华为技术有限公司内部技术标准

PUSH使用指导

V1.0

2016年08月18日发布 2016年08月28日实施

华为技术有限公司

Huawei Technologies Co., Ltd.

版权所有 侵权必究

All rights reserved

PUSH使用指导V1.0

|  |  |  |  |
| --- | --- | --- | --- |
| 拟制 | 陆松超l00169780 | 日期 | 2016.8.6 |
| 审核 |  | 日期 |  |
| 批准 |  | 日期 |  |



华为技术有限公司

版权所有 侵权必究

（仅供内部使用）

修订记录

| 日期 | 修订版本 | CR号 | 修改章节 | 修改描述 | 作者 |
| --- | --- | --- | --- | --- | --- |
| 2016-8-10 | 1.0 |  | 初稿 |  | 陆松超 |
|  |  |  |  |  |  |

目 录

1 PUSH概述 6

1.1 华为PUSH使用概览 6

1.2 华为PUSH的QOS 6

1.3 华为PUSH通道的安全性 7

1.3.1 APP Provider和Huawei Push之间的HTTPS通道说明 7

1.3.2 Huawei Push直接的安全加密通道说明 7

1.3.3 华为Push SDK和其他APP直接的消息通道说明 8

1.4 PUSH功能概览 8

1.5 PUSH术语和名词解释 8

1.6 PUSH中国区现网性能 8

2 PUSH开发指南 9

2.1 PUSH开放API简介 9

2.2 PUSH使用流程 9

2.3 PUSH使用场景详解 10

2.3.1 Android平台PUSH消息 10

2.4 开发指导与示例 10

2.4.1 Push开放HTTPS API接口 10

2.4.2 Push Java SDK开放API开发指导 19

3 限制与约束 32

3.1 PUSH消息送达率无法保证 32

3.2 PUSH消息不同终端平台的限制 32

3.3 PUSH消息的及时性 32

3.4 PUSH消息在终端的显示和交互方式限制 32

3.5 PUSH消息大小限制 32

3.6 PUSH平台离线缓存消息限制 33

3.7 PUSH消息流控限制 33

3.8 PUSH功耗流量限制 33

3.9 安全性约束 33

4 FAQ 34

PUSH使用指导

关键词：PUSH Android

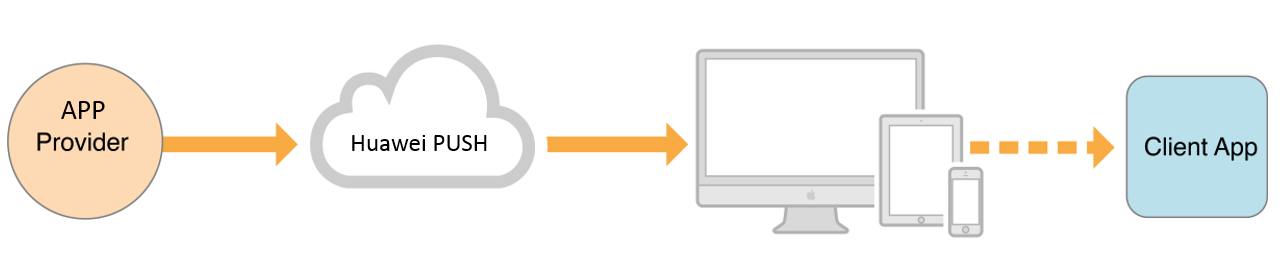
摘 要：本文描述华为PUSH平台的主要功能、开放API、适用场景和PUSH平台约束

缩略语清单：

| 缩略语 | 英文全名 | 中文解释 |
| --- | --- | --- |
| QOS | Quality of Service | 服务质量 |
|  |  |  |

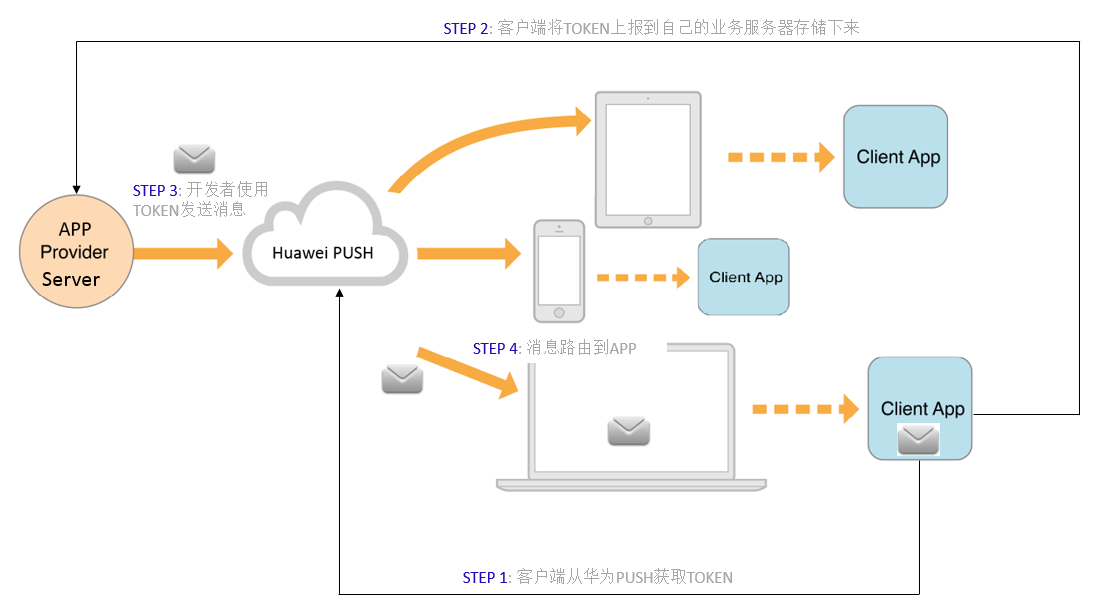
# PUSH概述

## 华为PUSH使用概览



华为PUSH建立了一条从应用服务器到终端APP之间的通知消息传输通道，APP的开发者通过华为PUSH平台可以方便的向华为设备上安装的APP推送通知消息。

APP提供者通过四个步骤使用华为PUSH，具体如下图所示：



STEP1: 开发者的APP集成华为PUSH客户端SDK，调用客户的获取华为PUSH TOKEN[[1]](#footnote-1)接口获取到TOKEN；

STEP 2: 开发者的APP对于获取到的华为PUSH TOKEN需要保存到开发者自己的服务器上；

STEP 3: 开发者利用服务器上保存的TOKEN向设备上APP发送PUSH消息；

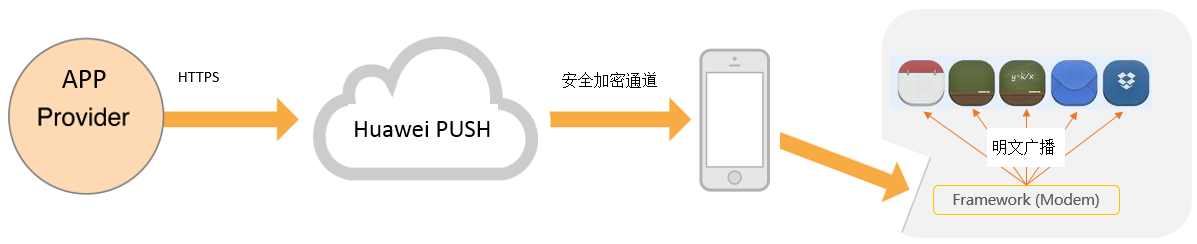
STEP 4: 消息携带的TOKEN作为鉴权和路由机制会通过华为PUSH平台将消息发送到最终用户设备上；

