Dialog Insight

Smart Marketing Catalyst

Webhooks

User Guide

11/8/2017

Table of Contents

Best practices

Choosing the right subscription

Dialog Insight offers multiple subscription types, each conducting to a large number of notifications. So it is important to properly choose the data you need in order to subscribe only to these specific notifications and not to all.

Processing notifications

Once a webhook is enabled, Dialog Insight will send all the requested notifications. Depending on the number of interactions done by contacts, or based on the number of errors, there can be quite a lot of notifications. So it is not recommended to process these notifications as they arrive since any processing error or delay can cause notifications to be lost.

The purpose of the system that receives webhook calls is to accept the responsibility of the notification as soon as possible and to confirm its reception.

For this reason, we recommend you use an asynchronous system that automatically registers all the notifications sent by Dialog Insight in a database or file. Then, you can use another automated process to manage one notification at a time to produce the final result – add contacts to your CRM, update contacts based on your rules, etc.

Delivery attempts

A notification is sent as soon as a targeted action is performed in Dialog Insight's platform using a POST (http / https). This POST has a 15 second timeout. Then, we will make 4 more attempts in the next 2 hours to resend the notification if we have not received an http response with an OK code (code = 200). These attempts are scheduled as follows:

- 1st attempt: when action is performed in application
- 2nd attempt: 1 minute later
- 3rd attempt: 15 minutes later (so 16 minutes after the action was performed)
- 4th attempt: 30 minutes later (so 46 minutes after the action occurred)
- 5th attempt: 60 minutes later (20 106 minutes after the action was done)

Errors

If the notification still can't be delivered after these attempts, it is placed in error and abandoned.

Automatic deactivation

Notifications are automatically deactivated if you experiment a system failure, to prevent a large number of messages blocked after multiple attempts.

Your system will be considered in failure when at least 100 consecutive errors have occurred within a minimum of 8 hours.

This webhook will then be deactivated and an alert sent to users set to receive alerts related to this webhook.

To reactivate notifications, you must activate it again through the platform. Note that notifications that should have been sent during the deactivation period will be lost (as well as notifications that were pending during this period).

Email alerts

Alerts are sent by email in the following situations:

- After the 1st, 10th, 100 th, 1000 th, 10000 th error of each day (based on Dialog Insight' server time).
- When your webhook has been automatically deactivated.

Notification information

All notifications are sent in JSON format, and always start with an identifying parameter:

- *Type* – contains a chain of characters that represents the type of notification sent. This parameter can have such values as:
  - "*sending_Bounce"* for a delivery error
  - "*sending_ProductionError"* for a production error
  - "*contact_modified"* for a contact modification
  - "*contact_optout"* for a contact opt-out
  - "*contact_complaint"* for a spam complaint formulated by a contact
  - Etc. (see documentation for the complete list)
- "*EventUniqueID*" contains a chain of characters that uniquely identifies the notification
- "*dtExecution*" for the exact date and time the notification was sent
- "*idCompany*" contains the company's ID
- "*idProject*" contains the ID of the project where the notification is set
- "*isTest*" is present and contains "*true"* if this is a test notification sent from the webhook configuration interface; and is not present when a real notification is sent.
- "*ContactID*" contains a dictionary of values that identifies the contact, that is :
  - "*idContact*", the contact's ID in Dialog Insight's database.

As well as all the fields which are part of the project's primary key.

Note that the system must accept the *isTest* parameter as follows: if this parameter is "*true*", you must not try to deactivate a real contact or edit a profile upon reception of a test message.

Types de supported notifications

Messages

Error notifications for messages contain the parameters included in all system notifications, as well as:

- Information on the error (shown in the «*SendLog*" dictionary):
  - "*EventType"* contains one of the two possible mailing types in error:
    - "Batch" for batch mailings or campaigns
    - "Event" for single message mailings (confirmations, Web Services, tests)
  - "*idSendLog*" contains the ID of the mailing in error
  - "*idMessage"* contains the ID of the message used for the mailing
  - "*idMessageCategory"* contains the ID of the communication type used for the message
  - "*dtProcessed*" contains the date and time the mailing started to be sent

Delivery errors

Notifications for delivery errors inform you when a delivery error occurs, including when a contact is quarantined.

