Tutorial de consumo da API Valida Local

Claro

Para iniciar o consumo da API Valida Local, é necessário adquirir o token de acesso, através da key e da secret, da aplicação contratada no portal.

» Usar o postman para obter o token da API Valida Local

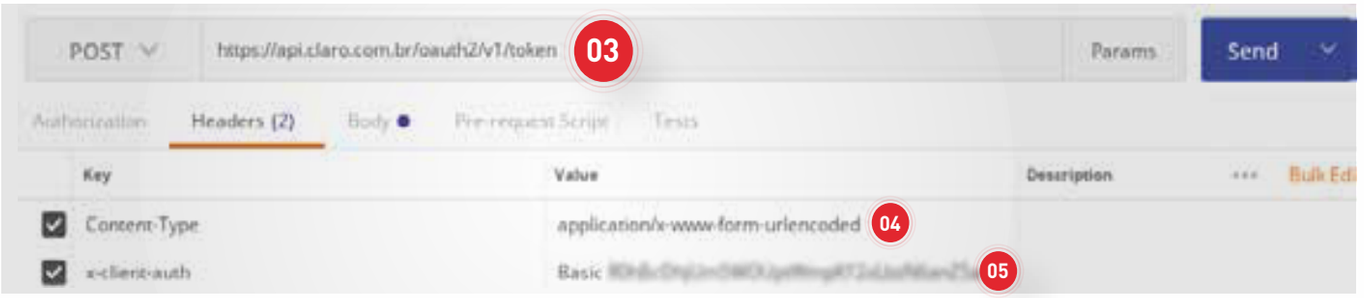
01 Inicie o postman.

02 Selecione o método POST.

03 Para o URI, insira https://api.claro.com.br/oauth2/v1/token

04 Na guia Headers, adicione chave de Content-Type e para o valor application/x-www-form-urlencoded.

05 Na guia Headers, adicione a chave de x-client-auth, seu valor será Basic base64(key:secret).

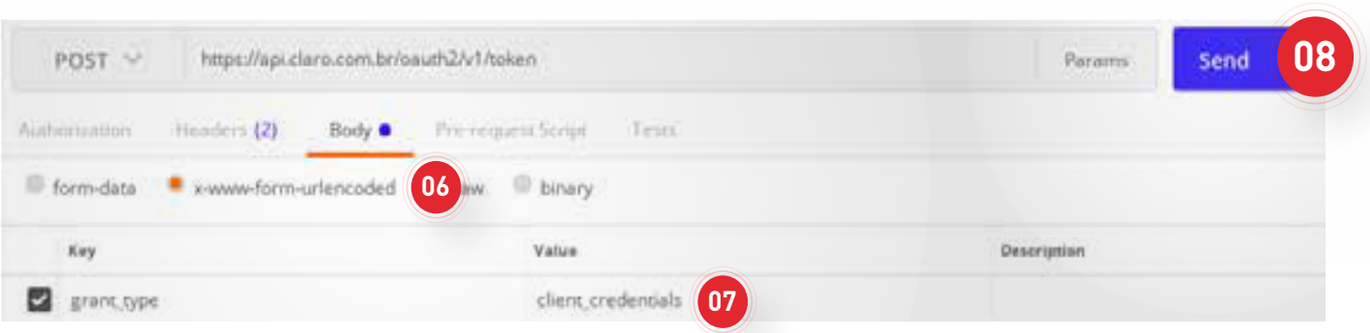


Observação, para gerar a base64 pode usar o site https://www.base64decode.org/, passando como entrada o key:secret, a saída deve ser algo como: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX==