## 华为PUSH的QOS

华为PUSH平台采用尽量投递的原则，当消息到达华为PUSH后，华为PUSH平台会尽力讲消息投递下去，如果设备不在线华为PUSH平台会将消息缓存起来，等到设备上线后华为PUSH平台会将缓存的消息再次推送给用户。如果用户很长时间不在线，这些缓存的消息不会无限制保留在华为PUSH平台上，默认为保存48小时，超时的消息会被丢弃。

## 华为PUSH通道的安全性

PUSH通道的安全性分为三个层面，首先PUSH平台使用者通过HTTPS通道将消息发送到华为PUSH平台，其次华为PUSH平台和终端设备之间使用的是安全的加密通道，最后消息到达Android系统后如果是透传消息那么消息是通过广播明文的方式到达最终APP的。如果是通知消息由Push NC显示，点击后拉起App，拉起方式不是广播。



### APP Provider和Huawei Push之间的HTTPS通道说明

获取Access Token的URL为：<https://login.vmall.com/oauth2/token>

消息发送的URL为：<https://api.vmall.com/rest.php>

华为Push平台对外的证书是有赛门铁克提供的:



用户使用时为了确保连接的是华为Push需要使用赛门铁克的公钥对华为的证书进行验签，并校验证书中的地址是\*.vmall.com。

### Huawei Push直接的安全加密通道说明

采用的加密算法是RSA 1024，如果需要更强的安全性请用户自行加解密信息。

### 华为Push SDK和其他APP直接的消息通道说明

透传类消息是通过Android广播的方式直接明文到达APP.

## PUSH功能概览

PUSH服务作为华为终端云平台的基础能力，主要功能有：

* 支持单发、群发机制
* 支持通知栏消息和透传消息两种方式
* 有限支持第三方Android平台PUSH消息发送
* 支持缓存用户消息的能力(当终端用户不在线时)，默认保存48小时，最大保存30天
* 支持Android平台多用户条件下向不同用户发送PUSH消息的机制

## PUSH术语和名词解释

|  |  |  |
| --- | --- | --- |
| 术语名称 | 含义 | 备注 |
| 通知栏消息 | 所谓通知栏消息是指消息通过PUSH平台发送到PUSH客户端的时候使用华为默认的消息呈现和点击动作（点击后是需要应用处理的）。 | 华为默认行为 |
| 透传消息 | 所谓透传消息是指消息通过PUSH平台到达PUSH客户端的时候会透传给对应的APP由APP自己控制消息呈现方式和点击动作。 | APP自定义行为 |
| PUSH TOKEN | PUSH TOKEN是华为PUSH平台分配个一个特定设备上APP的路由和鉴权信息，开发者使用华为PUSH TOKEN可以将消息通过华为PUSH平台路由到特定设备的特定APP。 |  |

## PUSH中国区现网性能

|  |  |
| --- | --- |
| **性能指标** | **描述** |
| 单个CP发送速率 | 当前从网关开放出来的API没有流控限制，但是如果总体流量过大发送时会出现Service Unavailable的信息。 |
| 设备在线时及时消息的时延 | 如果设备网络正常，并且下发速率控制在PUSH服务器承受的范围内，一般时延在1秒内。 |

# PUSH开发指南

## PUSH开放API简介

|  |  |  |  |
| --- | --- | --- | --- |
| **开放途径** | **开放API** | **API作用描述** | **备注** |
| 华为网关  <https://api.vmall.com>/rest.php | openpush.message.single\_send | 单发及时透传消息; | 推荐使用 |
| openpush.message.batch\_send | 群发及时透传消息 | 推荐使用 |
| openpush.message.psSingleSend | 单发及时通知栏消息 | 推荐使用 |
| openpush.message.psBatchSend | 群发及时通知栏消息 | 推荐使用 |
| openpush.openapi.notification\_send | Android:  支持发送给特定用户、发送给所有用户；  支持单发和群发；  支持定时发送；  支持Android多用户； | 1.及时消息不建议使用本接口，后续会下线  2. 本接口适合发送标签类消息  3. 本接口适合发送需要定时发送功能的业务场景 |
| openpush.openapi.query\_msg\_result | 查询一次发送请求中发送给特定用户的消息状态 |  |

## PUSH使用流程

开通PUSH权益

获取PUSH SDK

开发应用

STEP 1: 开通PUSH权益

|  |
| --- |
| 在使用华为PUSH之前首先需要开通APP的PUSH权益，具体流程请参考：  <http://developer.huawei.com/cn/consumer/wiki/index.php?title=%E6%8E%A5%E5%85%A5%E8%AF%B4%E6%98%8E> |

STEP 2: 获取PUSH SDK

|  |
| --- |
| PUSH SDK包括两个部分：   * Android Client SDK; * 服务器开放API SDK;   可以通过下面的地址下载  http://developer.huawei.com/cn/consumer/wiki/index.php?title=PushSDK%E4%B8%8B%E8%BD%BD |

STEP３：开发应用

|  |
| --- |
| 应该开发过程中，用户通过APP调用Android Client SDK获取华为PUSH的Push Token后需要在对应的服务器侧妥善保管，后续向用户发送及时PUSH消息时直接通过本Push Token方式进行； |

## PUSH使用场景详解

### Android平台PUSH消息

|  |  |  |
| --- | --- | --- |
| 业务场景 | 适用接口 | 备注 |
| Android平台及时消息 | openpush.message.single\_send  openpush.message.batch\_send  openpush.message.psSingleSend  openpush.message.psBatchSend | 批量消息最大的PUSH TOKEN数量为1000个。 |
| Android平台向所有APP用户发送消息 | openpush.openapi.notification\_send |  |
| 查询PUSH请求状态 | openpush.openapi.query\_msg\_result |  |

## 开发指导与示例

关于服务器侧调用PUSH和客户端使用PUSH的示例代码请参考SDK (开发语言为Java)

<http://developer.huawei.com/cn/consumer/wiki/index.php?title=PushSDK%E4%B8%8B%E8%BD%BD>

### Push开放HTTPS API接口

Push开放API使用的是https协议,具体使用参数请参考:

|  |
| --- |
| <http://developer.huawei.com/cn/consumer/wiki/index.php?title=%E8%B0%83%E7%94%A8%E8%A7%84%E8%8C%83> |

