API Documentation

CitCALL

Version 3.3

Date	: 30 January 2019
Version	: 3
Prepared by	: Muhammad Chairul

Document History

Version	Author (s)	Date	Description
1.0	Muhammad Chairul	5 Feb 2018	
3.0	Muhammad Chairul	4 Jun 2018	Update using API key
3.1	Muhammad Chairul	18 Aug 2018	Add Verify Method
3.2	Muhammad Chairul	06 Oct 2018	Revise Verify Method
3.3	Muhammad Chairul	30 Jan 2019	Update and add SMS OTP method.

1. Introduction

This document will provide instructions on how to integrate CITCALL services by using CITCALL HTTP application programming interface (HTTP API). Use HTTP API for making miscall. CITCALL's API is based on REST standards, enabling you to use your browser for accessing URLs. Use any HTTP client in any programming language to interact with our API.

CONTENTS

2.Authorization

2.1 IP Whitelist

We only allow incoming request from your registered IP(s). You can register your IP(s) at our Dashboard on option menu.

2.2. Authorization methods

The majority of requests to CITCALL's API require authentication. That can be done by setting the Authorization HTTP header. The Authorization header must include a type and the credentials themselves.

Authorization: <type> <credentials>

There are two different authorization types supported by the CITCALL API.

type	format	notes
Apikey	Citcall Generated Apikey	Recommended.
Basic	Base64 encoded userid and password combination	Not recommended because the password is included in every request.

2.2.1 API key authorization

You can find out more about API key creation and management at our Dashboard on API menu.

API key Authorization header example:

Authorization: Apikey 0090kijj11208jdnclandjakw9281id0

2.2.2 Basic authorization

Basic authorization type can be used in situations when the API key is not available. For example, API methods for generating API keys should be authenticated with the Basic type.

In this case, the credentials included in the Authorization header should be a Base64 encoded userid and password combination. More formally, basic authentication header can be constructed in three steps:

- Username and password are concatenated using the colon (:) as a separator **userid:password**.
- The resulting string is encoded using the RFC2045-MIME variant of Base64.
- Encoded string is added as credentials after the "Basic" type.

```
Userid: "citcall"
Password: "citcall"
Concatenated string: "citcall:citcall"
Base64 encoded string: "Y2l0Y2FsbDpjaXRjYWxs"
Authorization header: "Basic Y2l0Y2FsbDpjaXRjYWxs"
```

3. API reference

You will interact with CITCALL API using JSON POST on a single URL.

These are the basic rules to integrate with CITCALL API system:

- Using HTTP POST
- Header Content-Type should be application/json

Example request

Raw data:

```
{
        "msisdn":"081234567890",
        "gateway":1
}
```

Example response of successful request:

```
{
        "rc":"00",
        "trxid":"20170709083044690027711524",
        "msisdn":"+6281234567890",
        "via":"voice",
        "token":"622130401234",
        "msisdn":"622130401234",
        "dial_code":"200",
        "dial_status ":"OK",
        "call_status":"ANSWERED",
        "result":"Success",
        "currency": "IDR",
        "price": "165"
}
```

Example response of failed request:

```
{
        "rc":"98",
        "info":" Authorization
failed"
```

3.1 Asynchronous Miscall

HTTP Request

POST http://104.199.196.122/gateway/v3/asynccall

Request structure

To provide miscall with asynchronous method using the Citcall API you need to submit a JSON object to the url defined above. The JSON object takes the following properties:

Parameter	Mandatory	Notes
msisdn	yes	End-User mobile number
gateway	Yes	Gateway number (0,1,2) *Usage for* Gateway number (0) – initial M-OTP sending Gateway number (1) – fail to send M-OTP & 1st retry again Gateway number (2) – fail to send M-OTP & 2nd retry again If still fail then we suggest to give pop up box with no signal notification for user to get better signal
valid_time	No	Time in second for valid OTP (If this parameter exist you will be able to do verify later).
limit_try	no	Maximum limit retry for verify with maximum allowed 20 (If this parameter doesn't exist will be set automatically to 5).

The response returns the following:

- **rc**: Respon Code. (see appendix Appendix 5.1)
- **trxid**: unique message ID automatically generated by Citcall.
- **msisdn**: End-User mobile number.
- **token**: Number received by the end user.
- **gateway**: Gateway number for the trial number of time.
    - **For initial OTP** : use gateway 0
    - **For fail & send again** : use gateway 1
    - **is**& so on.