06 Na guia Body, marque a caixa x-www-form-urlencoded

07 Na guia Body, adicione a chave de grant_type, seu valor será client_credentials

08 Envie a solicitação, pressionando o botão Envio



09 O token, tem 24 horas de duração, se não for solicitado novamente. Salve o access_token, para poder consumir a API.

```
{
    "refresh_token_expires_in": "0",
    "api_product_list": "[claro-validations-phones-pay-per-use, claro-score-pay-per-use, claro-alerta-pay-per-use, claro
        -validations-address-v1-pay-per-use]",
    "api_product_list_json": [
        "claro-validations-phones-pay-per-use",
        "claro-score-pay-per-use",
        "claro-alerta-pay-per-use",
        "claro-validations-address-v1-pay-per-use"
    ],
    "organization_name": "clarobrasil",
    "developer.email": "jefferson.barbosa@claro.com.br",
    "token_type": "BearerToken",
    "issued_at": "1663208186995",
    "client_id": "GHAp8cRnv9JmZjJclXo3zjvykmGv6F3p",
    "access_token": [redacted],
    "application_name": "1b606b2e-860f-42ca-a402-78038475c896",
    "scope": "",
    "expires_in": "86399",
    "refresh_count": "0",
    "status": "approved"
}
```


09

» Usar o postman para consumir API Validation Locations V1

01 Inicie uma nova aba no postman

02 Selecione o método GET

03 Para o URI, insira https://api.claro.com.br/customers/v1/validationslocations

04 Na guia Headers, adicione chave de Content-Type e para o valor application/json.

05 Na guia Headers, adicione chave de x-client-auth e para o valor Bearer access_token. Observação, substituía access_token pelo valor do atributo access_token, adquirido a solicitação do token.

06 Na guia Headers, adicione chave de X-CustomerID e para o valor coloque a razão social da sua empresa, sem espaço. Valor obrigátorio “QUOD”

07 Na guia Headers, adicione chave de guid e para o valor coloque identificador único da transação que identifica o regristo.

08 Na guia Headers, adicione chave de cpf e para o valor coloque o cpf que deseja validar, sem caracteres. Com 11 caracteres.

09 Na guia Headers, adicione chave de cep e para o valor coloque o cep que deseja validar, sem caracteres. Com até 8 caracteres.

10 Na guia Headers, adicione chave de number e para o valor o número do endereço.

11 Na guia Headers, adicione chave de phone e para o valor o número telefone do cliente, sem caracteres. Com 11 caracteres.

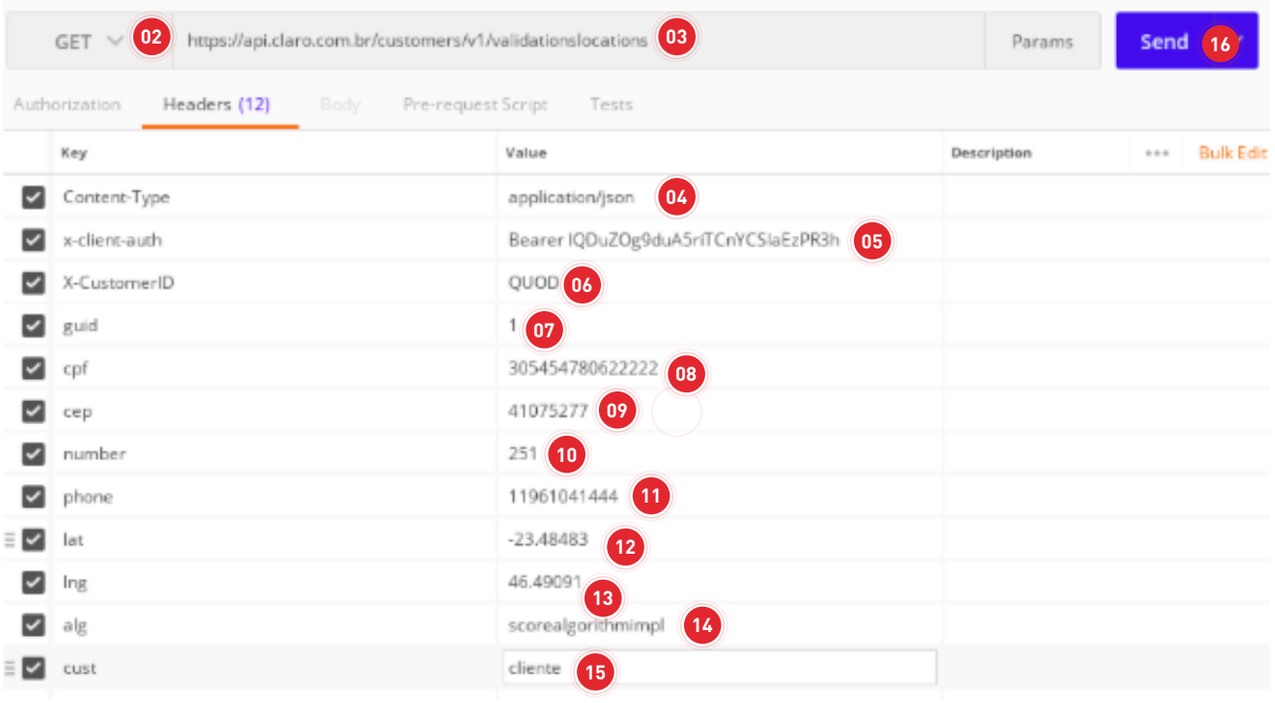
12 Na guia Headers, adicione chave de lat e para o valor a latitude do endereço que deseja validar, em double.