#### 获取下发的Access Token

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口访问地址：  <https://login.vmall.com/oauth2/token>  接口请求报文样例：   |  | | --- | | POST /oauth2/token HTTP/1.1  Host: login.vmall.com Content-Type: application/x-www-form-urlencoded  grant\_type=client\_credentials&client\_secret=xxxxdtsb4abxxxlz2uyztxxxfaxxxxxx&client\_id=appid100 |   接口请求字段说明：   |  | | --- | | **grant\_type**：必选 ，此时必须为”client\_credentials”  **client\_id**：必选，申请应用时获得的应用 ID。  **client\_secret**：必选，申请应用时获得的应用密钥。  **scope**：可选，以空格分隔的权限列表，若不传递此参数，默认权限包括（nsp.auth nsp.user nsp.vfs nsp.ping openpush.message） |   接口应答报文样例：   |  | | --- | | 成功：  HTTP/1.1 200 OK Content-Type: application/json Cache-Control: no-store { 　　"access\_token": "BFEsZqBuaW\/\/9GGn1mLt+KykzfBObIy6V3eN20ufoNiAERl33oVGzlN5TE5qeA==", 　　"expires\_in": 86400, }  错误：  HTTP/1.1 400 Bad Request Content-Type: application/json Cache-Control: no-store { 　　"error": "invalid\_code", 　　"error\_description": "The code is invalid" } |   接口应答报文说明：   |  | | --- | | **access\_token：**要获取的Access Token；  **expires\_in：**Access Token的有效期，以秒为单位；  **scope：**Access Token的访问范围，即用户实际授予的权限列表（用户在授权页面时，有可能会取消掉某些请求的权限）；  **error：**错误码。  **error\_description：**错误描述信息，用来帮助理解和解决发生的错误。 | |

#### 单发透传及时消息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口访问地址：  <https://api.vmall.com/rest.php>  接口请求报文样例：   |  | | --- | | POST /rest.php HTTP/1.1  Host: api.vmall.com  Content-Type: application/x-www-form-urlencoded  access\_token= mF\_9.B5f-4.1JqM &msgType=1&requestID=1\_1362472787848&cacheMode=1&nsp\_fmt=JSON&expire\_time=2016-08-24+16%3A42%3A31&message=hello%7E%7E+you+got+a+push+message&priority=0&nsp\_svc=openpush.message.single\_send&nsp\_ts=1472017414&deviceToken=00000000000000000000000000000000 |   接口请求字段说明：   |  | | --- | | access\_token: 必选 使用OAuth2进行鉴权时的AccessToken  **nsp\_fmt**：必选 ，取值建议固定为“JSON”，另外两种取值为：“php-rpc”和“JS”。  nsp\_ts: 必选，服务请求时间戳，自GMT 时间 1970-1-1 0:0:0至今的秒数。如果传入的时间与服务器时间相差5分钟以上，服务器可能会拒绝请求。  nsp\_svc: 必选， 本接口固定为openpush.message.single\_send  关于业务字段msgType, requestID,catheMode,expire\_time,message,priority,deviceToken的取值解释详见“2.4.2.2 单发透传及时信息”中参数说明章节。 |   接口应答报文样例：   |  | | --- | | 成功：  HTTP/1.1 200 OK  Pragma: no-cache  Cache-Control: no-cache  Expires: Thu, 01 Jan 1970 00:00:00 GMT  Content-Type: application/json;  charset=utf-8  {  "resultcode ":"0",  "message":"success",  "requestID":"201608231040425822EF0CD"  }  错误：  HTTP/1.1 200 OK  Date: Tue, 15 Jan 2013 08:12:38 GMT  Pragma: no-cache  Cache-Control: no-cache  Expires: Thu, 01 Jan 1970 00:00:00 GMT  Content-Type: text/plain; charset=utf-8  NSP\_STATUS: 109  {"error":"nsp\_ts error"} |   接口应答报文说明：   |  | | --- | | resultcode**：**错误码，详细描述请参考“2.4.2.2 单发透传及时消息”章节的错误码描述部分；  message: resultcode对应的错误描述  requestID**：**每次下发请求过程中华为PUSH平台分配的请求ID; | |

#### 单发通知栏及时消息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口访问地址：  <https://api.vmall.com/rest.php>  接口请求报文样例：   |  | | --- | | POST /rest.php HTTP/1.1  Host: api.vmall.com  Content-Type: application/x-www-form-urlencoded  access\_token=accessToken&msgType=1&android=hello%7E%7E+you+got+a+push+message&cacheMode=1&nsp\_fmt=JSON&expire\_time=2016-08-24+17%3A17%3A44&userType=-1&nsp\_svc=openpush.message.psSingleSend&nsp\_ts=1472019480&deviceToken=00000000000000000000000000000000 |   接口请求字段说明：   |  | | --- | | access\_token: 必选 使用OAuth2进行鉴权时的AccessToken  **nsp\_fmt**：必选 ，取值建议固定为“JSON”，另外两种取值为：“php-rpc”和“JS”。  nsp\_ts: 必选，服务请求时间戳，自GMT 时间 1970-1-1 0:0:0至今的秒数。如果传入的时间与服务器时间相差5分钟以上，服务器可能会拒绝请求。  nsp\_svc: 必选， 本接口固定为openpush.message.psSingleSend  关于业务字段msgType, android,catheMode,expire\_time,userType,deviceToken的取值解释详见“2.4.2.3 单发通知栏及时信息”中参数说明章节。 |   接口应答报文样例：   |  | | --- | | 成功：  HTTP/1.1 200 OK  Pragma: no-cache  Cache-Control: no-cache  Expires: Thu, 01 Jan 1970 00:00:00 GMT  Content-Type: text/plain;  charset=utf-8  {  " resultcode ":"0",  " message":"success",  " requestID":"201608231040425822EF0CD"  }  错误：  HTTP/1.1 200 OK  Date: Tue, 15 Jan 2013 08:12:38 GMT  Pragma: no-cache  Cache-Control: no-cache  Expires: Thu, 01 Jan 1970 00:00:00 GMT  Content-Type: text/plain; charset=utf-8  NSP\_STATUS: 109  {"error":"nsp\_ts error"} |   接口应答报文说明：   |  | | --- | | resultcode**：**错误码，详细描述请参考“2.4.2.3 单发通知栏及时消息”章节的错误码描述部分；  message: resultcode对应的错误描述  requestID**：**每次下发请求过程中华为PUSH平台分配的请求ID; | |

