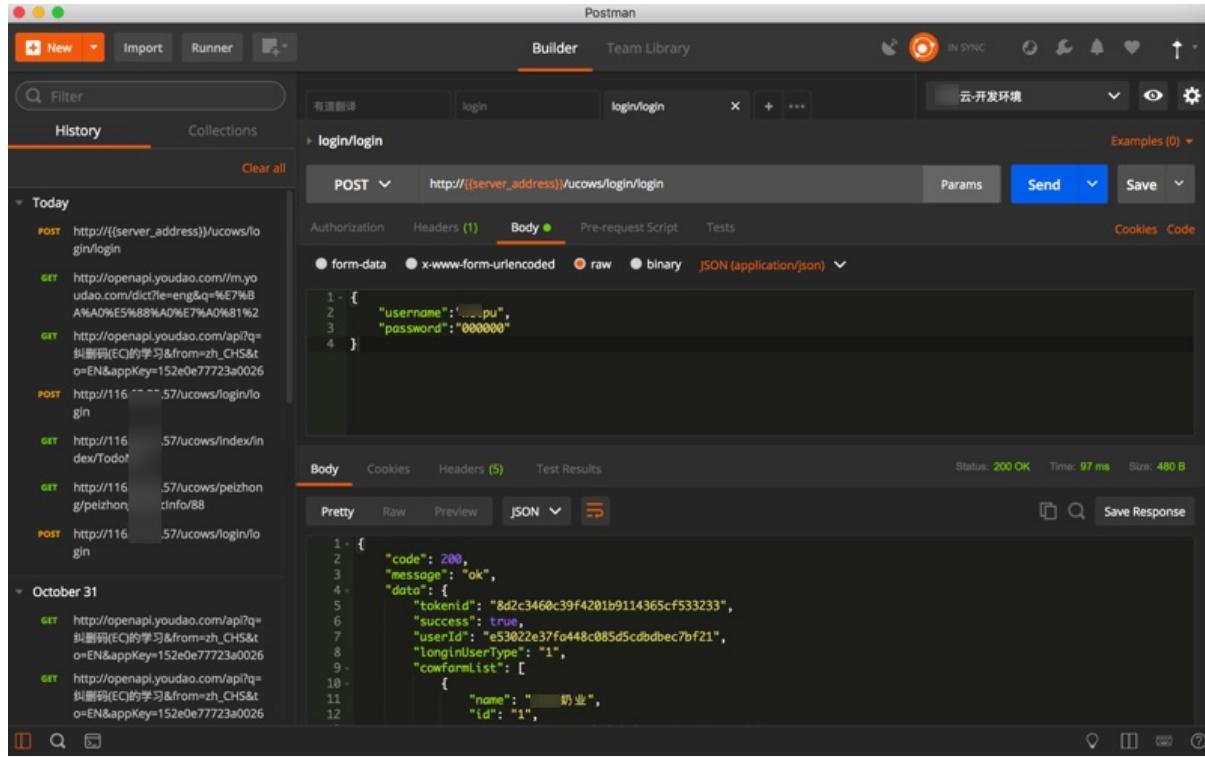
Postman支持两种主题：

深色主题

深色：

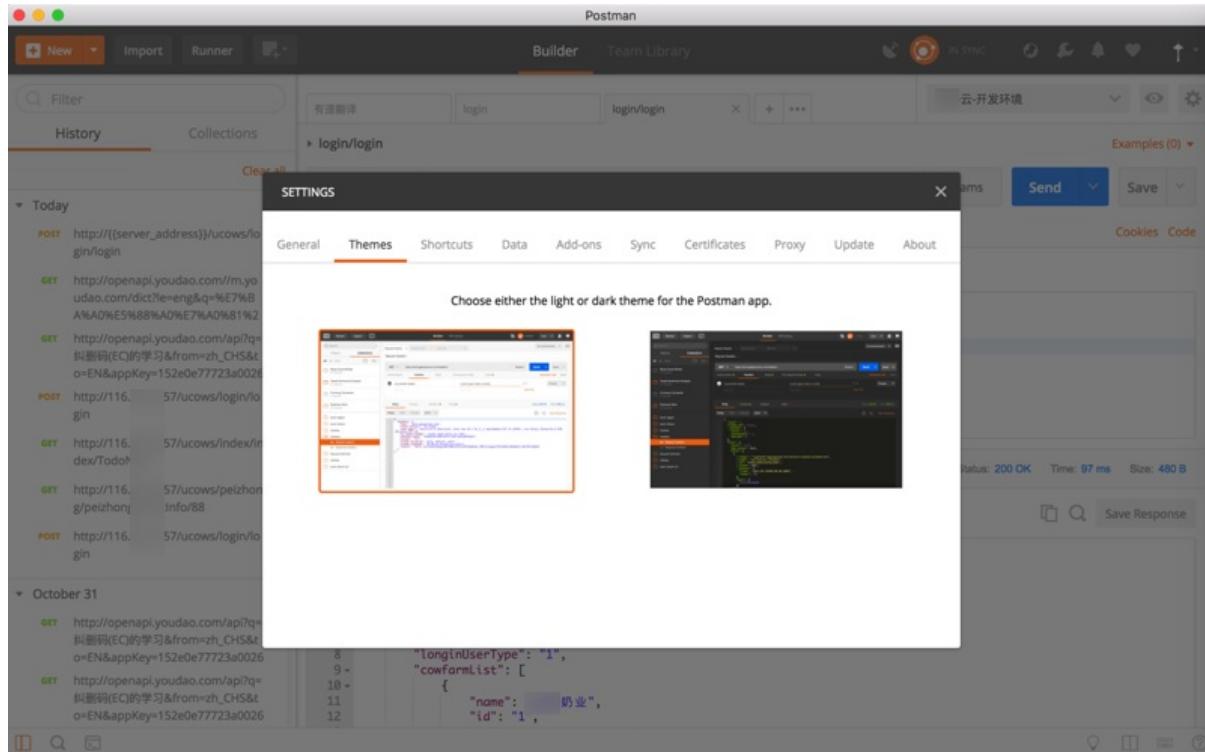
当前是深色主题，效果很不错：

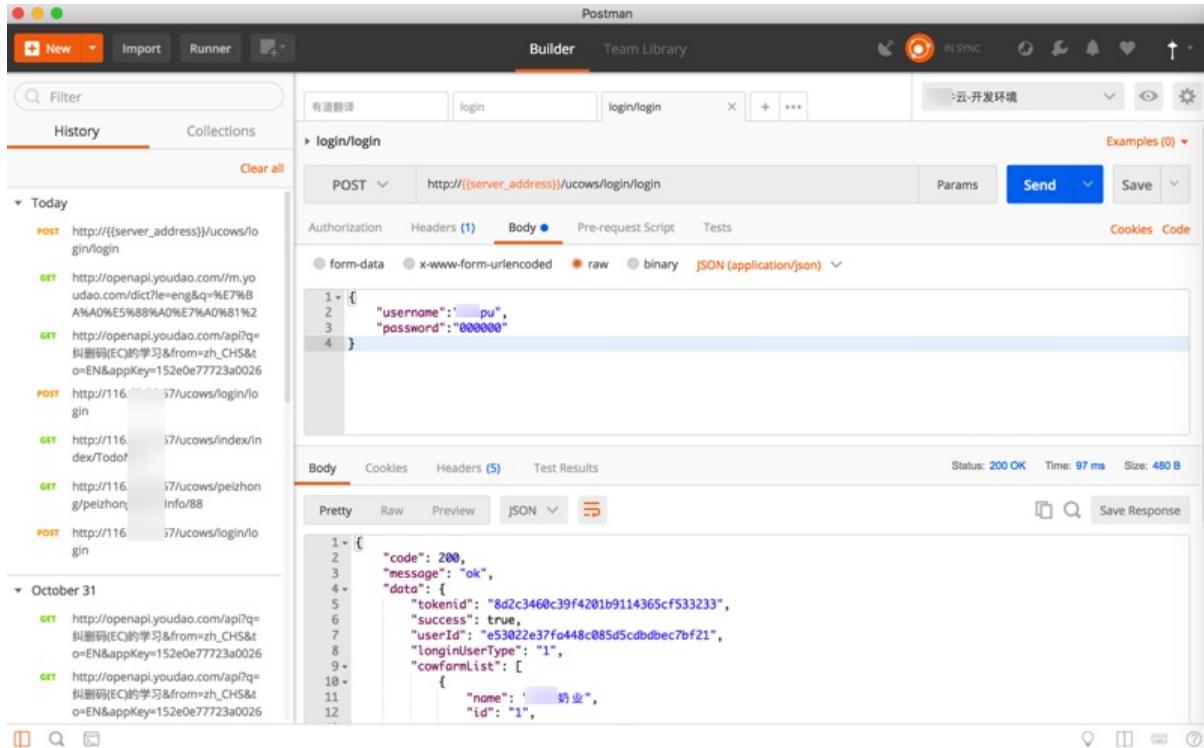




浅色主题

可以切换到 浅色主题：





crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 13:51:30

Postman生成API文档

在服务端后台的开发人员测试好了接口后，打算把接口的各种信息发给使用此API的前端的移动端人员时，往往会遇到：

- 要么是用复制粘贴 -> 格式不友好
- 要么是用Postman中截图 -> 方便看，但是不方便获得API接口和字段等文字内容
- 要么是用Postman中导出为**JSON** -> json文件中信息太繁杂，不利于找到所需要的信息
- 要么是用文档，比如去编写**Markdown**文档 -> 但后续API的变更需要实时同步修改文档，也会很麻烦

