**邮件和JSON**

邮件编程

SMTP概述

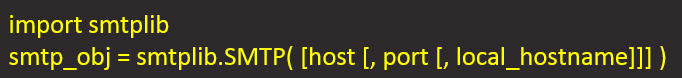
SMTP即简单邮件传输协议，使用TCP协议25端口

它是一组用于由源地址到目的地址传输邮件的规则，由它来控制信件的中转方式

python的smtplib提供了一种很方便的途径发送电子邮件。它对smtp协议进行了简单的封装

SMTP对象

python发送邮件，第一步是创建SMTP对象



创建SMTP对象也可以不给定参数，之后再通过对象的其他方法进行绑定

设置邮件

标准邮件需要三个头部信息

FROM：发件人

TO：收件人

subject：主题



sendmail方式

python smtp对象使用sendmail方法发送邮件



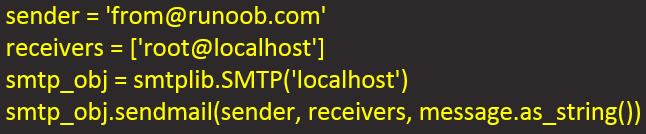
sendmail方法三个必须的参数有

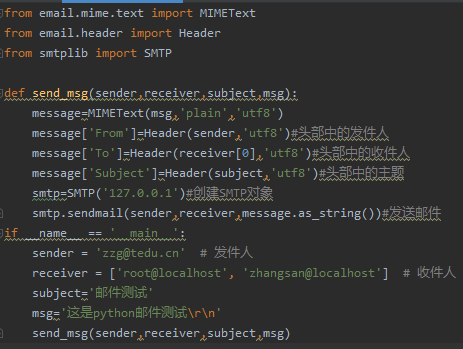
收件人

发件人

消息主题msg是一个字符串，表示邮件

将准备好的邮件发送

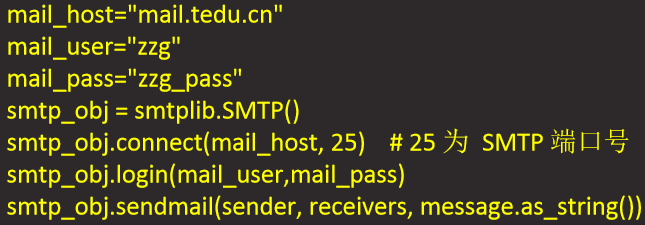




SMTP认证

如果本机没有smtp功能，也可以使用第三方的邮件服务器

第三方邮件服务器往往需要认证



JSON

JSON概述

JSON是一种轻量级的数据交换格式

易于人阅读和编写，同时也易于机器解析和生成

基于javascript programming language

json采用完全独立于语言的文本格式，但是也使用了类似于C语言家族的习惯（包括C，C++等）

这些特性使JSON成为理想的数据交换语言

JSON结构

JSON主要有两种结构

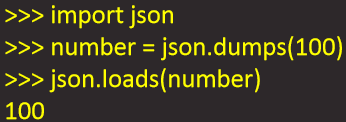
键/值对的集合：python中主要对应成字典

值的有序列表：在大部分语言中，它被理解为数组

|  |  |
| --- | --- |
| Python | JSON |
| dict | object |
| list,tuple | array |
| str | string |
| int,float | number |
| True | true |
| False | false |
| None | null |