#### 群发透传及时消息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口访问地址：  <https://api.vmall.com/rest.php>  接口请求报文样例：   |  | | --- | | POST /rest.php HTTP/1.1  Host: api.vmall.com  Content-Type: application/x-www-form-urlencoded  access\_token=accessToken&msgType=1&cacheMode=1&nsp\_fmt=JSON&expire\_time=2013-09-30+19%3A55&message=hello%7E%7E+you+got+a+push+message&nsp\_svc=openpush.message.batch\_send&deviceTokenList=%5B%2200000000000000000000000000000000%22%2C%2200000000000000000000000000000000%22%5D&nsp\_ts=1472020032 |   接口请求字段说明：   |  | | --- | | access\_token: 必选 使用OAuth2进行鉴权时的AccessToken  **nsp\_fmt**：必选 ，取值建议固定为“JSON”，另外两种取值为：“php-rpc”和“JS”。  nsp\_ts: 必选，服务请求时间戳，自GMT 时间 1970-1-1 0:0:0至今的秒数。如果传入的时间与服务器时间相差5分钟以上，服务器可能会拒绝请求。  nsp\_svc: 必选， 本接口固定为openpush.message.batch\_send  关于业务字段msgType, message,catheMode,expire\_time,deviceTokenList的取值解释详见“2.4.2.4 群发透传及时信息”中参数说明章节。 |   接口应答报文样例：   |  | | --- | | 成功：  HTTP/1.1 200 OK  Pragma: no-cache  Cache-Control: no-cache  Expires: Thu, 01 Jan 1970 00:00:00 GMT  Content-Type: text/plain;  charset=utf-8  {  " resultcode ":"0",  " message":"success",  " requestID":"201608231040425822EF0CD"  }  错误：  HTTP/1.1 200 OK  Date: Tue, 15 Jan 2013 08:12:38 GMT  Pragma: no-cache  Cache-Control: no-cache  Expires: Thu, 01 Jan 1970 00:00:00 GMT  Content-Type: text/plain; charset=utf-8  NSP\_STATUS: 109  {"error":"nsp\_ts error"} |   接口应答报文说明：   |  | | --- | | resultcode**：**错误码，详细描述请参考“2.4.2.4 群发透传及时消息”章节的错误码描述部分；  message: resultcode对应的错误描述  requestID**：**每次下发请求过程中华为PUSH平台分配的请求ID; | |

#### 群发通知栏及时消息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口访问地址：  <https://api.vmall.com/rest.php>  接口请求报文样例：   |  | | --- | | POST /rest.php HTTP/1.1  Host: api.vmall.com  Content-Type: application/x-www-form-urlencoded  access\_token=accessToken&msgType=1&android=hello%7E%7E+you+got+a+push+message&cacheMode=1&nsp\_fmt=JSON&expire\_time=2013-09-30+19%3A55&userType=-1&nsp\_svc=openpush.message.psBatchSend&deviceTokenList=%5B%2200000000000000000000000000000000%22%2C%2200000000000000000000000000000000%22%5D&nsp\_ts=1472020575 |   接口请求字段说明：   |  | | --- | | access\_token: 必选 使用OAuth2进行鉴权时的AccessToken  **nsp\_fmt**：必选 ，取值建议固定为“JSON”，另外两种取值为：“php-rpc”和“JS”。  nsp\_ts: 必选，服务请求时间戳，自GMT 时间 1970-1-1 0:0:0至今的秒数。如果传入的时间与服务器时间相差5分钟以上，服务器可能会拒绝请求。  nsp\_svc: 必选， 本接口固定为openpush.message.psBatchSend  关于业务字段msgType, android,catheMode,expire\_time,userType,deviceTokenList的取值解释详见“2.4.2.5 群发通知栏及时信息”中参数说明章节。 |   接口应答报文样例：   |  | | --- | | 成功：  HTTP/1.1 200 OK  Pragma: no-cache  Cache-Control: no-cache  Expires: Thu, 01 Jan 1970 00:00:00 GMT  Content-Type: text/plain;  charset=utf-8  {  " resultcode ":"0",  " message":"success",  " requestID":"201608231040425822EF0CD"  }  错误：  HTTP/1.1 200 OK  Date: Tue, 15 Jan 2013 08:12:38 GMT  Pragma: no-cache  Cache-Control: no-cache  Expires: Thu, 01 Jan 1970 00:00:00 GMT  Content-Type: text/plain; charset=utf-8  NSP\_STATUS: 109  {"error":"nsp\_ts error"} |   接口应答报文说明：   |  | | --- | | resultcode**：**错误码，详细描述请参考“2.4.2.5 群发通知栏及时消息”章节的错误码描述部分；  message: resultcode对应的错误描述  requestID**：**每次下发请求过程中华为PUSH平台分配的请求ID; | |

#### 发送特定PUSH消息

本接口后续逐步下线这里不再阐述，如果使用Java SDK可以参考“2.4.2.6 发送特定PUSH消息”章节中的描述。

#### 查询PUSH消息状态

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 接口访问地址：  <https://api.vmall.com/rest.php>  接口请求报文样例：   |  | | --- | | POST /rest.php HTTP/1.1  Host: api.vmall.com  Content-Type: application/x-www-form-urlencoded  access\_token=accessToken&nsp\_fmt=JSON&nsp\_svc=openpush.openapi.query\_msg\_result&request\_id=201608231040425822EF0CD&nsp\_ts=1472021073&token=000000000000000000 |   接口请求字段说明：   |  | | --- | | access\_token: 必选 使用OAuth2进行鉴权时的AccessToken  **nsp\_fmt**：必选 ，取值建议固定为“JSON”，另外两种取值为：“php-rpc”和“JS”。  nsp\_ts: 必选，服务请求时间戳，自GMT 时间 1970-1-1 0:0:0至今的秒数。如果传入的时间与服务器时间相差5分钟以上，服务器可能会拒绝请求。  nsp\_svc: 必选， 本接口固定为openpush.openapi.query\_msg\_result  关于业务字段request\_id和token的取值解释详见“2.4.2.7 查询PUSH消息状态”中参数说明章节。 |   接口应答报文样例：   |  | | --- | | 成功：  HTTP/1.1 200 OK  Pragma: no-cache  Cache-Control: no-cache  Expires: Thu, 01 Jan 1970 00:00:00 GMT  Content-Type: text/plain;  charset=utf-8  {  "request\_id ":"201608231040425822EF0CD ",  "result":[{"status":0,"token":"00000000000000000000000000000000"},  {"status":1,"token":"00000000000000000000000000000001"}]  }  错误：  HTTP/1.1 200 OK  Date: Tue, 15 Jan 2013 08:12:38 GMT  Pragma: no-cache  Cache-Control: no-cache  Expires: Thu, 01 Jan 1970 00:00:00 GMT  Content-Type: text/plain; charset=utf-8  NSP\_STATUS: 109  {"error":"nsp\_ts error"} |   接口应答报文说明：   |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | requestID**：**每次下发请求过程中华为PUSH平台分配的请求ID;  result: JSON ARRAY结构体   |  |  |  |  | | --- | --- | --- | --- | | 参数名称 | 类型 | 是否必须 | 描述 | | token | String | M | 用户标识 | | status | Int | M | 消息状态:  0: 成功送达  1: 待发送  2: 被覆盖  3: 过期丢弃  11: 设备不在线缓存  41: 不在线不缓存  12: 无路由缓存  42: 无路由不缓存  13: 前转  43: APP已卸载 | | |