3.2 SMS

HTTP Request

POST http://104.199.196.122/gateway/v3/sms

Request structure

To provide SMS using the Citcall API you need to submit a JSON object to the url defined above. The JSON object takes the following properties:

Parameter	Mandatory	Notes
msisdn	yes	End-User mobile number
senderid	yes	Your Registered Senderid (use "CITCALL" for default).
text	yes	SMS body (ie: text of the message)

The response returns the following:

- **rc:** Respon Code. (see appendix Appendix 5.1)
- **info:** Status of the message(s).
- **sms_count:** The number of parts the sent SMS was split into**.**
- **senderid:** senderid from request.
- **msisdn:** Recipients information.
- **text:** The text or the message (or SMS body) information.
- **trxid:** unique message ID automatically generated by Citcall.
- **currency:** billing currency for price.
- **price:** price billed by CITCALL for the Transaction.

3.3 SMS OTP

HTTP Request

POST http://104.199.196.122/gateway/v3/smsotp

Request structure

To provide SMS OTP or PREMIUM SMS using the Citcall API you need to submit a JSON object to the url defined above. The JSON object takes the following properties:

3.3.1 Default

Parameter	Mandatory	Notes
msisdn	yes	End-User mobile number
senderid	yes	Your Registered Senderid (use "CITCALL" for default).
text	yes	SMS body (ie: text of the message)

The response returns the following:

- **rc:** Respon Code. (see appendix Appendix 5.1)
- **info:** Status of the message(s).
- **sms_count:** The number of parts the sent SMS was split into**.**
- **senderid:** senderid from request.
- **msisdn:** Recipients information.
- **text:** The text or the message (or SMS body) information.
- **trxid:** unique message ID automatically generated by Citcall.
- **currency:** billing currency for price.
- **price:** price billed by CITCALL for the Transaction.

3.3.2 Advanced

Parameter	Mandatory	Notes

msisdn	yes	End-User mobile number
senderid	yes	Your Registered Senderid (use "CITCALL" for default).
text	optional	SMS body (ie: text of the message). You can set your static text at our Dashboard. If you have set static text on our dashboard this parameter is no longer required. if not, then this parameter is required . If this paramater exist we will overide static text. we will change the word XXXXX with the OTP code in the text. example: Your text = '**your OTP code is XXXXX valid until 1 hour**'. Send text = `**your OTP code is 12345 valid until 1 hour**`.
token	optional	Your OTP code. if you had static text at our dashboard and this parameter doesn't exist we will create 5 digit code automatically.
valid_time	no	Time in second for valid OTP (If this parameter exist you will be able to do verify later).
limit_try	no	Maximum limit retry for verify with maximum allowed 20 (If this parameter doesn't exist will be set automatically to 5).

The response returns the following:

- **rc:** Respon Code. (see appendix Appendix 5.1)
- **info:** Status of the message(s).
- **sms_count:** The number of parts the sent SMS was split into**.**
- **senderid:** senderid from request.
- **msisdn:** Recipients information.
- **text:** The text or the message (or SMS body) information.
- **trxid:** unique message ID automatically generated by Citcall.
- **currency:** billing currency for price.
- **price:** price billed by CITCALL for the Transaction.
- **token:** OTP code for verify.

3.4 Verify

This section provide the information for verify token after miscall delivered.

HTTP Request

POST http://104.199.196.122/gateway/v3/verify

Request structure

To provide verification using the Citcall API you need to submit a JSON object to the url defined above. The JSON object takes the following properties:

Parameter	Mandatory	Notes
trxid	yes	Trxid from miscall request response
msisdn	yes	End User Phone number
token	yes	Token for verify

The response returns the following:

- **rc:** Respon Code. (see appendix Appendix 5.1)
- **info:** this field describes the status code and provides additional information explaining the status.
- **trxid:** unique message ID automatically generated by Citcall.
- **trxid_verify:** trxid from miscall request.
- **currency:** billing currency for price.
- **msisdn:** End-User mobile number.
- **token:** Number received by the end user.
- **token_verify:** Token from Request.