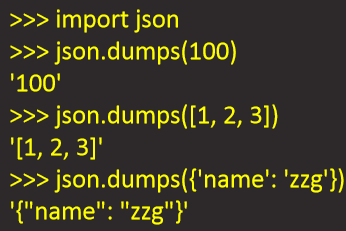
dumps方法

对编码后的json对象进行decode解码，得到原始数据，需要使用的json.loads()函数



loads方法

使用简单的json.dumps方法对简单的数据类型进行编码





**requests模块**

requests基础

requests简介

requests是用python语言编写的、优雅而简单的http库

requests内部采用urillib3

requests使用起来比urllib3更简单快捷

requests需要单独安装



requests特性

支持keep-alive的连接池

支持通用的域名以及URL地址

支持使用cookie

支持使用类似浏览器的SSL验证

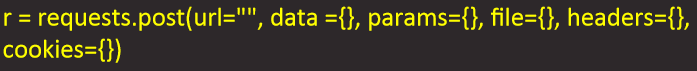
文件上传、下载

GET和POST

通过requests发送一个GET请求，需要在URL里请求的参数可通过params传递

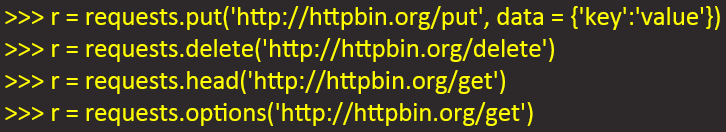


与GET不同的是，POST请求新增了一个可选参数data，要要通过POST请求传递的body里的数据可以通过data传递



其他方法

其他HTTP请求类型：PUT，DELETE，HEAD以及OPTIONS使用起来一样简单



请求参数

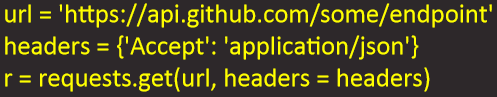
当访问一个URL时，我们经常需要发送一些查询的字段作为过滤信息，例如：httpbin.com/get?key=val,这里的key=val就是限定返回条件的参数键值对

当利用python的requests去发送一个需要包含这些参数键值对时，可以将它们传给params



设定头部

用户也可以自己设定请求头



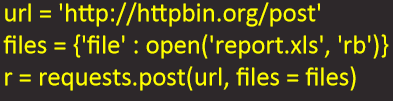
发送请求数据

有时候，用户需要将一些数据放在请求的body内；这时候，就需要对data传参了（仅POST，DELETE，PUT等方法有该参数，GET没有，因为GET请求没有body）



传递文件

可以通过requests传一些文件，使用的是file参数



requests应用

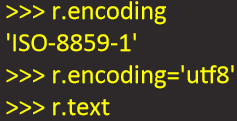
响应内容

读取服务器响应的内容



请求发出后，requests会基于HTTP头部对响应的编码作为有根据的推测

可以找出requests使用了什么编码，并且能够使用r.encodeing属性来改变它



其他响应内容格式

也可以用字节的方式访问请求响应体，尤其是非文本请求（如图片）

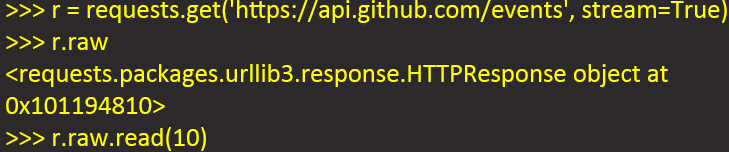


requests中还有一个内置的JSON解码器，助你处理数据



在罕见的情况下，可能想获取来自服务器的原始套接字响应，那么可以访问r.raw

如果确实想这么干，那请确保在初始请求中设置了stream=True



响应状态码

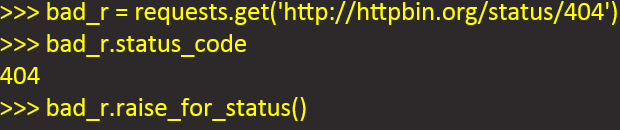
检测响应状态码



为方便引用，requests还附带了一个内置的状态码查询对象

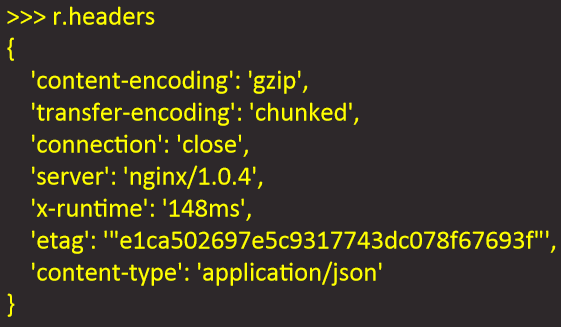


如果发送了一个错误请求（一个4XX客户端错误，或者5XX服务器错误相应），可以通过response.raise\_for\_status()来抛出异常



响应头

可以产看以字典形式展示的服务器响应头



cookie

http是一种无状态的请求/响应协议，用户通过浏览器访问web站点后，Web服务端没有可用信息来判断是哪个用户发起的请求，更加无法知道下次访问的还是不是上次访问的用户，无法识别当前用户