这都会导致别人查看和使用API时很不方便。

-> 对此，Postman提供了发布API文档的功能，可以很好的解决此类问题。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间：2017-12-29 21:02:40

预览和发布API文档

下面介绍Postman中如何预览和发布API文档。

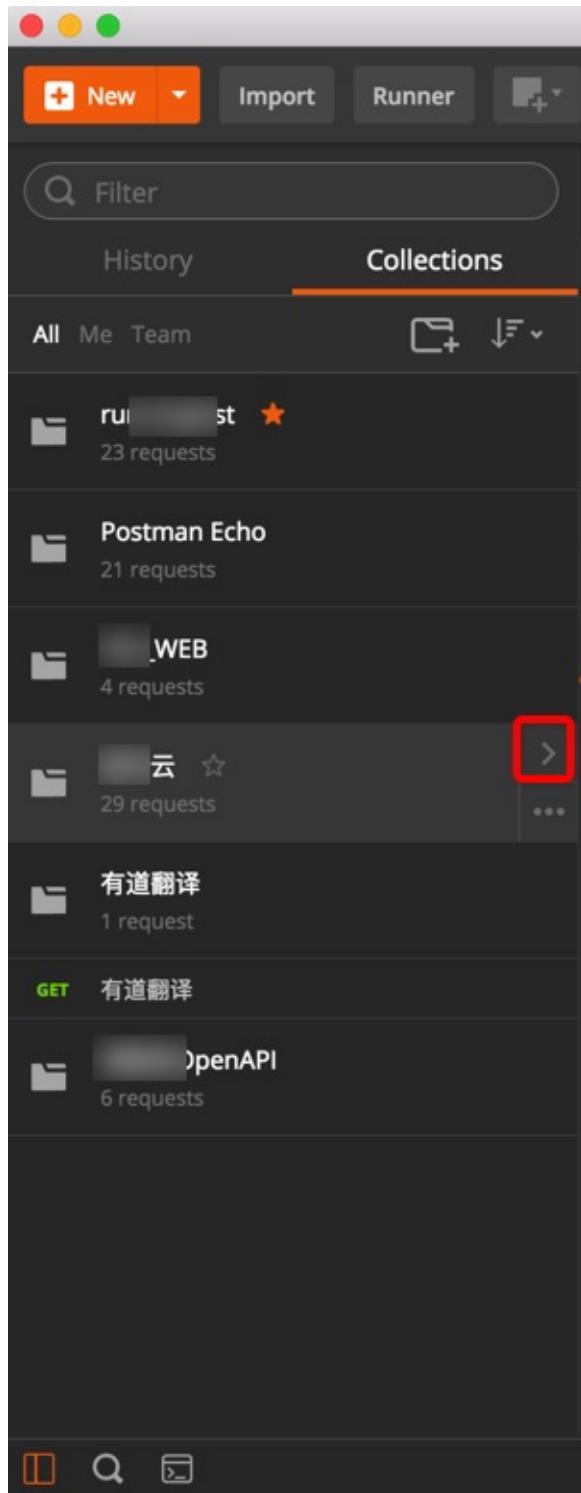
简要概述步骤

1. Collection
2. 鼠标移动到某个Collection，点击三个点
3. Publish Docs
4. Publish
5. 得到Public URL
6. 别人打开这个Public URL，即可查看API文档

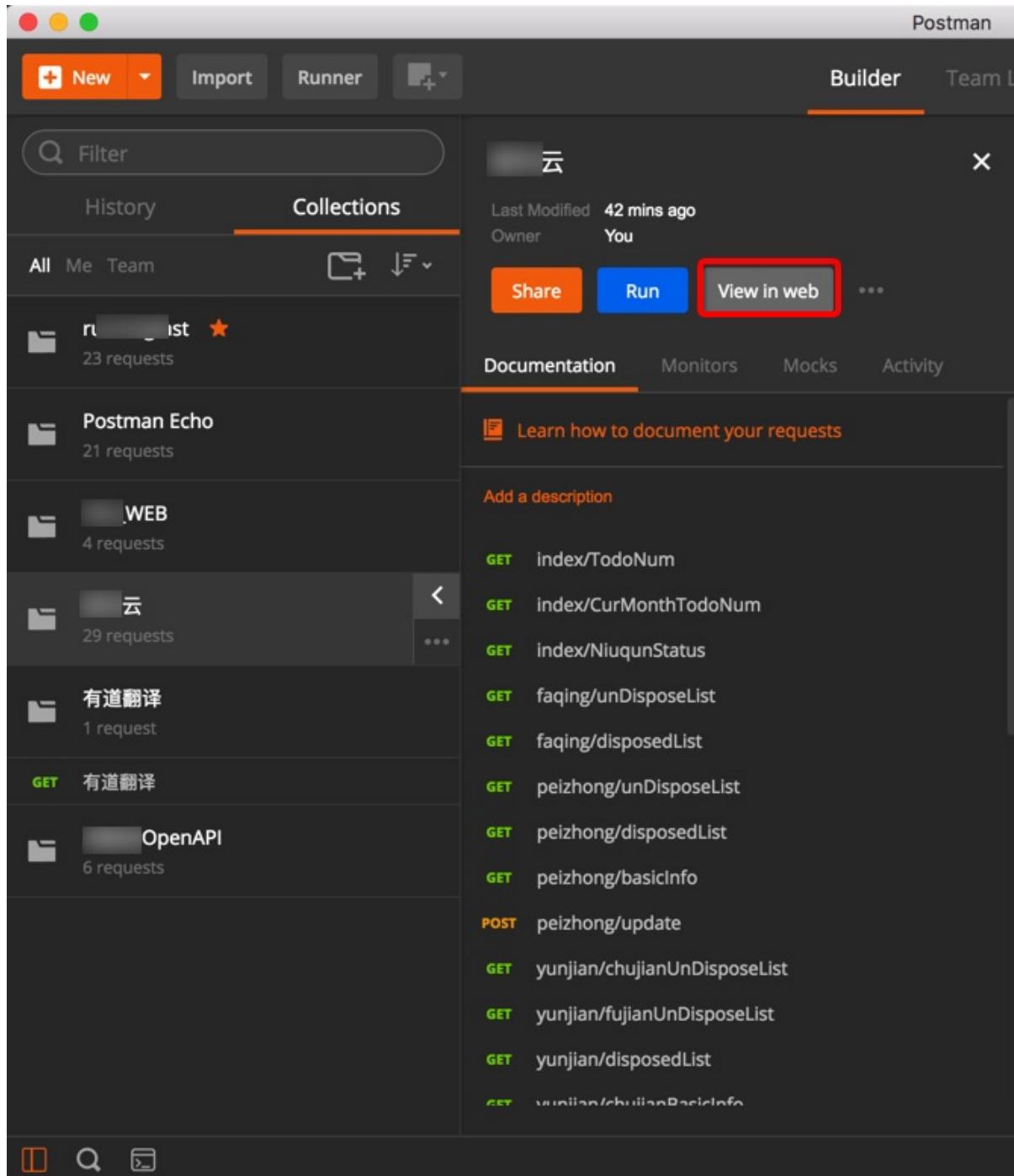
详细解释具体操作

预览API文档

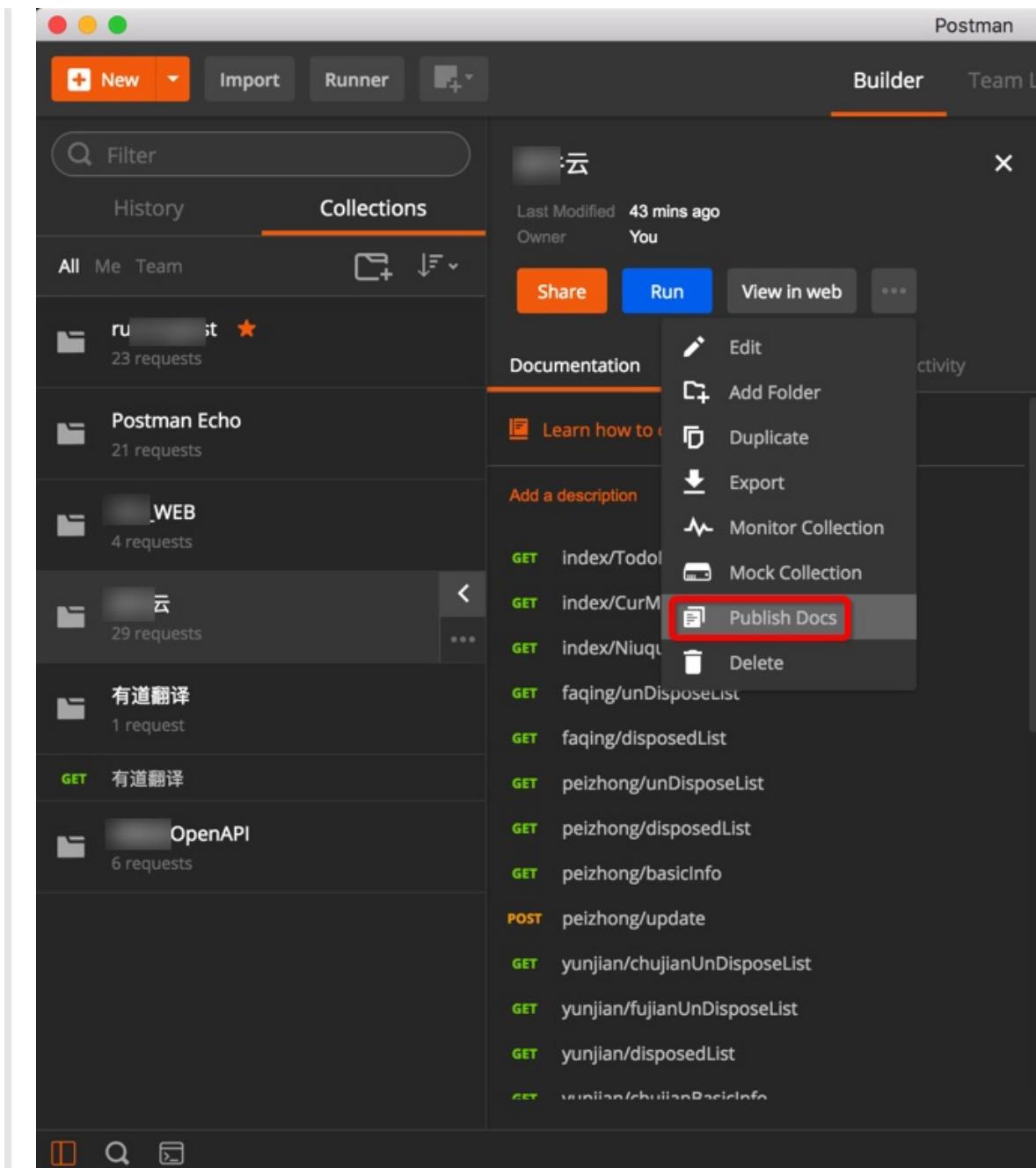
点击分组右边的大于号>