13 Na guia Headers, adicione chave de lng e para o valor a longitude do endereço que deseja validar, em double.

14 Na guia Headers, adicione chave de alg e para o valor o valor o algoritmo do validador, deve-se adotar o valor scorealgorithmimpl

15 Na guia Headers, adicione chave de cust e para o valor o nome do cliente que está validando

16 Envie a solicitação, pressionando o botão Envio



» Dados de request

Explicação dos dados do json

01 guid, valor único da transação que identificação o regristo. Integer, obrigátorio.

02 cpf , valor do CPF, sem caractere. String(11), obrigátorio.

03 cep, código do CEP, sem caractere. String(8), obrigátorio.

04 cep, código do CEP, sem caractere. String(8), obrigátorio.

05 phone, número da telefone, sem caractere. String(11), obrigátorio

06 lat, latitude do endereço. Double, obrigátorio.

07 lng, longitude do enredeço. Double, obrigátorio.

08 alg, nome do algoritmo, o valor deve ser scorealgorithmimpl. String, obrigátorio.

09 cust, nome do cliente. String, obrigátorio.

» Dados de response

Status com retorno com sucesso.

Exemplo de json de retorno.

```
{
  "data": {
    "guid": string,
    "cpf": int,
    "cep": int,
    "number": int,
    "phone": string,
    "lat": long,
    "lng": long,
    "customer": string,
    "algorithm": string,
    "score": int,
```

Explicação dos dados do json

01 guid, valor único da transação que identificação o regristo.
Integer, obrigátorio.

02 cpf , valor do CPF, sem caractere. String(11), obrigátorio.

```
    "flagScore": string,
    "flagFixa":  string
  }
{
```

03 cep, código do CEP, sem caractere. String(8), obrigátorio.

04 number, número da residência. String, obrigátorio. fornecido para o cliente encontrado

05 phone, número da telefone, sem caractere. String(11), obrigátorio.

06 lat, latitude do endereço. Double, obrigátorio.

07 lng, longitude do enredeço. Double, obrigátorio.

08 alg, nome do algoritmo, o valor deve ser scorealgorithmimpl. String, obrigátorio.

09 customer, nome do cliente. String, obrigátorio.

10 score, o score do resultado apresentado

Status com retorno com erro.

Exemplo de json de retorno.

```
    "apiVersion"
    "transactionId"
    "error"
        "detailedMessage"
        "errorCode"
        "message"
        "httpCode"
            "href"
            "rel"
        }
```

```
    }
}
```

Código de retorno 400

01 Campo apiVersion, obrigatório, tipo string, representa a versão da API.

Código de retorno 400

02 Campo transactionId, obrigatório, tipo string, representa o UUID da transação.

03 Campo errorCode, obrigatório, tipo string, representa o código de erro HTTP.

04 Campo errorCode, obrigatório, tipo string, representa o código de erro.

05 Campo message, obrigatório, tipo string, representa o Mensagem de erro.

06 Campo detailedMessage, obrigatório, tipo string, representa o Mensagem detalhada.

Código de retorno 400

07 Campo rel, obrigatório, tipo string, sempre “related”.

08 Campo href, obrigatório, tipo string, representa o endereço para o detalhamento dos erros.

» Status de response

Código de retorno 200 (Success)

Código de retorno 400 (Bad Request)

A solicitação não pôde ser entendida pelo servidor devido à sintaxe incorreta. O cliente NÃO DEVE repetir o pedido sem modificações.

