OpenAPI 的签名规则

为方便用户对接金易联，梳理金易联的一些开放接口，提供给客户对接使用，以OpenAPI的形式暴露给客户使用；

请求公共参数

每个调用OpenAPI请求都必须包含以下公共参数，如果请求参数不合理，该请求会被拒绝。

当请求的HTTP Method为GET时，公共参数以URL中的“query string"的格式输入；当请求的HTTP Method为POST时，公共参数以在请求体中的“x-www-form-urlencoded”格式或者URL中的“query string”格式输入。

公共参数名	说明	示例
key	api key，唯一代表一个client 身份	2762aee5-4fa8-437e-85af-1dbfbe466298
ts	请求的时间戳，格式为ISO8601格式，精确到毫秒。如果不带时区，默认为＋0800，即北京时间	2015-08-29T12:31:24.556
nonce	. 832	zXwagyl3ksf
sigVer	“1”	1
sig	请求签名，产生方式详见下面几节	heBO3tbI1FHfhvt5x5cpswMlsCE=

请求签名如何使用

签名的使用方式是：

1. 对接的client在请求被发送前根据请求数据和api secret产生一个密钥，并作为请求参数`sig`的值。sig将会和其他参数一起发给服务器；
2. 服务器端收到请求后，根据请求数据和内部保存的api secret重新计算一个签名，并将其与请求中`sig`参数值比较，以决定该请求是否合法。

步骤1 获取HTTP方法名和URL Path

如果请求HTTP Method为Get，则方法名为“GET” （全大写）；如果为Post，则方法名为“POST”（全大写）。

URL Path是指请求URL中的**base URL后，至Query String开始的部分**。

步骤2将所有参数排序和拼接

将请求中query string和body中所有参数放到一起，按照参数名字典升序排序，以"参数名=参数值"格式拼接每一个参数，最后将所有参数用"&"拼接到一起。 这些拼接到一起的参数包括所有公共参数和api特定的参数，但是不包括sig自己（这时sig还没计算出来）。

关于排序

所谓按照参数名字典升序排序，举个例子：

假设有一系列参数的key和value：**key2=v2&key1=v1&key4=v4&key3=v3**

参数名字是：key2，key1，key4，key3，

按照字典序排序后是：key1，key2，key3，key4

排序，然后拼接的参数字符串则是：**key1=v1&key2=v2&key3=v3&key4=v4**

HTTP允许一个参数的值为空，例如"foo=&bar=4"中的foo。尽管此时对于HTTP协议来讲，在query string/body中明确写出"foo="和完全不写foo是等价的，但是对于签名的产生不是等价的。我们规定**如果参数的值为空，则该参数不参与拼接和计算。**

注意这里只是规定空参数**不参与sig的产生**，并不限制实际在调用接口时是否传递空参数。没错，你依然可以在调用接口时传递空参数。

此外，参数拼接过程在client端应当发生在参数被URL Encoded之前；在服务器端应该发生在参数被URL Encoded以后。拼接的参数不需要经过URL Encode，例如符号“&”，“：”或者中文等都应该保持原样。拼接过程与请求被传输过程中的编码过程是两件独立的过程。

例如：创建账户”api请求的参数为:

参数名	参数值
key	2762aee5-4fa8-437e-85af-1dbfbe466298
sigVer	1
nonce	123456789
ts	2015-08-29T12:31:24.556
userId	u12345
accountName	爱丽丝

则拼接的结果为

```
accountName=&userId=u12345&key=2762aee5-4fa8-437e-85af-1dbfbe466298&nonce=123456789&sigVer=1&ts=2015-08-29T12:31:24.556
```

步骤3 产生规范化字符串

将步骤1的HTTP方法名、URL Path和步骤2产生的结果用":"拼接，产生规范化字符串(unifiedString)。

```
unifiedString = {Params}
```

例如创建一个账户的请求拼接后的结果是：

```
accountName=&userId=u12345&key=2762aee5-4fa8-437e-85af-1dbfbe466298&nonce=123456789&sigVer=1&ts=2015-08-29T12:31:24.556
```

