

# Citcon UPI WebSDK

Version 0.2.1

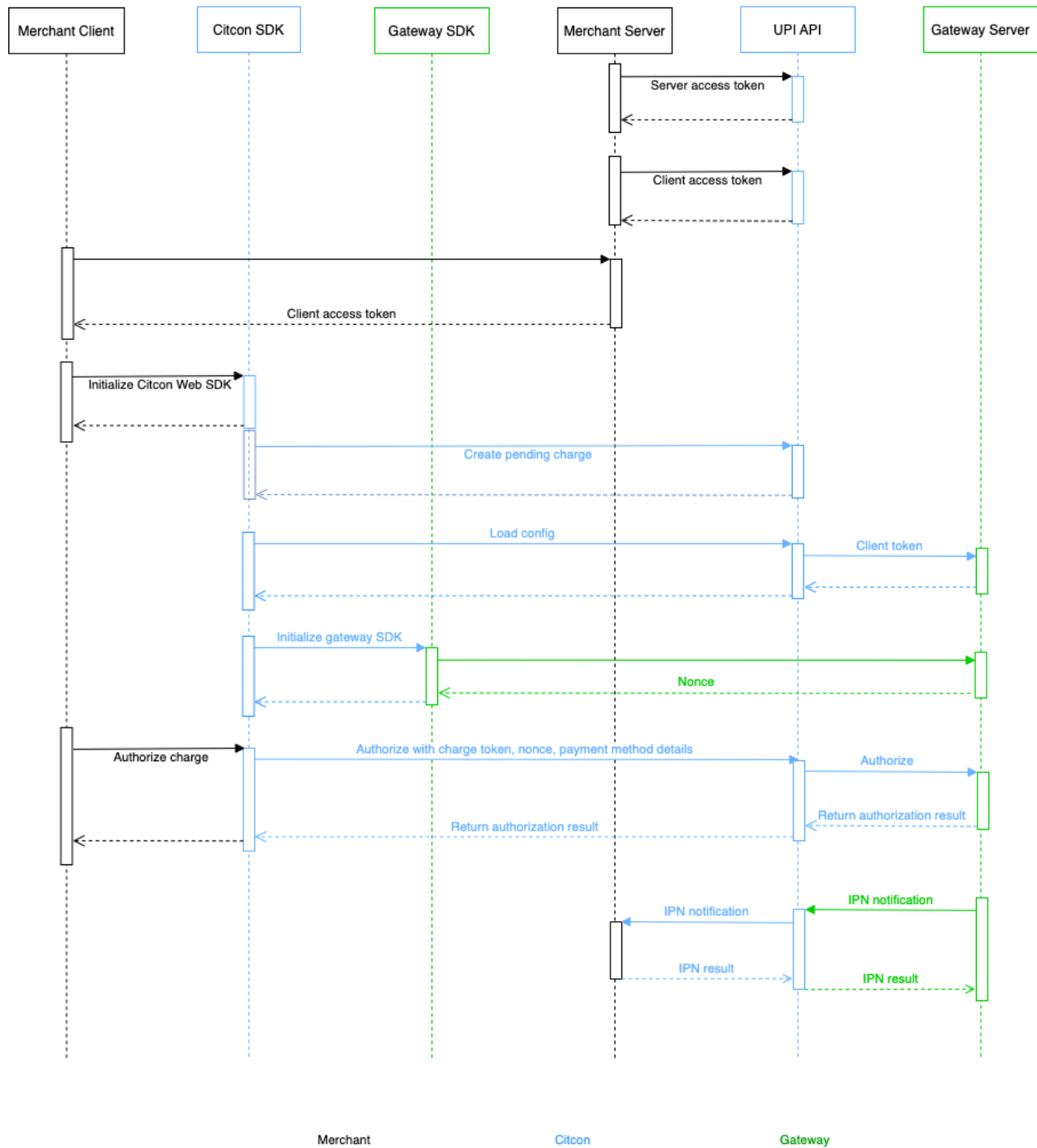
Version No.	Modify Activity	Modify Description	Editor	Modify Date
0.1.0	Creation	Paypal & Venmo Support	Long Zhou	2022-05-04
0.2.0	Add Parameter	Add Parameter	Long Zhou	2022-08-04
0.2.1	Add autoCapture	Add autoCapture	Long Zhou	2022-08-08

## 1. Introduction

Citcon UPI SDK was designed for online merchants to integrate Citcon payment solutions effortlessly into their own app. By using the SDK, merchant developers can focus on business logic without having to understand the plumbing of payment transactions. This version of the SDK supports PayPal , Venmo.