4. CALLBACK

4.1 Asynchronous Miscall Callback

Parameters of Asynchronous Miscall Callback:

Code	Description
rc	Response Code. (see appendix Appendix 5.1)
trxid	unique message ID automatically generated by Citcall.
msisdn	End-User mobile number.
via	Route used to end-user.
token	Number received by the end user.
dial_code	Dial code. (see appendix Appendix 5.2)
dial_status	Dial Status. (see appendix Appendix 5.2)
call_status	Call Status. (see appendix Appendix 5.2)
price	price billed by CITCALL for the Transaction.
result	Result.

Notes:

- Callback URL: provided by client.
- Callback using JSON POST.
- To add callback on dashboardIt is still manually doing by Citcall's administrator, please send the callback url.

4.2 SMS CALLBACK

Parameters of SMS Callback:

Code	Description
trxid	unique message ID automatically generated by Citcall.
result	the status of the message.
description	description of the result.
reportedDate	time stamp of reported message. (yyyy-mm-dd hh:mm:ss)
currency	billing currency for price.
price	price billed by CITCALL for the Transaction.

Notes:

- Callback URL: provided by client.
- Callback using JSON POST.
- To add callback on dashboardIt is still manually doing by Citcall's administrator, please send the callback url

4.2 SMS OTP CALLBACK

Parameters of SMS OTP Callback:

Code	Description
trxid	unique message ID automatically generated by Citcall.
result	the status of the message.
description	description of the result.
reportedDate	time stamp of reported message. (yyyy-mm-dd hh:mm:ss)
currency	billing currency for price.
price	price billed by CITCALL for the Transaction.
token	OTP code for verify. (if available)

Notes:

- Callback URL: provided by client.
- Callback using JSON POST.

To add callback on dashboardIt is still manually doing by Citcall's administrator, please send the callback url

5 APPENDIX

5.1 RESPONSE CODE

Response Code	Description
00	Success or processed.
06	unknown error / failed
07	Invalid Gateway
14	insuficient amount
34	Service temporary unavilable
42	invalid msisdn
43	invalid sms content
44	senderid closed
66	Maintenance in progress
76	Wrong Password
77	userid not found
78	Data not found!
79	Invalid token
80	Expired token
81	maximum try reached!
88	missing parameter
96	apikey not found or non active
97	invalid json format
98	Authorization failed
99	wrong method

5.2 SIP Codes:
We consider dial status 180,183 & 2xx to be succeed result (misscall delivered).

Code	Description	Code	Description
1XX	**Provisional**	428	Use Identity Header
100	Trying	429	Provide Referrer identity
180	Ringing	430	Flow Failed
181	Call is being forwarded	433	Anonymity Disallowed
182	Queued	436	Bad Identity-Info
183	Session in progress	437	Unsupported Certificate
199	Early dialog	438	Invalid Identity Header
2XX	Successful	439	First Hop Lacks Outbound Support
200	Ok	470	Consent Needed
201	Accepted	480	Temporary Unavailable
204	No Notification	481	Call/Transaction Does Not Exist
3XX	Redirection	482	Loop Detected
300	Multiple choices	483	Too Many hops
301	Moved permanently	484	Address Incomplete
302	Moved temporary	485	Ambiguous
305	Use proxy	486	Busy Here
4XX	Client Failure	487	Request Terminated
400	Bad Request	488	Not Acceptable here
401	Unauthorized	489	Bad Event
402	Payment Required	491	Request Pending
403	Forbidden	493	Undecipherable
404	Not Found	494	Security Agreement Required
405	Method Not Allowed	5xxx	Server Failure
406	Not Acceptable	500	Internal Server error
407	Proxy Authentication Required	501	Not Implemented
408	Request Timeout	502	Bad Gateway
409	Conflict	503	Service Unavailable
410	Gone	504	Server Time-out
411	Length Required	505	Version Not Supported
412	Conditional Request Failed	513	Message Too Large
413	Request Entity too Large	580	Precondition Failure
414	Request-URI Too Long	6xx	Global Failure
415	Unsupported Media Type	600	Busy Everywhere
416	Unsupported URI Scheme	603	Decline
417	Unknown Resource-Priority	604	Does Not Exist Anywhere
420	Bad Extension	606	Not Acceptable
421	Extension Required		
422	Session Interval Too Small		
423	Internal Too Brief		
424	Bad Location Information		