步骤4产生签名

将步骤3的结果根据api secret产生一个HMAC Sha1摘要(**注意:摘要应为2进制格式**)，并且以Base64编码产生签名. 将步骤3的结果传入HMAC Sha1 进行计算时，要对字符串进行UTF8编码。Mac和部分Linux操作系统默认会采用UTF8编码，所以开发时无需特别指定；但如果是Windows中文版，则会采用GBK编码。在这种情况下，需要在您使用的开发平台上明确指定使用UTF8进行sig的生成。

```
sig = base64HmacSha1(apiSecret, encode(unifiedString, 'utf-8'))
```

例如，当api secret是"MY3c6h402vU4dZNeHrRVnkP3rVWM4l8Az396Pu3KouAkyWKs", 则步骤3得到的规范化字符串得到的签名为

```
heBO3tbl1FHfhvt5x5cpswMlsCE=
```

Golang签名示例

```
ts := time.Now().Add(1 * time.Minute).Format("2006-01-02T15:04:05.000")
nonce := fmt.Sprintf("ts123%d", time.Now().Nanosecond())

appKey := "V1eSG6lAg6PB4VhJ509AMgPR5OTw0JA"
appSecret := "R0DWiCTJK7ZpHXKOqqZ3I5fyqFarDRE"

body := map[string]interface{}{"appId": appKey, "userId": "u12345678"}
data := map[string]interface{}{"test": "test1", "version": 1}

params := map[string]interface{}{"ts": ts, "key": appKey, "nonce": nonce, "sigVer": "1"}
params["appId"] = body["appId"]
params["userId"] = body["userId"]
params["data"] = data

var paramsArray []string
treeSet := treeset.NewWithStringComparator()
for key := range params {
   treeSet.Add(key)
}
iter := treeSet.Iterator()
for iter.Next() {
   k := iter.Value().(string)
   v := params[k]
   if v != nil && v != "" {
      tpe := reflect.TypeOf(v)
      switch tpe.Kind() {
      case reflect.Array, reflect.Slice, reflect.Struct, reflect.Map:
         value, _ := json.Marshal(v)
         paramsArray = append(paramsArray, fmt.Sprintf("%s=%v", k, string(value)))
      default:
         paramsArray = append(paramsArray, fmt.Sprintf("%s=%v", k, v))
      }
   }
}

errRes := dispatch.Errors{}
sig := util.GenSignature(appSecret, strings.Join(paramsArray, "&"))
params["sig"] = sig

paramsArray = append(paramsArray, fmt.Sprintf("sig=%v", sig))

cli := resty.New()
res, err := cli.R().
   SetBody(params).
   SetError(&errRes).
   Post("http://127.0.0.1:3000/api/v1/swan/open/test?" + strings.Join(paramsArray, "&"))

if err != nil || res.StatusCode() != 200 {
   t.Error("request error: ", err, errRes, "body: ", string(res.Body()))
}
```