### Push Java SDK开放API开发指导

获取接入AccessToken

调用API发送PUSH消息

#### 2.4.2.1 获取下发的Access Token

使用SDK获取AccessToken的示例代码如下：

|  |
| --- |
| /\*  \* 获取token的方法 appId为开发者联盟上面创建应用的APP ID appKey为开发者联盟上面创建应用的 APP SECRET  \* APP ID：appid100 应用包名：com.open.test | APP SECRET：xxxxdtsb4abxxxlz2uyztxxxfaxxxxxx  \*/  OAuth2Client oauth2Client = **new** OAuth2Client();  oauth2Client.initKeyStoreStream(Demo.**class**.getResource("/mykeystorebj.jks").openStream(), "123456");  String appId = "appid100";  String appKey = "xxxxdtsb4abxxxlz2uyztxxxfaxxxxxx";    AccessToken access\_token = oauth2Client.getAccessToken("client\_credentials", appId, appKey);  NSPClient client = **new** NSPClient(access\_token.getAccess\_token());  client.initHttpConnections(30, 50);//设置每个路由的连接数和最大连接数  client.initKeyStoreStream(Demo.**class**.getResource("/mykeystorebj.jks").openStream(), "123456");  //如果访问https必须导入证书流和密码 |

#### 单发透传及时消息

API介绍：

|  |
| --- |
| 本接口是发送给指定用户终端设备上特定APP使用的，如果设备在线发送的消息及时下发到终端用户，否则最多在PUSH系统存储48小时，如果48小时内用户上线华为PUSH平台会将该消息再次推送给用户，否则消息将无法到达最终用户。  消息到达Android设备后 |

调用从华为网关开放的API：

|  |
| --- |
| openpush.message.single\_send |

示例代码：

|  |
| --- |
| **public** **static** **void** single\_send(NSPClient client)  **throws** NSPException  {  **long** currentTime = System.*currentTimeMillis*();  SimpleDateFormat dataFormat = **new** SimpleDateFormat(***TIMESTAMP\_NORMAL***);    //目标用户，必选。  //由客户端获取， 32 字节长度。手机上安装了push应用后，会到push服务器申请token，申请到的token会上报给应用服务器  String token = "00000000000000000000000000000000";    //发送到设备上的消息，必选  //最长为4096 字节（开发者自定义，自解析）  String message = "hello~~ you got a push message";    //必选  //0：高优先级  //1：普通优先级  //缺省值为1  **int** priority = 0;    //消息是否需要缓存，必选  //0：不缓存  //1：缓存  // 缺省值为0  **int** cacheMode = 1;    //标识消息类型（缓存机制），必选  //由调用端赋值，取值范围（1~100）。当TMID+msgType的值一样时，仅缓存最新的一条消息  **int** msgType = 1;    //可选  //如果请求消息中，没有带，则MC根据ProviderID+timestamp生成，各个字段之间用下划线连接  String requestID = "1\_1362472787848";    //unix时间戳，可选  //格式：2013-08-29 19:55  // 消息过期删除时间  //如果不填写，默认超时时间为当前时间后48小时  String expire\_time = dataFormat.format(currentTime + 3 \* 60 \* 60 \* 1000);    //构造请求  HashMap<String, Object> hashMap = **new** HashMap<String, Object>();  hashMap.put("deviceToken", token);  hashMap.put("message", message);  hashMap.put("priority", priority);  hashMap.put("cacheMode", cacheMode);  hashMap.put("msgType", msgType);  hashMap.put("requestID", requestID);  hashMap.put("expire\_time", expire\_time);    //设置http超时时间  client.setTimeout(10000, 15000);  //接口调用  PushRet resp = client.call("openpush.message.single\_send", hashMap, PushRet.**class**);    //打印响应  System.***err***.println("单发接口消息响应:" + resp.getResultcode() + ",message:" + resp.getMessage());  } |

接口参数详解：

|  |  |  |
| --- | --- | --- |
| **参数名称** | **作用** | **备注** |
| deviceToken | 华为PUSH TOKEN |  |
| message | 发送的PUSH消息体 |  |
| priority | 0：高优先级  1：普通优先级  缺省值为1 |  |
| cacheMode | 0：不缓存  1：缓存  缺省值为0 |  |
| msgType | push消息分为缓存消息和不缓存消息，若为缓存消息，push会缓存此消息，首次下发不成功的情况下有一次重发的机会，重发后不再次重发，删除缓存消息；不缓存消息不能重发。 缓存消息不能无限时间保留，超过过期时间后会被删除，不再重发。  缓存消息在未超时的时间内可能被新的下发不成功的消息覆盖，从而不能重发。push的缓存消息中，视deviceToken和msgType相同的缓存消息为同一类缓存消息消息，新缓存消息可能覆盖旧的。  若想让每条缓存消息都不被覆盖，可将msgType取值为-1，这样只有过期后才会删除。 若想相同的deviceToken和msgType的缓存消息只保留最新的一条，可以将msgType取值为1-100之间。 | 1-100的值只是标记作用，本身没啥含义 |
| requestID | 不用填写，后续该字段会逐步不用 |  |
| expire\_time | 消息过期时间，可选  timestamp格式ISO 8601：2013-06-03T17:30:08+08:00 |  |
| userType | android多用户场景下，用户标识。0：当前用户，1：主用户。默认值为-1，表示调用方发消息时，未填写该字段 |  |

openpush.message.single\_send的错误码详解：

|  |  |  |
| --- | --- | --- |
| **错误码** | **错误码描述** | **备注** |
| 101 | Invalid api\_key. | 不存在的api key，一般指不存在的appId |
| 20005 | Illegal cache Mode. | 不合法的存储模式 |
| 20006 | Illegal msg Type. | 不合法的消息类型 |
| 20203 | No permission to send message to these tmIDs. | 无权发送消息给此token |
| 20204 | Invalid expire time. | 无效的超时时间 |
| 20205 | Not enough quota. | 没有足够的限额 |
| 80200004 | Illegal token. | 不合法的token |
| 80200007 | Illegal priority. | 不合法的优先级 |
| 80200015 | Illegal message. | 不合法的消息 |
| 80200018 | Invalid user type. | 不存在的用户类型 |
| 1 | error occured. | 内部错误 |

#### 单发通知栏及时消息

API介绍：

|  |
| --- |
| 本接口是发送给指定用户终端设备上特定APP使用的，如果设备在线发送的消息及时下发到终端用户，否则最多在PUSH系统存储48小时，如果48小时内用户上线华为PUSH平台会将该消息再次推送给用户，否则消息将无法到达最终用户。  消息到达Android设备后 |

