> 《Zabbix企业级分布式监控系统第2版》随书代码
>
> 代码仓库地址 https://github.com/zabbix-book/zabbix_v2
>
> 书籍购买地址 https://item.jd.com/12653708.html

531页

```
{"method": "getid", "params": ["arg"], "id": 1}
```

```
{
    "result":       "id  is  000",
    "error":        null,
    "id":             1
}
```

表13-1　Zabbix API支持的数据类型

类　　型	说　　明
bool	布尔值为true或者false
flag	当该值不等于空或者false时，被认为是true
integer	整数
float	浮点数
string	文本字符串
timestamp	UNIX 时间戳
array	数组
object	关联数组
query	可以是一个数值，也可以是部分参数· extend：返回所有的对象值· count：返回值的数量

532页

表13-2　查询操作（get方法）支持的参数

参　　数	类　　型	描　　述
countOutput	flag	返回结果的个数，而非实际的数据
editable	boolean	如果设置为true，用户可对返回的对象进行写操作。默认值为false
excludeSearch	flag	返回不匹配给定参数的结果
filter	object	返回过滤后的结果，参数的值可以为一个数组或者单个值，text字段不能使用此参数
limit	integer	限制返回结果的数量
nodeids	string/array	返回给定节点的对象信息
output	query	返回对象的属性。默认值为extend
preservekeys	flag	返回以ID为键的数组
search	object	搜索匹配的字符串，仅用于字符和文本字段
searchByAny	boolean	如果设置为true，则返回 filter 或search字段中所有的值。默认值为false
searchWildcardsEnabled	boolean	如果设置为true，允许使用“*”作为搜索参数的通配符。默认值为false
sortfield	string/array	对给定的参数属性进行排序
sortorder	string/array	排序。ASC为升序排列；DESC为降序排列
startSearch	flag	搜索以某个参数开始的结果

533页

https://www.zabbix.com/documentation/4.0/manual/api/reference/host/get

```
{
    "jsonrpc": "2.0",
    "method": "host.get",
    "params": {
        "filter": {
            "host": [
                "Zabbix server",
                "Linux server"
            ]
        }
    },
    "auth": "038e1d7b1735c6a5436ee9eae095879e",
    "id": 1
```

```
}
```

534页

```
shell# curl -X POST     \
-H 'Content-Type:application/json' \
-d  '{"jsonrpc": "2.0",
    "method":"user.login",
    "params":{"user":"Admin",
                "password":"zabbix"},
    "auth": null,
    "id":0
    }'  \
http://127.0.0.1/zabbix/api_jsonrpc.php
```

得到的结果如下：

```
{
    "jsonrpc":"2.0",
    "result":"b545519b2e2f7b08e1a63b360f24b533",
    "id":0
}
```

```
mysql> select * from sessions where
sessionid="b545519b2e2f7b08e1a63b360f24b533";
+----------------------------------+--------+------------+--------+
| sessionid                        | userid | lastaccess | status |
+----------------------------------+--------+------------+--------+
| b545519b2e2f7b08e1a63b360f24b533 |    1   | 1539053137 |    0   |
+----------------------------------+--------+------------+--------+
```

```
shell# curl -X  POST  \
-H  'Content-Type: application/json'  \
-d '{"jsonrpc":"2.0",
     "method":"host.get",
     "params":{
         "output":"extend",
         "filter":{"host":""}
     },
     "auth":"b545519b2e2f7b08e1a63b360f24b533",
     "id":1
    }' \
http://127.0.0.1/zabbix/api_jsonrpc.php
```

535页

```
{
    "jsonrpc":"2.0",
    "result":[
        {
             "maintenances":[],
              "hostid":"10084",
              "proxy_hostid":"0",
              "host":"Zabbix server",
              "status":"0",
              "disable_until":"0",
              "error":"",
              "available":"1",
              "errors_from":"0",
              "lastaccess":"0",
              "ipmi_authtype":"-1",
              "ipmi_privilege":"2",
              "ipmi_username":"",
              "ipmi_password":"",
              "ipmi_disable_until":"0",
              "ipmi_available":"0",
              "snmp_disable_until":"0",
              "snmp_available":"0",
              "maintenanceid":"0",
              "maintenance_status":"0",
              "maintenance_type":"0",
              "maintenance_from":"0",
              "ipmi_errors_from":"0",
              "snmp_errors_from":"0",
              "ipmi_error":"",
              "snmp_error":"",
              "jmx_disable_until":"0",
              "jmx_available":"0",
              "jmx_errors_from":"0",
              "jmx_error":"",
              "name":"Zabbix server"
        },
        ......省略部分输出......
     ],
    "id":1
}
```