如果只是预览，比如后台开发员自己查看API文档的话，可以选择：View in web



等价于点击Publish Docs去发布：



View in Web后，有 Publish的选项（见后面的截图）

View in Web后，会打开预览页面：

比如：

奶牛云

<https://documenter.getpostman.com/collection/view/669382-42273840-6237-dbae-5455-26b16f45e2b9>

The screenshot shows the Postman application interface. On the left, there's a sidebar with a list of API endpoints categorized under 'Introduction'. The main area displays two API requests:

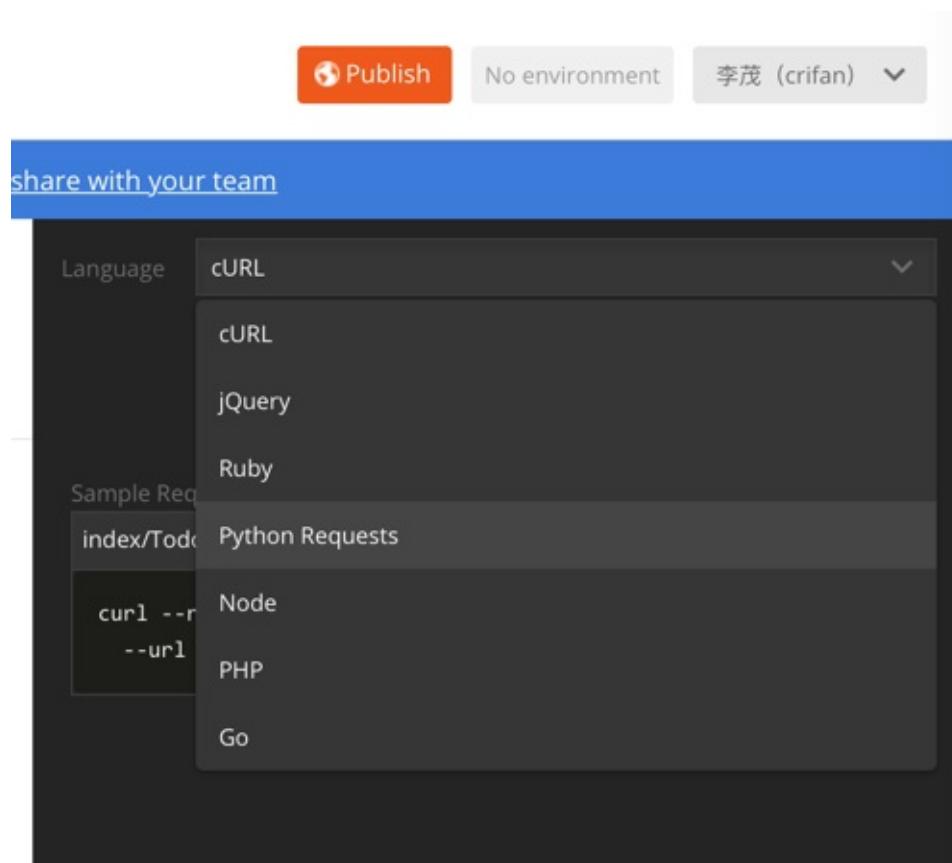
- GET index/TodoNum**: Shows a sample request URL: `http://116.57/ucows/index/index/TodoNum`. To its right is a cURL command: `curl --request GET \ --url http://116.62.25.57/ucows/index/index/TodoNum`.
- GET index/CurMonthTodoNum**: Shows a sample request URL: `http://116.57/ucows/index/index/CurMonthTodoNum`. To its right is a cURL command: `curl --request GET \ --url http://116.62.25.57/ucows/index/index/CurMonthTodoNum`.

