**创建机器人**

[https://open-dev.popo.netease.com/robot](https://open-dev.popo.netease.com/robo)

#### 





**自定义机器人**

自定义机器人是一种仅支持在单一群聊中进行消息推送的机器人。

任何在群内具有机器人管理权限的群成员都可以在群内快速的创建和管理自定义机器人，且无需管理员进行审批。创建后，用户可通过调用 webhook 地址的形式通过自定义机器人向群内推送消息。

**创建自定义机器人**

1. 打开目标群组并在右上角的「更多」中选中「机器人」



1. 点击列表中的「添加机器人」按钮，在添加机器人弹窗中选择自定义机器人并点击「创建」按钮



1. 设置自定义机器人的头头像、名称以及简介，并点击下一步

**获取 webhook 地址**

请妥善保管 webhook 地址，不要公布在 Github 等公开网站上，以防泄露后被恶意使用发送垃圾信息

**为自定义机器人进行安全配置**



1. 安全设置可以避免 webhook 地址被恶意调用后向群内推送一些不相关信息；
2. 在配置了安全设置后，仅当符合安全设置的消息才可以通过调用 webhook 地址将消息推送至群内。
3. 每个自定义机器人需至少配置一个安全配置项

**安全配置项**

**安全设置一：自定义关键词**

1. 仅当消息中包含已设置的关键词时，消息才会发送，否则会返回错误
2. 最多可添加 10 个关键词，单个关键词不设置字数限制。
3. 消息中的关键词需精准匹配

**安全设置二：签名校验**

1. 设置签名校验后，向 webhook 发送请求需要签名校验来保障来源可信。所校验的签名需要通过时间戳与秘钥进行算法加密，即将 timestamp + "\n" + 密钥当做签名字符串，使用 HmacSHA256 算法计算签名，再进行 Base64 编码。其中，timestamp 是指距当前时间不超过 1 小时（3600 秒）的时间戳，时间单位：ms。例如，1599360473000。
2. 时间戳需要在发送时间的 1 小时内，即请求参数时间戳减去当前时间戳小于 3600s
3. 签名生成方式,以下提供了不同语言的代码示例，用于计算获得签名字符串。

Java 示例代码：

package sign;

import javax.crypto.Mac;

import javax.crypto.spec.SecretKeySpec;

import java.nio.charset.StandardCharsets;

import java.security.InvalidKeyException;

import java.security.NoSuchAlgorithmException;

import org.apache.commons.codec.binary.Base64;

public class SignDemo {

public static void main(String[] args) throws NoSuchAlgorithmException, InvalidKeyException {

String secret = "NEe9bmeJuidgFZT0h0ViDrTijYnFC5oRUw7TwIwNIF1BE9e0zZvLSvBHG2T1AJ2f";

long timestamp = System.currentTimeMillis();

System.out.printf("timestamp: %d\n", timestamp);

System.out.printf("sign: %s", GenSign(secret, timestamp));

}

private static String GenSign(String secret, long timestamp) throws NoSuchAlgorithmException, InvalidKeyException {

//把timestamp+"\n"+密钥当做签名字符串

String stringToSign = timestamp + "\n" + secret;

//使用HmacSHA256算法计算签名

Mac mac = Mac.getInstance("HmacSHA256");

mac.init(new SecretKeySpec(stringToSign.getBytes(StandardCharsets.UTF\_8), "HmacSHA256"));

byte[] signData = mac.doFinal(new byte[]{});

return new String(Base64.encodeBase64(signData));

}

}

Go 示例代码：

func GenSign(secret string, timestamp int64) (string, error) {

//timestamp + key 做sha256, 再进行base64 encode

stringToSign := fmt.Sprintf("%v", timestamp) + "\n" + secret

var data []byte

h := hmac.New(sha256.New, []byte(stringToSign))

\_, err := h.Write(data)

if err != nil {

return "", err

}

signature := base64.StdEncoding.EncodeToString(h.Sum(nil))

return signature, nil

}

Python 示例代码：

import hashlib

import base64

import hmac

def gen\_sign(timestamp, secret):

# 拼接timestamp和secret

string\_to\_sign = '{}\n{}'.format(timestamp, secret)

hmac\_code = hmac.new(string\_to\_sign.encode("utf-8"), digestmod=hashlib.sha256).digest()