536页

```
shell# curl -i  -X POST  -d  '
    {"jsonrpc": "2.0",
        "method":"user.login",
        "params":{
```

```
            "user":"Admin",
            "password":"zabbix"
        },
        "auth": null,
        "id":0
    }'  http://127.0.0.1/zabbix/api_jsonrpc.php
#可以看到HTTP返回了412错误
HTTP/1.0 412 Precondition Failed
Date: Tue, 09 Oct 2018 03:07:17 GMT
Server: Apache/2.4.6 (CentOS) PHP/5.4.16
X-Powered-By: PHP/5.4.16
Access-Control-Allow-Origin: *
Access-Control-Allow-Headers: Content-Type
Access-Control-Allow-Methods: POST
Access-Control-Max-Age: 1000
Content-Length: 0
Connection: close
Content-Type: text/html; charset=UTF-8
```

认证成功的提示内容如下：

```
{
    "jsonrpc":"2.0",
    "result":"b545519b2e2f7b08e1a63b360f24b533",
    "id":0
}
```

认证失败的提示内容如下：

```
{
    "jsonrpc":"2.0",
    "error":
        {
            "code":-32602,
            "message":"Invalid params.",
            "data":"Login name or password is incorrect."
         },
    "id":0
}
```

537页

https://github.com/zabbix-book/zabbix_api_example/blob/master/zabbix_api_host_get.py

```
 #!/usr/bin/env python
2. #coding=utf-8
3.
```

```
#导入模块，urllib2是一个模拟浏览器HTTP方法的模块
import json
import urllib2
import sys
from urllib2 import Request, urlopen, URLError, HTTPError

#url和url头部信息 [要写成中文注释吗？下同]
#Zabbix的API 地址、用户名、密码，这里修改为实际的参数
zabbix_url="http://127.0.0.1/zabbix/api_jsonrpc.php"
zabbix_header = {"Content-Type":"application/json"}
zabbix_user    = "Admin"
zabbix_pass    = "zabbix"
auth_code      = ""

#用户和密码
#用户认证信息，最终目的是得到一个sessionid
#下面生成一个JSON格式的数据，用户名和密码
auth_data = json.dumps(
    {
        "jsonrpc":"2.0",
        "method":"user.login",
        "params":
            {
                "user":zabbix_user,
                "password":zabbix_pass
            },
            "id":0
    })

#构造请求数据
request = urllib2.Request(zabbix_url,auth_data)
for key in zabbix_header:
    request.add_header(key,zabbix_header[key])


#认证和获取sessionid
try:
    result = urllib2.urlopen(request)
#对认证出错的处理
except HTTPError, e:
    print 'The server couldn\'t fulfill the request, Error code: ', e.code
except URLError, e:
    print 'We failed to reach a server.Reason: ', e.reason
else:
    response=json.loads(result.read())
    result.close()
'''
    #如果访问成功或者失败，则会显示如下数据
    sucess result:
```

```
52.         {"jsonrpc":"2.0",
53.          "result":"0d225d8d2a058625f814f3a0749cd218",
54.          #result后面的值是sessionid，每次访问都会发生变化
55.          "id":0}
56.     error  result:
57.         {'code': -32602,
58.          'data': 'Login name or password is incorrect.',
59.          'message': 'Invalid params.'}
60. '''
61.     #判断sessionid是否在返回的数据中
62.     if  'result'  in  response:
63.         auth_code=response['result']
64.     else:
65.         print  response['error']['data']
66.
67. #请求数据
68. json_data={
69.     "method":"host.get",
70.     "params":{
71.         "output": "extend",
72.     }
73. }
74. json_base={
75.     "jsonrpc":"2.0",
76.     "auth":auth_code,
77.     "id":1
78. }
79. json_data.update(json_base)
80. #用得到的sessionid来验证，获取主机信息（用http.get方法）
81. if len(auth_code) == 0:
82.     sys.exit(1)
83. else:
84.     get_host_data = json.dumps(json_data)
85.     #发送请求数据
86.     request = urllib2.Request(zabbix_url,get_host_data)
87.     for key in zabbix_header:
88.         request.add_header(key,zabbix_header[key])
89.
90.     #获取主机列表数据
91.     try:
92.         result = urllib2.urlopen(request)
93.     except URLError as e:
94.         if hasattr(e, 'reason'):
95.             print 'We failed to reach a server.'
96.             print 'Reason: ', e.reason
97.         elif hasattr(e, 'code'):
98.             print 'The server could not fulfill the request.'
99.             print 'Error code: ', e.code
100.     else:
```

```
101.         response = json.loads(result.read())
102.         result.close()
103.
104.         #将所有的主机信息显示出来
105.         print response
106.         #显示主机的个数
107.         print "Number Of Hosts: ", len(response['result'])
```

