链滴

如何设计一个对外的回调接口

作者：mubai
原文链接：https://ld246.com/article/1592649123843
来源网站：链滴
许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

引言

最近的工作中涉及到第三方对接，用户在第三方的平台进行某些操作后，第三方会调用我们提供的回接口通知，然后我们再进行一些业务上的处理，比如奖励发放等。通过这篇文章总结自己设计回调接的一些思路。

接口设计

1、请求方式

为了方便扩展以及参数上的考虑，请求方式为 POST，Content-Type 为 application/json 。

2、参数定义

下面列举一些必要的请求参数：

参数	类型	说明
requestId	String	请求ID，第三自定义，每次请求唯一，用于幂等性处理
userId	String	用户id
appKey	String	媒体公钥
timestamp	Long	用于签名的时戳，毫秒
sign	String	签名，32位小写，M5(requestId + userId + appKey + appSecret + timestamp)
ext	JSONObject	额外参数，方后续扩展

注：用于签名的 appKey 和 appSecret 通过其他方式给到第三方，appKey 和 appSecret ——对应。

3、接口设计

参数校验

对参数进行非空判断和其他校验，过滤无效的请求。

时效限制

将 timestamp 与当前时间进行判断，超过 5 分钟则不处理该请求。该限制主要出于安全上的考虑。

签名校验

对部分参数使用哈希算法计算出签名，其中 appSecret 为公开的媒体私钥，和 appKey ——对应。过验签过滤无效或者非法的请求。

可以使用 MD5 以外的其他哈希算法。

签名参数可以先排序再哈希。

幂等性

为了避免第三方重复调用，或者用户以非正常方式重复触发请求，需要对请求进行幂等性处理，确保次请求的结果一致。

使用 Redis 来保存每次请求的 requestId。每次请求判断 Redis 中是否存在对应的 key，如果存在则绝当前请求，否则保存本次请求的 requestId。

根据具体业务对多次请求的情况进行处理，避免重复请求导致异常，比如奖励超发等。

错误码

针对不同的异常情况设计合理的错误码和错误信息，并通过接口返回。能够提高第三方对接效率，并能够针对部分异常情况进行特殊处理，比如重发请求等。

示例代码

PHP

```php
<?php
$request_id = $_POST['requestId'];
$user_id = $_POST['userId'];
$sign = $_POST['sign'];
$app_key = $_POST['appKey'];
$timestamp = $_POST['timestamp'];

// 参数校验，略

// 时效性验证
if (time() * 1000 - $timestamp > 300000) {
    $arr = array('code' => '-1', 'msg' => '时效性验证失败');
    echo json_encode($arr);
    exit();
}

// 从数据库查询 appKey 对应的 appSecret
$app_secret = '';

// 签名验证
$str = $request_id . $user_id . $app_key . $app_secret . $timestamp;
if (empty($sign) || md5($str) != $sign) {
    $arr = array('code' => '-2', 'msg' => '验签失败');
    echo json_encode($arr);
    exit();
}

// TODO 业务实现及异常处理

$arr = array('code' => '0', 'msg' => '成功');
echo json_encode($arr);
?>
```