调用从华为网关开放的API：

|  |
| --- |
| openpush.message.psSingleSend |

示例代码：

|  |
| --- |
| **public** **static** **void** ps\_single\_send(NSPClient client)  **throws** NSPException  {  **long** currentTime = System.*currentTimeMillis*();  SimpleDateFormat dataFormat = **new** SimpleDateFormat(***TIMESTAMP\_NORMAL***);    //目标用户，必选。  //由客户端获取， 32 字节长度。手机上安装了push应用后，会到push服务器申请token，申请到的token会上报给应用服务器  String token = "00000000000000000000000000000000";    //发送到设备上的消息，必选  //最长为4096 字节（开发者自定义，自解析）  String message = "hello~~ you got a push message";    //消息是否需要缓存，必选  //0：不缓存  //1：缓存  // 缺省值为0  **int** cacheMode = 1;    //标识消息类型（缓存机制），必选  //由调用端赋值，取值范围（1~100）。当TMID+msgType的值一样时，仅缓存最新的一条消息  **int** msgType = 1;    //可选  // 0: 当前用户  // 1: 主要用户  // -1: 默认用户  //  String userType = "-1";    //unix时间戳，可选  //格式：2013-08-29 19:55  // 消息过期删除时间  //如果不填写，默认超时时间为当前时间后48小时  String expire\_time = dataFormat.format(currentTime + 3 \* 60 \* 60 \* 1000);    //构造请求  HashMap<String, Object> hashMap = **new** HashMap<String, Object>();  hashMap.put("deviceToken", token);  hashMap.put("android", message);  hashMap.put("cacheMode", cacheMode);  hashMap.put("msgType", msgType);  hashMap.put("userType", userType);  hashMap.put("expire\_time", expire\_time);    //设置http超时时间  client.setTimeout(10000, 15000);  //接口调用  PushRet resp = client.call("openpush.message.psSingleSend", hashMap, PushRet.**class**);    //打印响应  System.***err***.println("单发接口消息响应:" + resp.getResultcode() + ",message:" + resp.getMessage());  } |

参数详解：

|  |  |  |
| --- | --- | --- |
| **参数名称** | **作用** | **备注** |
| deviceToken | 华为PUSH TOKEN |  |
| android | 发送的PUSH消息体 |  |
| cacheMode | 0：不缓存  1：缓存  缺省值为0 |  |
| msgType | 同前面描述 |  |
| userType | android多用户场景下，用户标识。0：当前用户，1：主用户。默认值为-1，表示调用方发消息时，未填写该字段 |  |
| expireTime | 消息过期时间，可选  timestamp格式ISO 8601：2013-06-03T17:30:08+08:00 |  |

openpush.message. psSingleSend的错误码详解：

|  |  |  |
| --- | --- | --- |
| **错误码** | **错误码描述** | **备注** |
| 20005 | Illegal cache Mode. | 不合法的存储模式 |
| 20006 | Illegal msg Type. | 不合法的消息类型 |
| 20203 | No permission to send message to these tmIDs. | 无权发送消息给此token |
| 20204 | Invalid expire time. | 无效的超时时间 |
| 20205 | Not enough quota. | 没有足够的限额 |
| 80200004 | Illegal token. | 不合法的token |
| 80200015 | Illegal message. | 不合法的消息 |
| 80200018 | Invalid user type. | 不存在的用户类型 |
| 1 | error occured. | 内部错误 |

#### 群发透传及时消息

调用从华为网关开放的API：

|  |
| --- |
| openpush.message.batch\_send |

示例代码：

|  |
| --- |
| **public** **static** **void** batch\_send(NSPClient client)  **throws** NSPException  {  //目标用户列表，必选  //最多填1000个，每个目标用户为32字节长度，由系统分配的合法TMID。手机上安装了push应用后，会到push服务器申请token，申请到的token会上报给应用服务器  String[] deviceTokenList = {"00000000000000000000000000000000", "00000000000000000000000000000000"};    //发送到设备上的消息，必选  //最长为4096 字节（开发者自定义，自解析）  String message = "hello~~ you got a push message";    //消息是否需要缓存，必选  //0：不缓存  //1：缓存  // 缺省值为0  Integer cacheMode = 1;    //标识消息类型（缓存机制），必选  //由调用端赋值，取值范围（1~100）。当TMID+msgType的值一样时，仅缓存最新的一条消息  Integer msgType = 1;    //unix时间戳，可选  //格式：2013-08-29 19:55  // 消息过期删除时间  //如果不填写，默认超时时间为当前时间后48小时  String expire\_time = "2013-09-30 19:55";    //构造请求  HashMap<String, Object> hashMap = **new** HashMap<String, Object>();  hashMap.put("deviceTokenList", deviceTokenList);  hashMap.put("message", message);  hashMap.put("cacheMode", cacheMode);  hashMap.put("msgType", msgType);  hashMap.put("expire\_time", expire\_time);    //设置http超时时间  client.setTimeout(10000, 15000);  //接口调用  PushRet resp = client.call("openpush.message.batch\_send", hashMap, PushRet.**class**);    //打印响应  System.***err***.println("群发接口消息响应: resultcode：" + resp.getResultcode() + ",message:" + resp.getMessage());  } |

参数详解：

|  |  |  |
| --- | --- | --- |
| **参数名称** | **作用** | **备注** |
| deviceTokenList | 华为PUSH TOKEN |  |
| message | 发送的PUSH消息体 |  |
| cacheMode | 0：不缓存  1：缓存  缺省值为0 |  |
| msgType | 同前面描述 |  |
| expire\_time | 消息过期时间，可选  timestamp格式ISO 8601：2013-06-03T17:30:08+08:00 |  |
| userType | android多用户场景下，用户标识。0：当前用户，1：主用户。默认值为-1，表示调用方发消息时，未填写该字段 |  |

openpush.message. batch\_send的错误码详解：

|  |  |  |
| --- | --- | --- |
| **错误码** | **错误码描述** | **备注** |
| 20005 | Illegal cache Mode. | 不合法的存储模式 |
| 20006 | Illegal msg Type. | 不合法的消息类型 |
| 20203 | No permission to send message to these tmIDs. | 无权发送消息给此token |
| 20204 | Invalid expire time. | 无效的超时时间 |
| 20205 | Not enough quota. | 没有足够的限额 |
| 80200004 | Illegal token. | 不合法的token |
| 80200008 | Illegal DeviceTokenList length. | token队列长度不合法 |
| 80200015 | Illegal message. | 不合法的消息 |
| 80200018 | Invalid user type. | 不存在的用户类型 |
| 1 | error occured. | 内部错误 |

#### 群发通知栏及时消息

调用从华为网关开放的API：

|  |
| --- |
| openpush.message.psBatchSend |

示例代码：

|  |
| --- |
| **public** **static** **void** ps\_batch\_send(NSPClient client)  **throws** NSPException  {  //目标用户列表，必选  //最多填1000个，每个目标用户为32字节长度，由系统分配的合法TMID。手机上安装了push应用后，会到push服务器申请token，申请到的token会上报给应用服务器  String[] deviceTokenList = {"00000000000000000000000000000000", "00000000000000000000000000000000"};    //发送到设备上的消息，必选  //最长为4096 字节（开发者自定义，自解析）  String message = "hello~~ you got a push message";    //消息是否需要缓存，必选  //0：不缓存  //1：缓存  // 缺省值为0  Integer cacheMode = 1;    //标识消息类型（缓存机制），必选  //由调用端赋值，取值范围（1~100）。当TMID+msgType的值一样时，仅缓存最新的一条消息  Integer msgType = 1;    //可选  // 0: 当前用户  // 1: 主要用户  // -1: 默认用户  //  String userType = "-1";    //unix时间戳，可选  //格式：2013-08-29 19:55  // 消息过期删除时间  //如果不填写，默认超时时间为当前时间后48小时  String expire\_time = "2013-09-30 19:55";    //构造请求  HashMap<String, Object> hashMap = **new** HashMap<String, Object>();  hashMap.put("deviceTokenList", deviceTokenList);  hashMap.put("android", message);  hashMap.put("cacheMode", cacheMode);  hashMap.put("msgType", msgType);  hashMap.put("userType", userType);  hashMap.put("expire\_time", expire\_time);    //设置http超时时间  client.setTimeout(10000, 15000);  //接口调用  PushRet resp = client.call("openpush.message.psBatchSend", hashMap, PushRet.**class**);    //打印响应  System.***err***.println("群发接口消息响应: resultcode：" + resp.getResultcode() + ",message:" + resp.getMessage());  } |