Below these, another section shows a POST request for **peizhong/update**:

- POST peizhong/update**: Shows a sample request URL: `http://116.57/ucows/peizhong/peizhong/update`. To its right is a cURL command with JSON data:

```
curl --request POST \
--url http://116.57/ucows/peizhong/peizhong/update \
--header 'authorization: Bear xxx xxx' \
--header 'content-type: application/json' \
--data '{
  "id": "88",
  "cow_code": "16-6963",
  "isPeizhong": "1",
  "gongniucow_id": "101202303",
  "jingyenum": 100,
  "peizhong_date": "2017-06-30",
  "isSexControl": 1,
  "peizhongyuan": "101"
}'
```

而右边的示例代码，也可以从默认的cURL换成其他的：



The screenshot shows the Postman application interface with the following details:

- Left Sidebar (API Endpoints):**
 - GET yunjian/fujianUnDisposeList
 - GET yunjian/disposedList
 - GET yunjian/chujianBasicInfo
 - GET yunjian/fujianBasicInfo
 - GET exception/unDisposeList
 - GET exception/disposedList
 - GET cow/search/list
 - GET cow/basicInfoByCowCode
 - GET cow/basicInfoByBd
 - POST cow/updateBasicInfo
 - GET cowshed/search/list
 - GET emp/empList/14 or /15
 - GET equipment/list
 - GET dict/fanzhiStatusList
 - POST login
 - GET report/niuqunStructure
 - GET dict/queryGongnju
 - GET equipment/equipment/checkCowBBind
 - POST login/login
- Middle Section (API Requests):**
 - GET index/TodoNum**
 - http://116 .57/ucows/index/index/TodoNum
 - http://116 .57/ucows/index/index/TodoNum
- Right Section (Code Samples):**
 - Sample Request: index/TodoNum**

```
import requests

url = "http://116 .57/ucows/index/index/TodoNum"

response = requests.request("GET", url)

print(response.text)
```

 - Sample Request: index/CurMonthTodoNum**

```
import requests

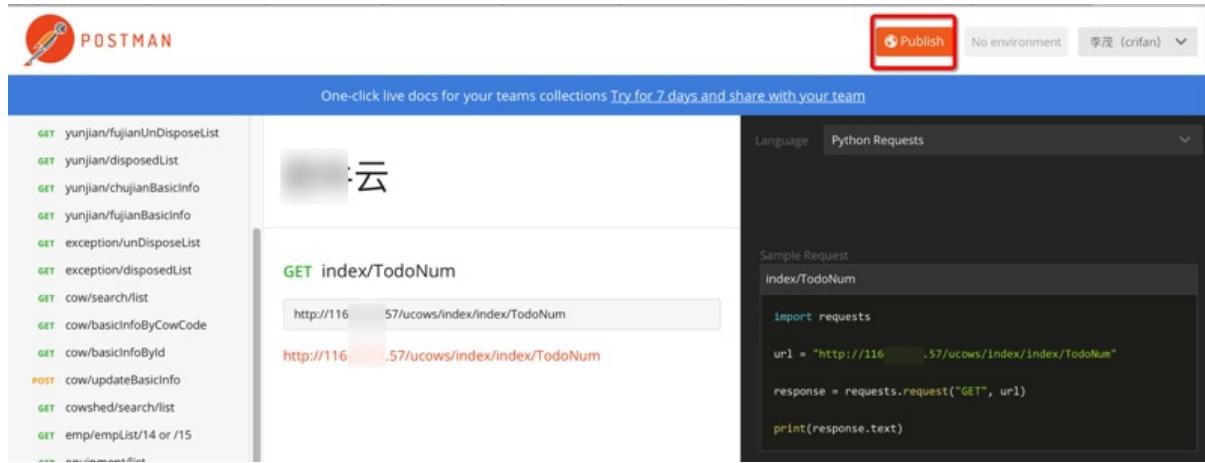
url = "http://116 .57/ucows/index/index/CurMonthTodoNum"

response = requests.request("GET", url)

print(response.text)
```