Código de retorno 400 (Unalthorized)

A solicitação requer autenticação do usuário.

Código de retorno 403 (Forbidden)

O servidor entendeu a solicitação, mas está se recusando a atendê-la.

Código de retorno 404 (Not Found)

O servidor não encontrou nada que corresponda ao request-URI.

Código de retorno 405 (Not Found)

O método solicitado, não suportado pela API.

Código de retorno 415 (Unsupported Media Type)

O servidor falhou ao atender uma solicitação. Por que o formato do payload não é um formato suportado.

Código de retorno 406 (Not Acceptable)

Servidor não pode produzir uma resposta que combine com a lista de valores aceitáveis definidas.

» Exemplo de código em curl

```
!/bin/bash

url_validationsaddress=https://api.claro.com.br/customers/v2/
validationsaddress
url_token=https://api.claro.com.br/oauth2/v1/token

key=XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
secret=XXXXXXXXXXXXXX
guid="X"
cpf=XXXXXXXXXXX
customerID="QUOD"
cep=XXXXXXXX
number=XXX
phone="XXXXXXXXXXX"
lat=XXXXXXXXX
lng=XXXXXXXXX
alg="scorealgorithmimpl"
cust="XXXXXXXXX"

senha=$( echo -n $key:$secret | base64)
curl -X POST $url_token -H 'Content-Type: application/
x-www-form-urlencoded' -H
'cache-control: no-cache' -d 'grant_type=client_credentials'
-H "x-client-auth: Basic
$senha" > json

grep 'access_token' json > atributoJson
token=$(cut -d'"' -f4 atributoJson )
curl -X GET $url_validationsaddress -H "Content-Type:
application/json" -H "X-
CustomerID: $customerID" -H "x-client-auth: Bearer $token"
-H "guid: $guid" -H "cpf:
$cpf" -H "cep: $cep" -H "number: $number" -H "phone: $phone"
-H "lat: $lat" -H "lng:
$lng" -H "alg: $alg" -H "cust: $cust"
```

» Exemplo de código em python

```
import requests, json
import base64

KEY="XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
SECRET="XXXXXXXXXXX"

SENHA = base64.b64encode( (KEY + ':' + SECRET ).
encode('ascii') ).decode('ascii')
GRANT_TYPE = "client_credentials";

URL = "https://api.claro.com.br";
END_POINT_ACCESS_TOKEN = "/oauth2/v1/token";
END_POINT_API_VALIDATIONSADDRESS =
"/customers/v1/validationslocations";

headers={'Content-Type':
'application/x-www-form-urlencoded','cache-control':'no-
cache','x-client-auth': 'Basic {}'.format(SENHA)}

data = {'grant_type': 'client_credentials'}

respToken = requests.post(URL + END_POINT_ACCESS_TOKEN ,
headers=headers , data= data)

if respToken.status_code != 200:
    print('Erro:' + str( respToken.status_code))
else:
    token = json.loads( respToken.text)['access_token']

    guid="X"
    cpf=XXXXXXXXXXX
    customerID="QUOD"
    cep=XXXXXXXX

    number=XXX
    phone="XXXXXXXXXXX"
    lat=XXXXXXXXX
    lng=XXXXXXXXX
    alg="scorealgorithmimpl"
```

```
    cust="XXXXXXXX"

    headers_app = {'Content-Type': 'application/json',
    'X-CustomerID': 'claro_exemplo',
    'guid':guid,
    'cpf':str(cpf),
    'customerID':customerID,
    'cep':str(cep),
    'number':str(number),
    'phone':phone,
    'lat':str(lat),
    'lng':str(lng),
    'alg':alg,
    'cust':cust,
    'cache-control':'no-cache',
    'user-agent':'curl/7.60.0',
    'x-client-auth': 'Bearer {}'.format(token)}

    resp = requests.get(
  URL + END_POINT_API_VALIDATIONSADDRESS,
    headers=headers_app )

    if resp.status_code != 200:
        print('Erro: ' + str(resp.text))
    else:
        print('Valida ' + str( resp.text ))
```