# 对结果进行base64处理

sign = base64.b64encode(hmac\_code).decode('utf-8')

return sign

**安全设置三：IP 白名单**

1. 设定后，只有来自 IP 地址范围内的请求才会被处理
2. 最多支持设置 10 个 IP 地址或地址段。

完成创建：

1. 完成以上所有设置后，即可通过调用 webhook 地址通过自定义机器人向群内进行消息推送。

**发送消息**

接口描述：

`POST /open-apis/robots/v1/hook/{{webhookIdentify}}`

请求参数:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 参数类型 | 是否必填 | 请求方式 | 说明 |
| message | String | 是 | RequestBody | 发送的消息 |
| timestamp | String | 否 | RequsetBody | 毫秒时间戳，  signData  不为空时必填 |
| signData | String | 否 | RequsetBody | 签名数据，开启签名校验必填 |

请求示例：

curl --location --request POST 'https://open.popo.netease.com/open-apis/robots/v1/hook/EmoPAOOgO3OT3n00h6EmCLmPkrvZAWxE' \

--header 'Content-Type: application/json' \

--data-raw '{

"message": "1234444",

"timestamp": "1704368223658",

"signData": "uSV3idOXZi7LPiw2V2rTKh2Kp+cDBnpcA2VjppY9bCQ="

}'

响应参数：

|  |  |  |
| --- | --- | --- |
| 参数名称 | 参数类型 | 说明 |
| msgId | String | 发送消息 id |

{

"data": {

"msgId": "24584304-0000000003"

},

"errcode": 0,

"errmsg": "ok"

}

注意事项：

1、时间戳需要在发送时间的 1 小时内，即请求参数时间戳减去当前时间戳小于 3600s；

2、每分钟最多发送 60 条消息，超出限制后会被禁止 10 分钟，10 分钟后自动解封。

**权限申请**

开放平台接口需要通过邮件的方式申请，申请前请查看要申请哪些接口

服务端API：<https://open.popo.netease.com/docs/robot/api/message-api>

事件订阅：<https://open.popo.netease.com/docs/robot/subscribe/message>

**1、邮件申请**

**1.1 邮件标题**

邮件申请说明如下（申请后需要主管回复批准才会处理，工作日当天会处理）：

1.1 邮件标题

POPO 开放平台机器人接口申请(需要主管审批)

**1.2 邮件内容：**

1. 机器人中文名称： xxx
2. 机器人appid：xxx
3. 业务简介：xxx
4. 对接人：xxx
5. 需要申请的接口：
6. 需要申请对应接口的理由：（请务必完整描述接口使用场景及用途）

接口申请格式为 "接口名称"： "接口路径"；“事件订阅名称”：“事件订阅类型” 示例如下：（请务必按格式进行填写）

服务端API

(1) 机器人发送消息：/POST /open-apis/robots/v1/im/send-msg

...

事件订阅

(1) 用户发送给机器人的消息：IM\_P2P\_TO\_ROBOT\_MSG

...

**1.3 邮件发送：**

1. 收件人：您的主管（需审批确认）;
2. 抄送人：张以立(zhangyili@corp.netease.com); 刘劲宇(gzliujingyu@corp.netease.com);任忠(renzhong01@corp.netease.com);岳喜春(yuexichun@corp.netease.com);

**1.4 其他注意事项**

1、申请说明：请大家根据需要按照最小权限申请，后续如果需要追加权限，需要主管再次确认

2、接口申请后，请先在开发者后台「机器人-版本管理与发布」中，将可使用范围配置为相关开发人员（后期和进行修改）后，进行申请发布；若已发布则忽略

3、接口申请通过后管理员会发送开通成功的回复邮件，对接人可查看[接口调用说明](https://open.popo.netease.com/docs/api/intro/apicall)和 [获取授权凭证](https://open.popo.netease.com/docs/robot/access-token) 以及对应接口的文档进行对接。 AppId、AppSecret 和 Open-Access-Token 不可以直接在前端或客户端保存和传输，接口申请人和对接人有义务保证申请账号的使用安全性。

**1.5 邮件模板**

邮件模板

目前申请新机器人：要现申请机器人，申请完之后要发送邮箱：参考这个<https://open.popo.netease.com/docs/robot/start/robot-api-apply>



**2、接口调用参考文档**

申请开通之后请先阅读接口调用文档：

<https://open.popo.netease.com/docs/robot/access-token>