These notifications contain common parameters such as identifiers, as well as information on the mailing and on the error (shown in the "*DeliveryErrorInfo*" dictionary):

- "*dsnMTA"* contains the domain name of the MX server where the exchange occurred
- "*dsnDiag"* contains the error code sent from the server, and which caused the quarantine
- "*BounceCode"* contains the SMTP error code
- "*isInvalidMailbox"* indicate whether the email address is invalid or not.
- "*BounceType"* contains the type of error - "*BounceBack*" or "*Bounce*"

Example of sent notification:


```
[
  {
    "type": "sending_Bounce",
    "EventUniqueID": "77cb9126-661a-43b9-9915-1c8f9e826f93",
    "dtExecution": "2016.09.19 10:54:49-04:00",
    "idCompany": 1,
    "idProject": 1234,
    "isTest": true,
    "ContactID": {
      "f_EMail": "EMail",
      "idContact": 1
    },
    "SendLog": {
      "EventType": "Batch",
      "idSendLog": 1,
      "idMessage": 1,
      "idMessageCategory": 456,
      "dtProcessed": "2016.09.19 10:54:49-04:00"
    },
    "DeliveryErrorInfo": {
      "dsnMTA": "mail.exemple.com",
      "dsnDiag": "smtp;550 5.1.1 The email account that you
tried to reach does not exist.",
      "BounceCode": "5.1.1",
      "isInvalidMailbox": false,
      "BounceType": "BounceBack"
    }
  }
]
```


Production errors

This type of notification allows you to be informed when an error occurs when the message is being prepared, before it is even sent.

These notifications contain common parameters, as well as information on the mailing and on the production error (shown in the "*ProductionErrorInfo*" dictionary):

- *"ErrorCode"* contains an integer number representing the code of the error below:
- *"ErrorMessage"* contains the description of the error

ErrorCode	Error	ErrorMessage
	System Errors	
0	NoError	No error
1	RecipientEmailSyntax	Recipient email syntax is invalid.
2	Compilation	There was a compilation error in the message.
3	Execution	There was an execution error in the message.
4	SenderEmailSyntax	Sender email syntax is invalid.
5	Submission	An unknown internal error occurred.
6	RequiredDataMissing	An unknown internal error occurred.
7	EligibilityError	The contact was not eligible to receive the message.
99	Undocumented	An unknown internal error occurred.
	Client Errors	
100	Generic	An unknown internal error occurred.
101	MissingData	An unknown internal error occurred.
102	InvalidData	An unknown internal error occurred.
103	NoContent	An unknown internal error occurred.

Example of sent notification:


```
[
  {
    "type": "sending_ProductionError",
    "EventUniqueID": "941a8d77-13f0-48f8-be49-9056f71d266d",
    "dtExecution": "2016.09.19 11:23:29-04:00",
    "idCompany": 1,
    "idProject": 1234,
    "isTest": true,
    "ContactID": {
      "f_EMail": "EMail",
      "idContact": 1
    },
    "SendLog": {
      "EventType": "Batch",
      "idSendLog": 1,
      "idMessage": 1,
      "idMessageCategory": 6098,
      "dtProcessed": "2016.09.19 11:23:29-04:00"
    },
    "ProductionErrorInfo": {
      "ErrorCode": "NoContent",
      "ErrorMessage": "An unknown internal error occured."
    }
  }
]
```

Contacts

Creation

This type of notification informs you when a new contact is created in Dialog Insight, either manually or through a form. No notification can be sent for contacts added through import.

Notifications related to the creation of a contact contains the same parameters as the system parameters, as well as all the project fields and their values (shown in the *"ContactData"* dictionary). This dictionary contains the values of all the project fields created by the client (where each field has a code prefixed with "f_"), as well as the administrative fields (source of contact creation, creation date, activation date, etc.) Here is an example of the sent notification:


```
[
  {
    "type": "contact_created",
    "EventUniqueID": "b221df6e-3d8a-485f-84f0-c9ff8e259f45",
    "dtExecution": "2016.09.19 11:24:15-04:00",
    "idCompany": 1,
    "idProject": 1234,
    "isTest": true,
    "ContactID": {
      "f_EMail": "EMail",
      "idContact": 1
    },
    "ContactData": {
      "f_FirstName": "FirstName",
      "f_LastName": "LastName",
      "dtActivated": null,
      "dtDeactivated": null,
      "SourceIndex_Creation": null,
      "idSource_Creation": null,
      "idSource_Modification": null,
      "idSource_Activation": null,
      "idSource_Deactivation": null,
      "isActive": null,
      "clKey": "xxxxxx",
      "dtCreated": null,
      "dtModified": null,
      "dtInvalidMailbox": null,
      "isInvalidMailbox": null,
      "Domain": null
    }
  }
]
```

Modification

This type of notification lets you be informed whenever a contact's profile is updated in Dialog Insight, either manually or through a form. No notification can be sent for contacts modified through import.

As for contact creation, notifications on contact edits contain the parameters included in the system parameters, as well as the project fields and their values, after the change (shown in the *"ContactData"* dictionary).