» Exemplo de código em java

```
package client.validationsLocations;

import java.io.IOException;
import java.net.URI;
import java.net.URISyntaxException;
import java.util.Base64;

import org.codehaus.jackson.JsonProcessingException;
```

```
import org.codehaus.jackson.map.ObjectMapper;
import org.springframework.http.HttpEntity;
import org.springframework.http.HttpHeaders;
import org.springframework.http.HttpMethod;
import org.springframework.http.MediaType;
import org.springframework.http.ResponseEntity;
import org.springframework.util.LinkedMultiValueMap;
import org.springframework.util.MultiValueMap;
import org.springframework.web.client.RestClientException;
import org.springframework.web.client.RestTemplate;

public class App {

   private static final String KEY =
"XXXXXXXXXXXXXXXXXXXXXXXXXXX";

   private static final String SECRET = "XXXXXXXXXXX";

   private static final String GRANT_TYPE = "client_credentials";

   private static final String URL = "https://api.claro.com.br";

   private static final String END_POINT_ACCESS_TOKEN =
"/oauth2/v1/token";

   private static final String
END_POINT_API_VALIDATIONS_LOCATIONS =
   "/customers/v1/validationslocations";

   public static void main( String[] args ) throws URISyntaxException,
JsonProcessingException, IOException {

      App app = new App();

      String token = app.getAccessToken();

      String guid="X";
      String cpf= "XXXXXXXXXXX";
      String customerID="QUOD";
      String cep = "XXXXXXXX";
      String number="XXX";
      String phone="XXXXXXXXXXX";
      String lat="XXXXXXXXX";
      String lng="XXXXXXXXX";
      String alg="scorealgorithmimpl";
      String cust="XXXXXXXXXX";
```

```
        System.out.println("Válida Localização: "
 + app.getValidationsLocations( token , guid , cpf ,
customerID,cep,number,phone,lat,lng,alg,cust ));
    }

    public String getValidationsLocations(String acessToken,
String guid , String cpf , String customerID, String cep,
String number,String phone,String lat, String lng, String alg,
String cust) throws RestClientException, URISyntaxException {

        HttpHeaders headers = new HttpHeaders();
        headers.setContentType(MediaType.APPLICATION_JSON);
        headers.add("cache-control","no-cache");
        headers.add("guid",guid);
        headers.add("cpf",cpf);
        headers.add("number",number);
        headers.add("cep",cep);
        headers.add("phone",phone);
        headers.add("lat",lat);
        headers.add("lng",lng);
        headers.add("alg",alg);
        headers.add("cust",cust);
        headers.add("x-client-auth", "Bearer " + acessToken);
        headers.add("X-CustomerID", customerID);

        HttpEntity entity = new HttpEntity(headers);

        RestTemplate restTemplate = new RestTemplate();
        ResponseEntity<String> result =
restTemplate.exchange(new URI(URL +
        END_POINT_API_VALIDATIONS_LOCATIONS),
HttpMethod.GET, entity, String.class);

        return result.getBody();
    }

    public String getAccessToken() throws URISyntaxException,
JsonProcessingException, IOException {
        String senha = Base64.getEncoder().encodeToString(
        App.KEY.concat(":").concat(App.SECRET).getBytes() );

        HttpHeaders headers = new HttpHeaders();
        headers.setContentType
(MediaType.APPLICATION_FORM_URLENCODED);
        headers.add("cache-control","no-cache");
```

```
        headers.add("x-client-auth", "Basic " + senha);

        MultiValueMap<String, String> parameters =
  new LinkedMultiValueMap<>();
        parameters.add("grant_type", GRANT_TYPE);

        HttpEntity<MultiValueMap<String, String>>
  request = new HttpEntity<>(parameters,
        headers);

        RestTemplate restTemplate = new RestTemplate();
        ResponseEntity<String> result =
  restTemplate.postForEntity(new URI(URL +
        END_POINT_ACCESS_TOKEN),
        request, String.class);

        return new ObjectMapper().readTree(result.getBody()).get
  ("access_token").asText() ;
    }
  }
```