Java签名示例：

```
public static void main(String[] args) {
    DateUtil instance = new DateUtil();
    Map payload = new HashMap();
    payload.put("userId", "u12344567");
    instance.post("/api/v1/open/test", payload);
}

String post(String path, Map<String, String> params) {
    String apiKey = "V1eSG6lAg6PB4VhJ509AMgPR5OTw0JA";
    String apiSecret = "R0DWiCTJK7ZpHXKOqqZ3I5fyqFarDRE";
    HttpClient httpClient = HttpClients.createDefault();

    URIBuilder builder = new URIBuilder().setScheme("http")
```

```
            .setHost("127.0.0.1:3000")
            .setPath(path);
    // clear the params with empty value
    Map<String, String> trimmedParams = new HashMap<>();
    for (String key : params.keySet()) {
        if (params.get(key) != null) {
            trimmedParams.put(key, params.get(key));
        }
    }
    addRequiredParams(trimmedParams, apiKey, apiSecret);

    try {
        URI uri = builder.build();
        RequestBuilder requestBuilder = RequestBuilder.post(uri);
        List<NameValuePair> kvs = new ArrayList<>();
        for (String key : trimmedParams.keySet()) {
            kvs.add(new BasicNameValuePair(key, trimmedParams.get(key)));
        }

        requestBuilder.setEntity(new UrlEncodedFormEntity(kvs, "UTF-8"));
        HttpUriRequest request = requestBuilder.build();
        HttpResponse resp = httpClient.execute(request);
        if (resp.getStatusLine().getStatusCode() >= 300) {
            System.err.println("Something wrong: " + resp.getStatusLine().toString());
        }
        BufferedReader input = new BufferedReader(new InputStreamReader(resp.getEntity().getContent(), "UTF-
8"));
        StringBuilder sb = new StringBuilder();
        char[] buf = new char[1000];
        int count;
        while ((count = input.read(buf)) > 0) {
            sb.append(buf, 0, count);
        }
        return sb.toString();
    } catch (IOException | URISyntaxException e) {
        throw new RuntimeException(e);
    }
}


void addRequiredParams(Map<String, String> params, String apiKey, String apiSecret) {
    params.put("key", apiKey);
    params.put("sigVer", "1");
    String ts = DateTimeFormatter.ofPattern("yyyy-MM-dd'T'HH:mm:ss.SSS").format(LocalDateTime.now());
    params.put("ts", ts);
    params.put("nonce", RandomStringUtils.randomAlphanumeric(16));
    String sig = getSig(apiSecret, params);
    params.put("sig", sig);
}

String getSig(String apiSecret, Map<String, String> params) {
    String HMAC_SHA1_ALGORITHM = "HmacSHA1";

    StringBuilder sb = new StringBuilder();
    Set<String> keySet = new TreeSet<>(params.keySet());
    for (String key: keySet) {
        sb.append(key);
        sb.append("=");
        sb.append(params.get(key));
        sb.append("&");
    }
    sb.setLength(sb.length() - 1); // trim the last "&"
    String unifiedString = sb.toString();

    // calc hmac sha1
    try {
        SecretKeySpec secret = new SecretKeySpec(apiSecret.getBytes(), "HmacSHA1");
        Mac mac = Mac.getInstance(HMAC_SHA1_ALGORITHM);
        mac.init(secret);
        byte[] hmac = mac.doFinal(unifiedString.getBytes()); // UTF8
```

```
        // base64 encode the hmac
        String sig = Base64.getEncoder().encodeToString(hmac);
        return sig;
    } catch (NoSuchAlgorithmException e) {
        e.printStackTrace();
    } catch (InvalidKeyException e) {
        e.printStackTrace();
    }

    return null;
}
```

Python签名示例：

```
 def openApi(self,appKey,appSecret,bodyOrQurry):
        if isinstance(self.bodyOrQurry,str):
            self.bodyOrQurry=self.querryTransformBody(self.bodyOrQurry)
        ts=str((datetime.datetime.now().isoformat()))[:-3]
        nonce = ''.join(random.sample(string.ascii_letters + string.digits, 16))
        params = dict({"ts": ts, "key": self.appKey, "nonce": nonce, "sigVer": "1"},**self.bodyOrQurry)
        paramsArray = sorted(params.items())
        qurry=""
        for key ,value in paramsArray:
            if key!="sig" and value:
                qurry="%s%s=%s&"%(qurry,key,value)
        qurry=qurry.strip('&')
        sig=hmac.new(bytes(self.appSecret,encoding='utf-8'),bytes(qurry,encoding='utf-8'),hashlib.sha1).digest()
        sig = base64.b64encode(sig).decode()
        return qurry+"&sig=%s"%(sig)
if __name__ == "__main__":
    appKey="V1eSG6lAg6PB4VhJ509AMgPR5OTw0JA"
    appSecret="R0DWiCTJK7ZpHXKOqqZ3I5fyqFarDRE"
        url="/api/v1/open/swan/game-engine/event/signal"
    bodyOrQurry={
                    "userId": "renwu621000118",
                    "signal": "event2",
                }
        qurryAndSig = openApi(appKey,appSecret,bodyOrQurry)
        requests.post("https://swan.finogeeks.club" + "%s?%s"%(url,qurryAndSig), verify=False, data=json.dumps
(bodyOrQurry),headers=headers)
```