Example de sent notification:


```
[
  {
    "type": "contact_modified",
    "EventUniqueID": "2cfdf919-5c96-4ecf-8fbb-0e0eefa4196e",
    "dtExecution": "2016.09.19 11:24:40-04:00",
    "idCompany": 1,
    "idProject": 1234,
    "isTest": true,
    "ContactID": {
      "f_EMail": "EMail",
      "idContact": 1
    },
    "ContactData": {
      "f_FirstName": "FirstName",
      "f_LastName": "LastName",
      "dtActivated": null,
      "dtDeactivated": null,
      "SourceIndex_Creation": null,
      "idSource_Creation": null,
      "idSource_Modification": null,
      "idSource_Activation": null,
      "idSource_Deactivation": null,
      "isActive": null,
      "clKey": "xxxxxx",
      "dtCreated": null,
      "dtModified": null,
      "dtInvalidMailbox": null,
      "isInvalidMailbox": null,
      "Domain": null
    }
  }
]
```

Opt-in

This type of notification informs you when a contact subscribes to one of your communications and when the contact's status passes to Subscribed, or Active if the project does not have a consent center (C-28).

Opt-in notifications contain parameters included in the system notifications, as well as the value of the field *isActive*, and the list of opt-in fields. The *"OptinInfo"* list has 2 objects:

- *"OptinField"* contains the code of the opt-in field
- *"OptinDate"* contains the date the contact subscribed

Example of sent notification:


```
[
  {
    "type": "contact_optin",
    "EventUniqueID": "408922e7-4e8b-456b-a1a0-2956be9a3a69",
    "dtExecution": "2016.09.22 11:00:19-04:00",
    "idCompany": 1,
    "idProject": 1234,
    "isTest": true,
    "ContactID": {
      "f_EMail": "EMail",
      "idContact": 1
    },
    "OptinInfo": {
      "Fields": [
        {
          "OptinField": "infolettre",
          "OptinDate": "2016.09.22 11:00:19-04:00"
        },
        {
          "OptinField": "infolettre2",
          "OptinDate": "2016.09.22 11:00:19-04:00"
        }
      ]
    },
    "isActive": true
  }
]
```

Opt-out

This type of notification is used to inform you when a contact has unsubscribed from a communication and his status has changed to Unsubscribed, or Inactive if the project does not have a consent center (C-28).

Opt-out notifications contain parameters included in the system parameters, as well as the value of the field *isActive*, and the list of opt-in fields. The *"OptinInfo"* list has 2 objects:

- *"OptinField"* contains the code of the opt-in field
- *"OptinDate"* contains the date the contact unsubscribed

Example of sent notification:


```
[
  {
    "type": "contact_optout",
    "EventUniqueID": "e92268e6-0b0c-4062-8b59-09b445133cd7",
    "dtExecution": "2016.09.22 11:30:30-04:00",
    "idCompany": 1,
    "idProject": 1234,
    "isTest": true,
    "ContactID": {
      "f_EMail": "EMail",
      "idContact": 1
    },
    "OptinInfo": {
      "Fields": [
        {
          "OptinField": "infolettre",
          "OptinDate": "2016.09.22 11:30:30-04:00"
        },
        {
          "OptinField": "infolettre2",
          "OptinDate": "2016.09.22 11:30:30-04:00"
        }
      ]
    },
    "isActive": false
  }
]
```

Quarantine

This type of notification informs you when a contact's email address is placed in quarantine after a message has been sent.

Quarantine notifications contains the parameters included in all system parameters, as well as all available information about the quarantine (shown in the *"QuarantineInfo"* dictionary):

- *"idSendLog"* contains the ID of the mailing that conducted to the quarantine
- *"idMessage"* contains the ID of the message used for the mailing
- *"dtProcessed"* contains the date and time the mailing entered production
- *"dtBounce"* contains the date and time the message return the quarantine error
- *"dtDelivered"* contains the date and time the message was delivered
- *"dsnMTA"* contains the domain name of the MX server where the exchange occurred
- *"dsnDiag"* contains the error code sent by the server, which has cause the quarantine

Example of sent notification:


```
 [
  {
    "type": "contact_quarantine",
    "EventUniqueID": "de0e65c1-fe4d-43ef-b8b9-9a4b217d4ff6",
    "dtExecution": "2016.09.19 11:25:52-04:00",
    "idCompany": 1,
    "idProject": 1234,
    "isTest": true,
    "ContactID": {
      "f_EMail": "EMail",
      "idContact": 1
    },
    "QuarantineInfo": {
      "idSendLog": 1,
      "idMessage": 1,
      "dtProcessed": "2016.09.19 11:15:52-04:00",
      "dtBounce": "2016.09.19 11:25:52-04:00",
      "dtDelivered": null,
      "dsnMTA": "mail.exemple.com",
      "dsnDiag": "smtp;550 5.1.1 The email account that you
tried to reach does not exist."
    }
```

```
  }
]
```


Spam complaint

This type of notification informs you when a contact reports a message as spam.