## 2. Payment Flow

### Payment Initiated from Web SDK



### 3. Integration Steps

As an example, this guide uses jQuery framework for frontend and PHP /JAVA for backend. You can use your own programming language to do your project

Please Note:

Citcon UPI has two kinds of access token, one is client access token, which has inquiry and charge permissions. This is the correct access token for WebSDK / Mobile SDK use.

The other one is server access token, which has full permissions: inquiry, charge, refund, capture, vault.

In Citcon WebSDK. Only client access token is needed.

#### Step 1: Load Core JS in html page

Load Web SDK core at the bottom of the html page, for example before `</body>` or after `</html>`. You can either use the latest version or specify a version

UAT

```
<script  
src="https://cdn.uat01.citconpay.com/latest/core/citconpay.core.js"></script>
```

Version base


```
<script  
src="https://cdn.uat01.citconpay.com/v0.2.3/core/citconpay.core.js"></scri  
pt>
```

## Production

```
<script  
src="https://cdn.citconpay.com/latest/core/citconpay.core.js"></script>
```

## Version base

```
<script  
src="https://cdn.citconpay.com/v0.2.3/core/citconpay.core.js"></script>
```

For version history, please refer:  [upi\\_websdk\\_demo/history.md at main · long2934/upi\\_websdk\\_demo](#)

## Step 2: Create access token and pending charge from server back end

### Frontend Sample Code:

```
$.ajax({  
  url: merchantUrl + '?action=create_transaction',  
  type: 'post',  
  dataType: 'json',  
  data: JSON.stringify({  
    reference: transaction_reference,  
    totalAmount: parseInt($("#txtAmount").val()),  
    currency: $("#currency").val(),  
    countryCode: $("#country").val()  
  }),  
});
```

```

success:function(resp){
  console.log('create pending transaction...' + JSON.stringify(resp));
  if(resp.status === 'success'){
    access_token = resp.data.access_token;
    chargeToken = resp.data.charge_token;
    transactionId = resp.data.transaction_id;
  }else{
    console.log(resp.data);
  }
},
async:false
});

```

This only PHP / JAVA sample code, you may need to implement by your own

For the back end, the only thing you need to do is create a http function called Citcon UPI API to create the pending charge, you may need a merchant key which you get from Citcon.

**UPI API url**

Sandbox: <https://api.sandbox.citconpay.com/v1>

Production: <https://api.citconpay.com/v1>

**Step 1 is create an access token from merchant key**

PHP code sample

```

function get_access_token(){
  $data = array(
    "token_type" => "client"
  );
  $url = API_URL . '/access-tokens';

```

//xxxx is your merchant private key which get from citcon

```

    $resp =
do_http_post_with_token($url, 'xxxx', json_encode($data));

    return $resp;
}

```

Java code sample

```

public AccessTokenResponse getAccessToken(){

    String url = Const.getUPIUrl(usingSandbox) +
"/access-tokens";
    String data = "{\n  \"token_type\": \"client\"}";
    String resp = callUPIAPI(url,merchantKey,data);
    AccessTokenResponse accessTokenResponse = new
Gson().fromJson (resp,AccessTokenResponse.class);

    return accessTokenResponse;
}

```

Here the token type is “client”. There are two kinds of token types: client and server. The difference is permission. client usually use for WebSDK & MobileSDK

The Response body for this step, will like this sample

```

{
  "status": "success",
  "app": "citcon_upi",
  "version": "v0.1.1",
  "data": {
    "access_token":
"UPI_eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1IjoiYnJhaW50cm
VlIiwiaWF0IjoxNjU4NTc1NzM0LCJleHAiOjE2NjAzMjA3MDkyMjF9.mM0cVF5_-
I8UGfkYhaW39mQjsU1SwKhYSMGs8RYGdkY",