540页

```
shell# python zabbix_api_host_get.py
{
    "available":"1",
    "maintenance_type":"0",
    "snmp_errors_from":"0",
    "hostid":"10084",
    "description":"",
    "ipmi_errors_from":"0",
    "jmx_errors_from":"0",
    "error":"",
    "tls_accept":"1",
    ......省略部分输出......
    "jmx_error":"",
    "jmx_available":"0",
    "maintenanceid":"0",
    "proxy_address":"",
    "snmp_available":"0",
    "flags":"0",
    "maintenance_from":"0",
    "ipmi_error":"",
    "errors_from":"0",
    "tls_subject":""
}
```

https://www.zabbix.com/documentation/4.0/manual/api/reference/host/create

官方文档中列出了host.create的用法，如下所示。

```
{'
    auth': '038e1d7b1735c6a5436ee9eae095879e',
    'id': 1,
    'jsonrpc': '2.0',
    'method': 'host.create',
    'params': {'groups': [{'groupid': '50'}],
            'host': 'Linux server[是否是：servers]'[此处需要复数],
```

```
            'interfaces': [{'dns': '',
                            'ip': '192.168.3.1',
                            'main': 1,
                            'port': '10050',
                            'type': 1,
                            'useip': 1}],
            'inventory': {'macaddress_a': '01234', 'macaddress_b': '56768'},
            'templates': [{'templateid': '20045'}]}
}
```

541页

```
mysql> select * from hosts where host="Template OS Linux"\G;
*************************** 1. row ***************************
            hostid: 10001
   proxy_hostid: NULL
              host: Template OS Linux
            status: 3
   disable_until: 0
......省略部分输出......
           tls_psk:
     proxy_address:
     auto_compress: 1
```

继续查询主机组，查询到Linux servers的groupid为2。

```
mysql> select * from hstgrp;
+---------+-----------------------------+----------+-------+
| groupid | name                        | internal | flags |
+---------+-----------------------------+----------+-------+
|       1 | Templates                   |        0 |     0 |
|       2 | Linux servers               |        0 |     0 |
|       4 | Zabbix servers              |        0 |     0 |
......省略部分输出......
|      24 | log                         |        0 |     0 |
+---------+-----------------------------+----------+-------+
```

https://github.com/zabbix-book/zabbix_api_example/blob/master/zabbix_api_host_create.py

542页

修改代码中的参数值，替换为实际参数值，如下所示。

```
shell# vim zabbix_api_host_create.py
{
    'method': 'host.create',
    'params': {'groups': [{'groupid': '2'}],
```

```
            'host': 'Web Linux server',
            'interfaces': [{'dns': '',
                            'ip': '192.168.8.1',
                            'main': 1,
                            'port': '10050',
                            'type': 1,
                            'useip': 1}],
            'inventory': {'macaddress_a': '01234', 'macaddress_b': '56768'},
            'templates': [{'templateid': '10001'}]}
}
```

运行代码，结果显示主机创建成功，hostid为10284（后续会用到）。

```
shell# python zabbix_api_host_create.py
{
    'jsonrpc': '2.0',
    'result': {'hostids[是否多了一个s][正确]': ['10284']},
    'id': 1
}
```

543页

https://www.zabbix.com/documentation/4.0/manual/api/reference/host/delete

```
{
    'jsonrpc': '2.0',
    'method': 'host.delete',
    'params': [
        '13',
        '32'
    ],
    'auth': '038e1d7b1735c6a5436ee9eae095879e',
    'id': 1
}
```

https://github.com/zabbix-book/zabbix_api_example/blob/master/zabbix_api_host_delete.py

```
shell# vim zabbix_api_host_delete.py
json_data={
        "method":"host.delete",
        "params":['10284']
}
```

运行脚本，如下所示。

```
shell# python zabbix_api_host_delete.py
{
    u"jsonrpc": "2.0",
    u"result": {
```

```
        u"hostids": [
            u"10284"
        ]
    },
    u"id": 1
}
```

544页

https://www.zabbix.com/documentation/4.0/manual/api

https://zabbix.org/wiki/Docs/api/libraries

```
shell# pip  install  pyzabbix #如图13-3所示
```