Spam complaint notifications contain the parameters included in all system parameters, as well as all available information about the complaint – the *feedbackLoop* report shown in the *"FBL_Report"* parameter.

Example of sent notification:


```
[
  {
    "type": "contact_complaint",
    "EventUniqueID": "25673f56-8222-4af4-b563-1e967304cf61",
    "dtExecution": "2016.09.19 11:26:11-04:00",
    "idCompany": 1,
    "idProject": 1234,
    "isTest": true,
    "ContactID": {
      "f_EMail": "EMail",
      "idContact": 1
    },
    "ComplaintInfo": {
      "FBL_Report": "\r\nFrom: <abusedesk@example.com>\r\nDate:
Thu, 8 Mar 2005 17:40:36 EDT\r\nSubject: FW: Earn money\r\nTo:
<abuse@example.net>\r\nMIME-Version: 1.0\r\nContent-Type:
multipart/report; report-type=feedback-report;\r\n
boundary=\"part1_13d.2e68ed54_boundary\"\r\n \r\n--
part1_13d.2e68ed54_boundary\r\nContent-Type: text/plain;
charset=\"US-ASCII\"\r\nContent-Transfer-Encoding: 7bit\r\n
\r\nThis is an email abuse report for an email message received
from IP\r\n10.67.41.167 on Thu, 8 Mar 2005 14:00:00 EDT. For
more information\r\nabout this format please see
http://www.mipassoc.org/arf/.\r\n \r\n--
part1_13d.2e68ed54_boundary\r\nContent-Type: message/feedback-
report\r\n \r\nFeedback-Type: abuse\r\nUser-Agent:
SomeGenerator/1.0\r\nVersion: 1\r\nOriginal-Mail-From:
<somespammer@example.net>\r\nOriginal-Rcpt-To:
<user@example.com>\r\nReceived-Date: Thu, 8 Mar 2005 14:00:00
EDT\r\nSource-IP: 10.67.41.167\r\nAuthentication-Results:
mail.example.com\r\n
smtp.mail=somespammer@example.com;\r\n
spf=fail\r\nReported-Domain: example.net\r\nReported-Uri:
```

```
http://example.net/earn_money.html\r\nReported-Uri:
mailto:user@example.com\r\nRemoval-Recipient:
user@example.com\r\n \r\n--
part1_13d.2e68ed54_boundary\r\nContent-Type:
message/rfc822\r\nContent-Disposition: inline\r\n \r\nFrom:
<somespammer@example.net>\r\nReceived: from
mailserver.example.net (mailserver.example.net\r\n
[10.67.41.167]) by example.com with ESMTP id M63d4137594e46;\r\n
Thu, 8 Mar 2005 14:00:00 -0400\r\nTo: <Undisclosed
Recipients>\r\nSubject: Earn money\r\nMIME-Version:
1.0\r\nContent-type: text/plain\r\nMessage-ID:
8787KJKJ3K4J3K4J3K4J3.mail@example.net\r\nDate: Thu, 2 Sep 2004
12:31:03 -0500\r\n \r\nSpam Spam Spam\r\nSpam Spam Spam\r\nSpam
Spam Spam\r\nSpam Spam Spam\r\n--
part1_13d.2e68ed54_boundary\r\n"
    }
  }
]
```

Managing received notifications

It is suggested to accept all notifications sent by Dialog Insight and to process them one by one after. Here is an example of the C# code that lets you accept these notifications in a .ashx file:

```
<%@ WebHandler Language="C#" Class="WebHookHandlerExample" %>

using System;
using System.Web;
using System.IO;
using System.Text;

public class WebHookHandlerExample : IHttpHandler
{
    public void ProcessRequest (HttpContext context)
    {
        // Read the posted JSON body
        string json;
        using (StreamReader stream = new
          System.IO.StreamReader(context.Request.InputStream, Encoding.UTF8))
        {
            json = stream.ReadToEnd();
        }

        // Do not process event contents here : queue the event to be
           processed asychronously.
        // Exemple: dumping to a file
        string filename = String.Format(@"C:\Temp\webhook-event-{0}.json",
                          Guid.NewGuid().ToString());
        File.WriteAllText(filename, json);
    }

    public bool IsReusable
    {
        get { return false; }
    }
}
```

And a PHP example:

```php
<?php
// Read the posted JSON body
$json = file_get_contents('php://input');

// Do not process event contents here : queue the event to be processed
   asychronously.
// Exemple: dumping to a file
$myfile = file_put_contents('C:\\temp\\webhook-event-' . uniqid('', true) .
'.json', $json);
?>
```

Contact
Canada: 1 866 529-6214
France: 01 84 88 40 66
Russia: +7 (495) 226-04-11

Email: info@dialoginsight.com
Website : www.dialoginsight.com
Blog: www.dialoginsight.com/en/resources/academy

@DialogInsight
Dialog Insight