参数详解：

|  |  |  |
| --- | --- | --- |
| **参数名称** | **作用** | **备注** |
| deviceTokenList | 华为PUSH TOKEN |  |
| android | 发送的PUSH消息体 |  |
| cacheMode | 0：不缓存  1：缓存  缺省值为0 |  |
| msgType | 同前面描述 |  |
| userType | android多用户场景下，用户标识。0：当前用户，1：主用户。默认值为-1，表示调用方发消息时，未填写该字段 |  |
| expireTime | 消息过期时间，可选  timestamp格式ISO 8601：2013-06-03T17:30:08+08:00 |  |

openpush.message.psBatchSend的错误码详解：

|  |  |  |
| --- | --- | --- |
| **错误码** | **错误码描述** | **备注** |
| 20005 | Illegal cache Mode. | 不合法的存储模式 |
| 20006 | Illegal msg Type. | 不合法的消息类型 |
| 20203 | No permission to send message to these tmIDs. | 无权发送消息给此token |
| 20204 | Invalid expire time. | 无效的超时时间 |
| 20205 | Not enough quota. | 没有足够的限额 |
| 80200004 | Illegal token. | 不合法的token |
| 80200008 | Illegal DeviceTokenList length. | token队列长度不合法 |
| 80200015 | Illegal message. | 不合法的消息 |
| 80200018 | Invalid user type. | 不存在的用户类型 |

#### 发送特定PUSH消息

调用从华为网关开放的API：

|  |
| --- |
| openpush.openapi.notification\_send |

openpush.openapi.notification\_send示例代码 (发送给个人及时Android消息)：

|  |
| --- |
| 不推荐使用，建议使用 openpush.message.single\_send  openpush.message.psSingleSend |

openpush.openapi.notification\_send示例代码 (发送给所有人及时Android消息)：

|  |
| --- |
| **public** **static** **void** notification\_send\_all(NSPClient client)  **throws** NSPException  {  //推送范围，必选  //1：指定用户，必须指定tokens字段  //2：所有人，无需指定tokens，tags，exclude\_tags  //3：一群人，必须指定tags或者exclude\_tags字段  Integer push\_type = 2;    //消息内容，必选  //该样例是点击通知消息打开url连接。更多的android样例请参考http://developer.huawei.com/ -> 资料中心 -> Push服务 -> API文档 -> 4.2.1 android结构体  String android =  "{\"notification\_title\":\"the good news!\",\"notification\_content\":\"Price reduction!\",\"doings\":3,\"url\":\"vmall.com\"}";    //消息发送时间，可选  //如果不携带该字段，则表示消息实时生效。实际使用时，该字段精确到分  //消息发送时间戳，timestamp格式ISO 8601：2013-06-03T17:30:08+08:00  String send\_time = "2013-09-03T17:30:08+08:00";    //消息过期时间，可选  //timestamp格式ISO 8601：2013-06-03T17:30:08+08:00  String expire\_time = "2013-09-05T17:30:08+08:00";    //可选  // 0: 当前用户  // 1: 主要用户  // -1: 默认用户  //  String userType = "-1";    //可选  // 1: Android (默认)  // 2: IOS  //  **int** deviteType = 1;    //构造请求  HashMap<String, Object> hashMap = **new** HashMap<String, Object>();  hashMap.put("push\_type", push\_type);  hashMap.put("android", android);  hashMap.put("device\_type", deviteType);  hashMap.put("send\_time", send\_time);  hashMap.put("expire\_time", expire\_time);  hashMap.put("userType", userType);    //设置http超时时间  client.setTimeout(10000, 15000);  //接口调用  String rsp = client.call("openpush.openapi.notification\_send", hashMap, String.**class**);    //打印响应  //响应样例：{"result\_code":0,"request\_id":"1380075138"}  System.***err***.println("通知栏消息接口响应：" + rsp);  } |

openpush.openapi.notification\_send示例代码 (发送给所有人定时Android消息)：

|  |
| --- |
| **public** **static** **void** notification\_send\_all(NSPClient client)  **throws** NSPException  {  //推送范围，必选  //1：指定用户，必须指定tokens字段  //2：所有人，无需指定tokens，tags，exclude\_tags  Integer push\_type = 2;    //消息内容，必选  //该样例是点击通知消息打开url连接。更多的android样例请参考http://developer.huawei.com/ -> 资料中心 -> Push服务 -> API文档 -> 4.2.1 android结构体  String android =  "{\"notification\_title\":\"the good news!\",\"notification\_content\":\"Price reduction!\",\"doings\":3,\"url\":\"vmall.com\"}";    //消息发送时间，可选  //如果不携带该字段，则表示消息实时生效。实际使用时，该字段精确到分  //消息发送时间戳，timestamp格式ISO 8601：2013-06-03T17:30:08+08:00  String send\_time = "2016-09-03T17:30:08+08:00"; //定时发送需要设置发送的时间    //消息过期时间，可选  //timestamp格式ISO 8601：2013-06-03T17:30:08+08:00  String expire\_time = "2016-09-05T17:30:08+08:00";    //可选  // 0: 当前用户  // 1: 主要用户  // -1: 默认用户  //  String userType = "-1";  //可选  // 1: Android (默认)  // 2: IOS  //  **int** deviteType = 1;    //构造请求  HashMap<String, Object> hashMap = **new** HashMap<String, Object>();  hashMap.put("push\_type", push\_type);  hashMap.put("android", android);  hashMap.put("device\_type", deviteType);  hashMap.put("send\_time", send\_time);  hashMap.put("expire\_time", expire\_time);  hashMap.put("userType", userType);    //设置http超时时间  client.setTimeout(10000, 15000);  //接口调用  String rsp = client.call("openpush.openapi.notification\_send", hashMap, String.**class**);    //打印响应  //响应样例：{"result\_code":0,"request\_id":"1380075138"}  System.***err***.println("通知栏消息接口响应：" + rsp);  } |