545页

```
import logging
import requests
import json

class _NullHandler(logging.Handler):
    def emit(self, record):
        pass

logger = logging.getLogger(__name__)
logger.addHandler(_NullHandler())

class ZabbixAPIException(Exception):
    pass

class ZabbixAPI(object):
    def __init__(self,server='http://localhost/zabbix',session=None,use_
authenticate=False,timeout=None): #构造URL
        if session:
            self.session = session
        else:
            self.session = requests.Session()

        self.session.headers.update({'Content-Type': 'application/json-
rpc','User-Agent': 'python/pyzabbix','Cache-Control': 'no-cache'})

        self.use_authenticate = use_authenticate
        self.auth = ''
        self.id = 0
        self.timeout = timeout
```

```
        self.url = server + '/api_jsonrpc.php'
        logger.info("JSON-RPC Server Endpoint: %s", self.url)

    def login(self, user='', password=''): #输入用户名和密码
        self.auth = ''
        if self.use_authenticate:
            self.auth = self.user.authenticate(user=user, password=password)
        else:
            self.auth = self.user.login(user=user, password=password)

    def confimport(self, confformat='', source='', rules=''):
        return self.do_request(method="configuration.import",params= {"format":
confformat, "source": source, "rules": rules})['result']

    def api_version(self):
        return self.apiinfo.version()

    def do_request(self, method, params=None): #使用API方法
        request_json = {'jsonrpc': '2.0','method': method,'params': params or
{},'id': self.id,}

        if self.auth and method != 'apiinfo.version':
            request_json['auth'] = self.auth

             logger.debug("Sending: %s", json.dumps(request_json,indent=4,
separators=(',', ': ')))
        response = self.session.post(self.url,data=json.dumps(request_json),
timeout=self.timeout)
        logger.debug("Response Code: %s", str(response.status_code))

        response.raise_for_status()

        if not len(response.text):
            raise ZabbixAPIException("Received empty response")

        try:
            response_json = json.loads(response.text)
        except ValueError:
            raise ZabbixAPIException( "Unable to parse json: %s" %
response.text)
        logger.debug("Response Body: %s", json.dumps(response_json,
indent=4,separators=(',', ': ')))

        self.id += 1

        if 'error' in response_json:  # some exception
            if 'data' not in response_json['error']:
                response_json['error']['data'] = "No data"
            msg = u"Error {code}: {message}, {data}".format(
```

```
                code=response_json['error']['code'],
                message=response_json['error']['message'],
                data=response_json['error']['data']
            )
            raise ZabbixAPIException(msg,response_json['error']['code'])
        return response_json

    def __getattr__(self, attr):
        """Dynamically create an object class (ie: host)"""
        return ZabbixAPIObjectClass(attr, self)

class ZabbixAPIObjectClass(object):
    def __init__(self, name, parent):
        self.name = name
        self.parent = parent

    def __getattr__(self, attr):
        """Dynamically create a method (ie: get)"""
        def fn(*args, **kwargs):
            if args and kwargs:
                raise TypeError("Found both args and kwargs")
            #将zapi.host.get(params)解析为do_request函数中的{'jsonrpc':
'2.0','method': 'host.get','params': params or {},'id': self.id,}
            return self.parent.do_request(
                '{0}.{1}'.format(self.name, attr),
                args or kwargs
            )['result']
        return fn
```

更多的例子，参见https://github.com/lukecyca/pyzabbix/tree/master/example。

547页

https://github.com/zabbix-book/zbx_tool

```
shell# zbx_tool
usage: zbx_tool [options]

Zabbix Automation Tools

optional arguments:
  -h, --help            show this help message
[代码输出的文字，无需翻译]                                显示帮助信息
  -H [LISTHOST], --host [LISTHOST]
                        查询主机
  -G [LISTGROUP], --group [LISTGROUP]
                        查询主机组
  -T [LISTTEMP], --template [LISTTEMP]
                        查询模板信息
```

```
  -A ADDGROUP, --add-group ADDGROUP
                        添加主机组
  -C 10.10.10.1 HostName group01,group02 Template01,Template02 ["agent",
"SNMP", "IPMI", "JMX"] "MACROS1_NAME,MACROS1_VALUE;MACROS2_NAME,MACROS2_VALUE",
--add-host 10.10.10.1 HostName group01,group02 Template01,Template02 ["agent",
"SNMP", "IPMI", "JMX"] "MACROS1_NAME,MACROS1_VALUE;MACROS2_NAME,MACROS2_VALUE"
                        添加主机，多个主机或模板之间使用分号分隔
  -d 10.10.10.1, --disable 10.10.10.1
                        禁用主机
  -D HostName [HostName ...], --delete HostName [HostName ...][是否删除? 下同]
                        删除主机，多个主机之间用分号分隔
  -SH hostname screename hsize, --add-host-all hostname screename hsize
                        将一个主机的所有图形添加到screen
  -SG groupname graphname screenname hsize, --add-group-graph groupname
graphname screenname hsize
                        将一个组的指定图形添加到screen
  -SN graphname screenname hsize, --add-graph-search graphname screenname hsize
                        将指定的图形添加到screen
  -v, --version         show program's version number[中文意思? ]
显示查询版本
```