cookie的设计实现很好的解决了这个问题。用户通过浏览器访问web站点后，服务端会将一些key/value组合的键值对通过set-cookie或set-cookie2返回给浏览器，用户再次访问web站点时浏览器会将符合条件的键值对再发送给服务端，这样服务端就可以通过这个键值信息识别出当前用户

cookie可以分为两类：会话cookie和持久cookie

会话cookie是一种临时cookie，没有设置它的有效时间，当用户退出的时候，它将会被删除

当设置了cookie的有效期后，它就是持久cookie，它可以被存储到硬盘上，当用户退出浏览器或机器重启时，它依然存在，可以被再次读取使用

cookie的版本

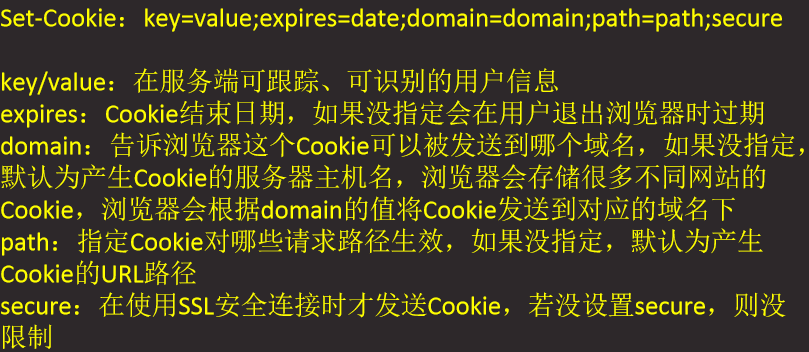
当前可使用的cookie规范有两个版本：cookie版本0和cookie版本1

cookie版本1是对0的扩展，版本1可以和版本0互操作，但是cookie版本1没有cooie版本0使用的广泛

版本0定义了set-cookie相应头部、cookie请求首部

set-cookie相应头部，其实就是服务端返回的

cookie信息，具体的语法如下



使用cookie



**zabbix编程**

zabbix api简介

部署zabbix

zabbix是一个基于web界面的提供分布式系统监视以及网络监视功能的企业级的开源解决方案

能监视各种网络参数，保证服务器系统的安全运营；并提供灵活的通知机制以让系统管理员快速地位/解决存在的各种问题

部署方式参见云计算监控课程

zabbix api概述

zabbix api允许以编程方式检索和修改zabbix的配置，并提供对历史数据的访问。它广泛用于

创建新的应用程序以使用zabbix

将zabbix与第三方软件集成

自动执行常规任务

JSON-RPC

zabbix api是基于web的api，作为web前端的一部分提供。它使用JSON-RPC2.0协议，这意味着两件事

该api保函一组独立的方法

客户端和API之间的请求和响应使用JSON格式进行编码

API结构

zabbix api保函许多方法，这些方法都名义上分组为单组的API

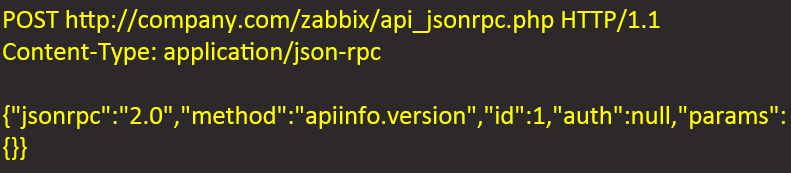
每个方法执行一个特定的任务。例如，方法host.create隶属于host这个API，用于创建新主机

历史上，API有时被称为“类”

大多数API至少包含四种方法：get，create，update和delete，分别是检索，创建，更新和删除数据。但是某些API提供完全不同的一组方法。

执行请求

设置前端后，就可以使用远程HTTP请求来调用api。为此，需要向api\_jsonrpc.php位于前端目录中的文件发送HTTP POST请求。例如，如果zabbix前端安装在http://comangy.com/zabbix，那么用http请求来调用apiinfo.version方法就如下面这样：