» Exemplo de código em php

```
$key = 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX';

  $secret = 'XXXXXXXXXXXXXXX';

  $guid="X";

  $cpf=XXXXXXXXXXX;
  $cep=XXXXXXXX;
  $number=XXX;
  $phone="XXXXXXXXXXX";
  $lat=XXXXXXXXX;
```

```
    $lng=XXXXXXXXX;
    $alg="scorealgorithmimpl";
    $cust="XXXXXXXXXX";

    $token = getAccessToken();

    $validationsLocations =
getValidationsLocations($token , $guid, $cpf , $cep,
    $number, $phone, $lat, $lng, $alg, $cust );

    echo $validationsLocations;

    function getAccessToken() {

        try {

            $senha = base64_encode($GLOBALS['key'] . ':' .
$GLOBALS['secret']);

            $ch = curl_init();
            curl_setopt($ch, CURLOPT_URL,
"https://api.claro.com.br/oauth2/v1/token");
            curl_setopt( $ch, CURLOPT_HTTPHEADER, [
            'Content-Type: application/x-www-form-urlencoded',
            'x-client-auth: Basic ' . $senha

            ]);

            curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
            curl_setopt($ch, CURLINFO_HEADER_OUT, true);
            curl_setopt($ch, CURLOPT_POST, true);
            curl_setopt($ch, CURLOPT_POSTFIELDS,
'grant_type=client_credentials');

            $response = curl_exec($ch);

            if (!$response){
                $error = curl_error($ch);
                $info = curl_getinfo($ch);
                die("cURL request failed,
error = {$error}; info = " . print_r($info, true));
            }

            if (curl_errno($ch)){
                curl_close($ch);
                echo 'error:' . curl_error($ch);
```

```
        } else {
            curl_close($ch);
            return json_decode($response)->access_token;
        }

    } catch (Exception $e) {
        return 'Erro '. $e;
    }
  }

  function getValidationsLocations($token ,
$guid, $cpf , $cep, $number, $phone, $lat, $lng, $alg, $cust ) {

    try {
    echo $token;

        $ch = curl_init();
        curl_setopt($ch, CURLOPT_URL,
        "https://api.claro.com.br/customers/v1/validationslocations");

        curl_setopt( $ch, CURLOPT_HTTPHEADER, [
            'Content-Type: application/json',
            'guid: ' . $guid,
            'cpf: ' . $cpf,
            'cep: ' . $cep,
            'number: ' . $number,
            'phone: ' . $phone,
            'lat: ' . $lat,
            'lng: ' . $lng,
            'alg: ' . $alg,
            'cust: ' . $cust,
            'X-CustomerID: QUOD',
            'x-client-auth: Bearer ' . $token
        ]);

        curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
        curl_setopt($ch, CURLINFO_HEADER_OUT, true);
        curl_setopt($ch, CURLOPT_HTTPGET, true);

        $response = curl_exec($ch);

        if (!$response){
            $error = curl_error($ch);
```

```
            $info = curl_getinfo($ch);
            die("cURL request failed, error =
{$error}; info = " . print_r($info, true));
        }

        if (curl_errno($ch)) {
            curl_close($ch);
            echo 'error:' . curl_error($ch);
        } else {
            curl_close($ch);
            return $response;
        }
    } catch (Exception $e) {
        return 'Erro '. $e;
    }
}
```

Claro