发布API文档

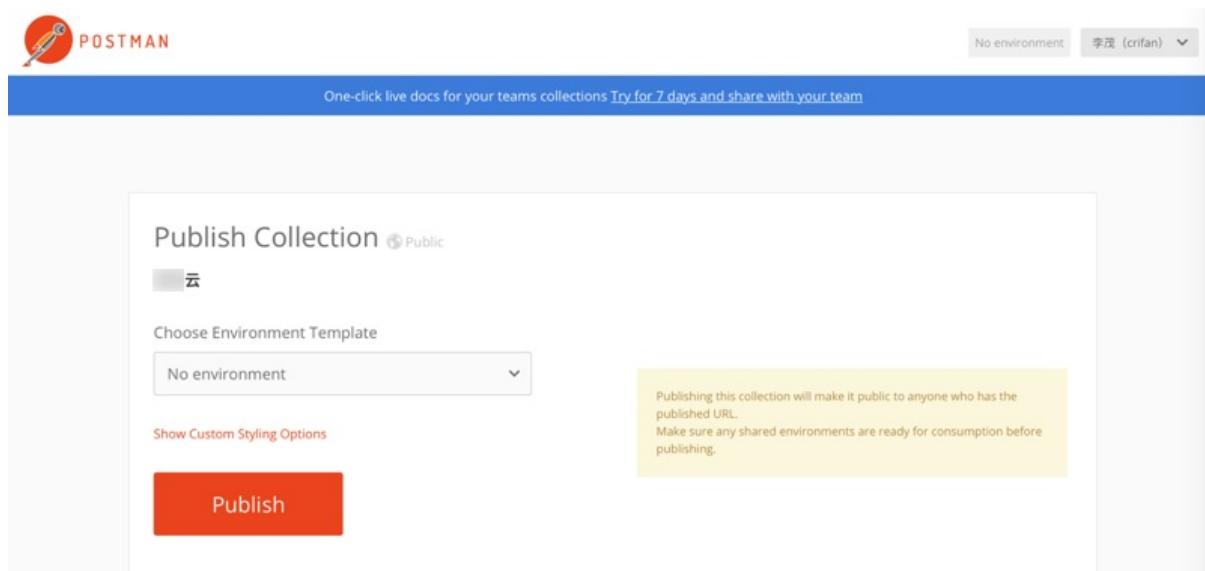
如果想要让其他人能看到这个文档，则点击 Publish：



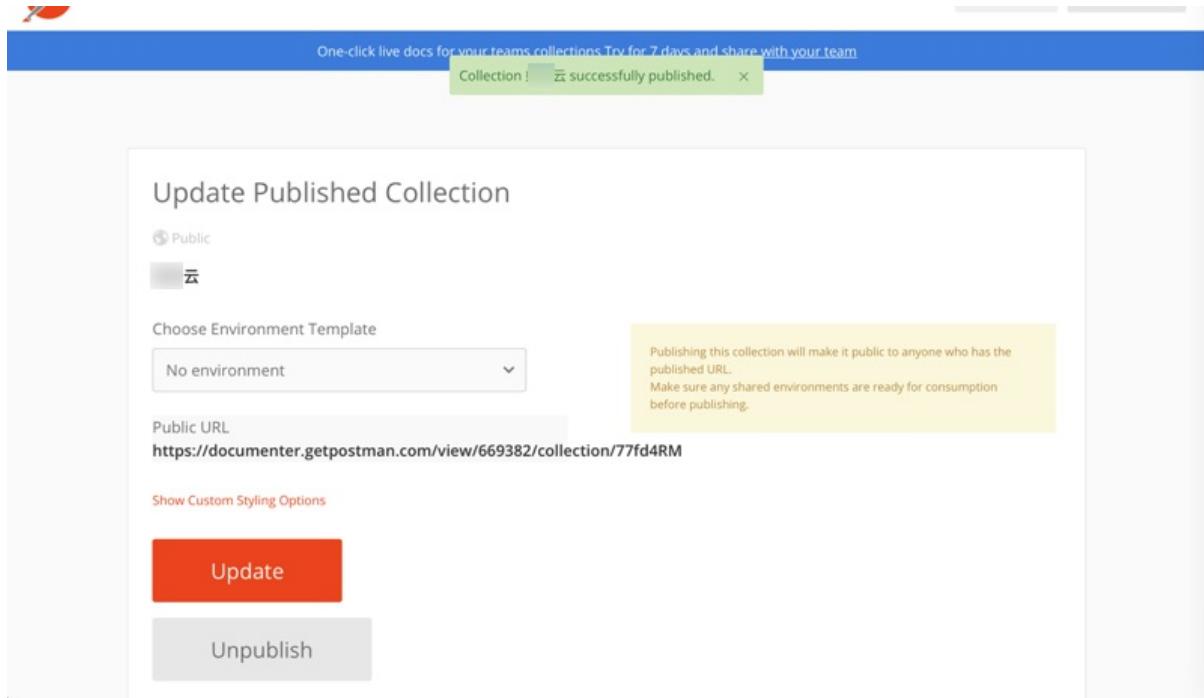
然后会打开类似于这样的地址：

Postman Documenter

[https://documenter.getpostman.com/collection/publish?](https://documenter.getpostman.com/collection/publish?meta=Y29sbGVjdG1vb19pZD00MjI3Mzg0MC02MjM3LWRiYWUtNTQ1NS0yNmIxNmY0NWUyYjkmb3duZXI9NjY5MzgyJmNvbGx1Y3Rpb25fbmFtZT01RTU1QTU1QjY1R)



点击Publish后，可以生成对应的公开的网页地址：



打开API接口文档地址：

<https://documenter.getpostman.com/view/669382/collection/77fd4RM>

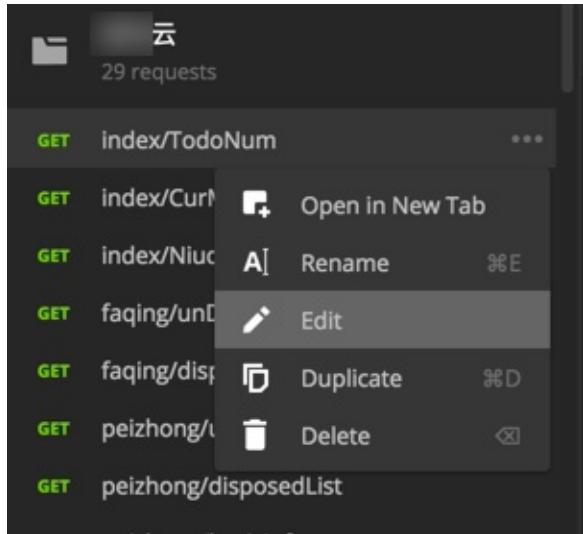
即可看到（和前面预览一样效果的API文档了）：

如此，别人即可查看对应的API接口文档。

已发布的API文档支持自动更新

后续如果自己的API接口修改后：

比如：



A screenshot of the Postman interface with the 'Builder' tab selected. On the left, there's a sidebar with collections and a list of requests. In the center, a modal window titled 'EDIT REQUEST' is open for the endpoint 'index/TodoNum'. The 'Name' field contains 'index/TodoNum'. The 'Description' field contains the text '测试API接口文档是否更新' and a URL 'http://116.57/ucows/index/index/TodoNum'. At the bottom right of the modal are 'Cancel' and 'Update' buttons.

(后来发现，不用再去进入此预览和发布的流程，去更新文档，而是Postman自动支持)

别人去刷新该文档的页面：

<https://documenter.getpostman.com/view/669382/collection/77fd4RM>

即可看到更新后的内容：