请求的content-type头部设置为以下值之一：

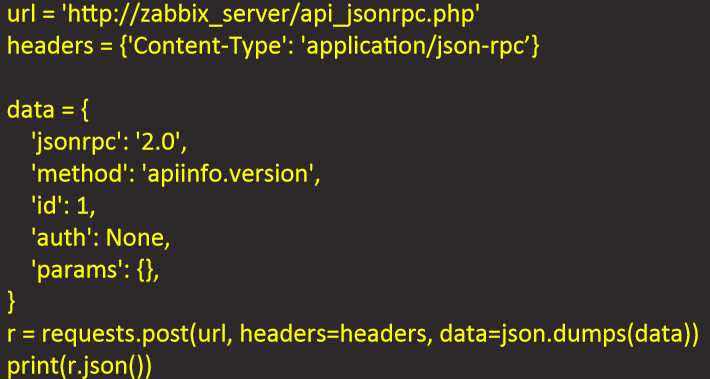
application/json-rpc

application/json

application/jsonrequest

使用API

通过zabbix提供的API接口，就可以使用python与其交互了



方法参考

工作流程

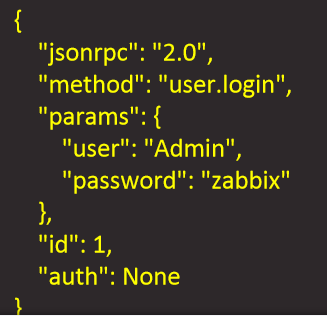
在访问大多数zabbix中的任何数据之前，需要登录并获取身份验证令牌

取得令牌后，访问其他数据只要出示该令牌即可，不需要再进行身份验证

通过zabbix api提供的各种方法实现数据的检索、项目的创建等

获取令牌

使用user.login方法登录并获取身份验证令牌



jsonrpc-api使用的JSON-RPC协议的版本；zabbix api实现JSON-RPC版本2.0

method-调用的API方法

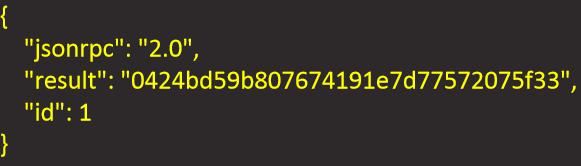
params-将被传递给API方法的参数

id-请求的任意标识符

auth-用户认证令牌；因为还没有一个，它的设置None

令牌响应信息

如果正确提供了凭据，API返回的响应将包含用户身份验证令牌



响应对象包含以下属性：

jsonrpc-JSON-RPC协议的版本

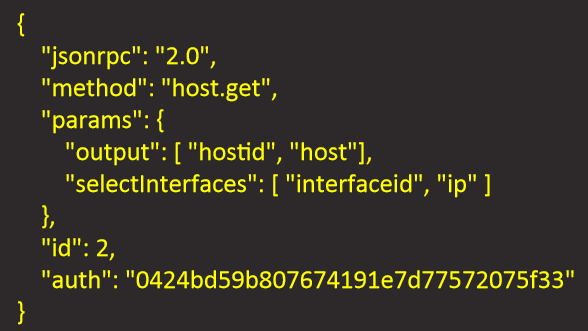
result方法返回的数据

id响应请求的标识符

检索主机

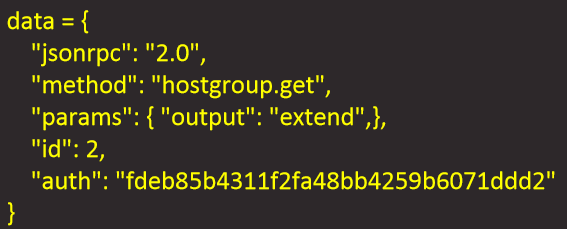
有一个有效的用户身份验证令牌，可以用来访问Zabbix中的数据

使用host.get方法检索所有已配置主机的ID，主机名和接口。auth属性设置为获取的身份验证令牌



获取主机组

获取主机组的方法与检索主机一样，只要修改请求数据即可



创建主机

创建主机操作与获取信息操作完全一样，只是传递的请求参数不一样