openpush.openapi.notification\_send参数详解：

|  |  |  |
| --- | --- | --- |
| **参数名称** | **作用** | **备注** |
| push\_type | 1：指定用户，必须指定tokens字段  2：所有人，无需指定tokens，tags，exclude\_tags | 建议只使用2的类型； |
| tokens | 当push\_type为1的时候为特定的用户PUSH TOKEN链表：  String tokens =  "00000000000000000000000000000000,00000000000000000000000000000000"; | 不建议使用，如果是及时下发的消息请使用:  openpush.message.single\_send  openpush.message.batch\_send  openpush.message.psSingleSend  openpush.message.psBatchSend |
| tags | 不用填，暂时不支持 |  |
| exclude\_tags | 不用填，暂时不支持 |  |
| android | 发送给Android设备的PUSH消息体 |  |
| send\_time | 消息发送时间，可选  如果不携带该字段，则表示消息实时生效。实际使用时，该字段精确到分  消息发送时间戳，timestamp格式ISO 8601：2013-06-03T17:30:08+08:00 |  |
| expire\_time | 消息过期时间，可选  timestamp格式ISO 8601：2013-06-03T17:30:08+08:00 |  |
| device\_type | 1：Android；  2：IOS | 由于IOS代理后续将不再支持，这里必须填1 |
| message | 不用关注，原先IOS支持的使用的字段 |  |
| target\_user\_type | 不用关注，原先IOS支持的使用的字段 |  |
| allow\_periods | PUSH消息允许展示时间段，消息到达PUSH Agent客户端后会依据此信息择时进行用户展示，如果没有填写就会立即展示； |  |
| user\_type | android多用户场景下，用户标识。0：当前用户，1：主用户。默认值为-1，表示调用方发消息时，未填写该字段 |  |

openpush.openapi.notification\_send的错误码详解：

|  |  |  |
| --- | --- | --- |
| **错误码** | **错误码描述** | **备注** |
| 80100001 | 根据系统上下文获取devAppId失败描述 |  |
| 80200001 | 请求参数非法 |  |
| 80500001 | 请求的时间格式不对 |  |
| 80300002 | APP没有开通使用PUSH权益，鉴权失败 |  |
| 80200004 | 非法的PUSH TOKEN |  |
| 80300005 | 调用PUSH接口的用户ID和APP无拥有关系 |  |

#### 查询PUSH消息状态

调用从华为网关开放的API：

|  |
| --- |
| openpush.openapi.query\_msg\_result |

openpush.openapi.query\_msg\_result示例代码 ：

|  |
| --- |
| **public** **static** **void** query\_msg\_result(NSPClient client)  **throws** NSPException  {  //开发者调用sengle\_send和batch\_send接口时返回的requestID字段值  String request\_id = "";    //用户标识  //如果携带该字段，则表示查询request\_id中的token对应的消息结果；如果不携带该字段，则查询request\_id对应的所有token的消息结果  String token = "";    //构造请求  HashMap<String, Object> hashMap = **new** HashMap<String, Object>();  hashMap.put("request\_id", request\_id);  hashMap.put("token", token);    //设置http超时时间  client.setTimeout(10000, 15000);  //接口调用  String rsp = client.call("openpush.openapi.query\_msg\_result", hashMap, String.**class**);    //打印响应  //响应样例：{"result":[{"status":0,"token":"00000000000000000000000000000000"}],"request\_id":"123456"}  System.***err***.println("查询查询消息发送结果接口：" + rsp);    } |

openpush.openapi.query\_msg\_result参数详解：

|  |  |  |
| --- | --- | --- |
| **参数名称** | **作用** | **备注** |
| request\_id | 调用发送请求时返回的唯一ID | 必选字段 |
| token | 华为PUSH TOKEN | 用户标识，本字段为可选字段。  如果携带该字段，则表示查询request\_id中的token对应的消息结果；如果不携带该字段，则查询request\_id对应的所有token的消息结果 |

openpush.openapi.query\_msg\_result错误码详解：

|  |  |  |
| --- | --- | --- |
| **错误码** | **错误码描述** | **备注** |
| 80100001 | 根据系统上下文获取devAppId失败描述 | 需要在联盟注册PUSH权益 |
| 80200001 | 请求参数非法 |  |

注意：当前版本如果出错会返回空的状态对象，此时表明没有查询到。

# 限制与约束

在使用华为PUSH平台发送消息时存在如下限制：

## PUSH消息送达率无法保证

影响PUSH消息送达率的因素有：

* 终端设备是否在线
* 用户手机上的APP是否被卸载
* 移动网络是否稳定
* 终端手机的安全控制策略
* 不同终端厂商设备对友商的支持度

鉴于上述原因华为PUSH平台不能保证100%的消息送达率。

华为PUSH平台首发在线（用户第一次发送并且接收消息的终端设备网络正常）成功率接近100%，如果设备离线华为PUSH平台也会给用户缓存消息，待用户上线有将这些消息尽量投递给用户，用户的消息只在平台缓存一次，如果用户在线后又发不通PUSH平台不在对此消息做第二次缓存。

## PUSH消息不同终端平台的限制

* 华为手机终端设备限制

华为手机管家可以拦截Android广播消息，PUSH消息因此也会被拦截。

* 华为Android设备（华为手表、华为平板、小K手表）限制

如果安装了手机管家可以拦截Android广播消息，PUSH消息因此也会被拦截。

* 非华为Android手机（小米、VIVO、OPPO、三星、酷派、中兴）限制

原则上安装了华为PUSH客户端的Android手机即可支持Huawei PUSH业务。某些友商手机可能会删除华为进程，导致消息不可达。

* 苹果手机限制

华为PUSH平台推送到IOS的消息是通过苹果的APNS推送的，这类消息的可靠性受限于PUSH平台和APNS链路的稳定性（网络稳定性、消息报文大小、支持的TPS数），以及APPLE APNS的可靠性和达到率。

## PUSH消息的及时性

首发在线，网络正常，时延在1s内。如果受限于网络环境的情况，消息时延无法保证。

## PUSH消息在终端的显示和交互方式限制

* 透传消息

由第三方APP开发者自己控制，PUSH仅仅提供通道能力。

* 通知栏消息

点击动作支持：打开公网网页、打开APP、打开本地网页

## PUSH消息大小限制

PUSH消息支持的android APP包名称最大为128个字节，消息内容最大限制为2K.

## PUSH平台离线缓存消息限制

设备离线消息缓存时间限制: 最大支持30天，默认为48小时。

单个APP的缓存消息数量限制：如果是覆盖类消息，前面发送的消息会被后续消息覆盖。

## PUSH消息流控限制

对于CP当前提供了两个通道：

* Open API

<http://open.hicloud.com:8080/PushCRS/AcceptDataServlet>

Open API有流控限制，用户短时间内发送大量的消息会被流控。

* 网关的通道

网关上发送消息限制于用户申请的PUSH权益。

## PUSH功耗流量限制

PUSH客户端和服务器在网络正常的情况下会保持长连接，实际应用中用户会经常开启和关闭手机的网络服务获取切换基站，导致PUSH客户端和PUSH服务器经常需要重新建立连接。在设计时PUSH客户端和服务器重连的次数会被控制（当前版本控制1天重连次数、1周重连次数和1月的重连次数），如果用户手机网络开启关闭过于频繁会导致无法及时收到消息。

## 安全性约束

PUSH平台只提供通道业务，如果用户使用PUSH传输银行卡、用户密码等隐私数据需要由APP供应商自己保证安全性。当前PUSH服务器和华为终端设备采用的加密算法是RSA 1024.

# FAQ

1. PUSH TOKEN为PUSH通知消息提供了消息路由和鉴权的机制 [↑](#footnote-ref-1)