The screenshot shows the Postman application interface. On the left, there's a sidebar titled 'Introduction' listing various API endpoints. The main area displays two API requests:

- GET index/TodoNum**
URL: `http://116.17.57.ucows/index/index/TodoNum`
A note says: 测试API接口文档是否更新
Another URL: `http://116.17.57.ucows/index/index/TodoNum`
- GET index/CurMonthTodoNum**
URL: `http://116.17.57.ucows/index/index/CurMonthTodoNum`
A note says: 测试API接口文档是否更新
Another URL: `http://116.17.57.ucows/index/index/CurMonthTodoNum`

On the right, there are two code snippets for Python Requests:

- Sample Request: index/TodoNum**

```
import requests
url = "http://116.17.57.ucows/index/index/TodoNum"
response = requests.request("GET", url)
print(response.text)
```
- Sample Request: index/CurMonthTodoNum**

```
import requests
url = "http://116.17.57.ucows/index/index/CurMonthTodoNum"
response = requests.request("GET", url)
print(response.text)
```

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2017-12-29 21:24:55

附录

下面列出相关参考资料。

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2017-12-14 11:21:48

参考资料

- Manage environments
- postman-变量/环境/过滤等 - 简书
- Postman使用手册3——环境变量 - 简书
- [postman使用之四：切换环境和设置读取变量 - 乔叶叶 - 博客园](#)
-

crifan.com, 使用[知识署名-相同方式共享4.0协议](#)发布 all right reserved, powered by Gitbook该文件修订时间: 2018-01-02 11:23:28