```

```

        "token_type": "client",
        "expiry": 1658662133487,
        "permission": [
            "inquiry",
            "charge",
            "vault",
            "installment"
        ]
    }
}

```

**Step 2 is create the pending charge from the access token**

PHP code sample

```

function create_pending_transaction($token){

    $json = file_get_contents('php://input');
    $json_data = json_decode($json,true);
    $urls = array (
        "ipn_url" => "https://dev.citconpay.com",
        "success_url" => "https://dev.citconpay.com",
        "fail_url" => "https://dev.citconpay.com",
        "mobile_url" => "https://dev.citconpay.com"
    );
    $transaction = array(
        "reference" => $json_data['reference'],
        "amount"=> $json_data['totalAmount'],
        "currency"=> $json_data['currency'],
        "auto_capture"=> false,

```



```

        "country"=> $json_data['countryCode'],
        "urls"=> $urls
    );
    $data_array['transaction'] = $transaction;
    $data = json_encode($data_array);
    $url = API_URL . '/charges';
    $resp = do_http_post_with_token($url,$token,$data);
    return $resp;
}

```

#### Java code sample

```

public PendingChargeData getPendingCharge(String
reference,String amount,String currency,String
countryCode,String token){
    String data = "{";
    data = data + "\"transaction\": {";
    data = data + "\"reference\": \""+reference+"\", ";
    data = data + "\"amount\": "+ amount+", ";
    data = data + "\"currency\": \""+currency+"\", ";
    data = data + "\"country\": \"" + countryCode + "\" , ";
    data = data + "\"auto_capture\": false, ";
    data = data + "\"urls\": {";
    data = data + "\"ipn\": \"http://ipn.com\", ";
    data = data + "\"success\": \"http://success.com\", ";
    data = data + "\"fail\": \"http://fail.com\", ";
    data = data + "\"mobile\": \"http://mobile.com\", ";
    data = data + "\"cancel\": \"http://cancel.com\"";
    data = data + "}";
    data = data + "}";
    data = data + "}";
    System.out.println(data);
    String url = Const.getUPIUrl(usingSandbox) + "/charges";
    String resp = callUPIAPI(url,token,data);
}

```

```

        PendingChargeResponse pendingChargeResponse = new
Gson().fromJson(resp, PendingChargeResponse.class);
        return pendingChargeResponse.getData();
    }

```

Here the IPN (Instant Payment Notify) url is the url which you want to get the result from Citcon UPI.

This Step response body will looks like

```

{
  "status": "success",
  "data": {
    "charge_token": "5412428009ca11edbeec5dd5c069f366",
    "transaction_id": "4000033920223962677253",

    "access_token": "UPI_eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1Ijo1YnJhaW50cmVlIiwiaWF0IjoxNjU4NTAwMDEwLCJleHAiOjE2NjAyNDQ5MTA0MDB9.gIF-jo1_-KUY43UeHPtN5wFQDqyMks2-LrSC5TKtZaI"
  }
}

```

for fully demo code, please download from here [🔗 GitHub - long2934/upi-websdk\\_demo: Demo for UPI WebSDK](#)

Step 3: Initialize WebSDK Core with access token

Add a element in the HTML file to host WebSDK widgets, such as

```
<div id="citcon-client-container"></div>
```

Web SDK then can mount a widget.

### Sample code

```
// the payment method is what the merchant chose at onboarding.
let paymentMethodArray = ['paypal', 'venmo'];
let defaultPaymentMethod = 'paypal';

const configObj = {
  accessToken: access_token,
  environment: 'uat', ///uat/prod, This is optional. default is production
  debug: true, // true/false, This is optional. default is false
  consumerID: "18000"
};

citconpay.client.core(configObj).then( clientInstance=>{
  console.log(' Init SDK...' + JSON.stringify(clientInstance));
  // mount UI
  // class name is css class, you can define your own class at your style sheet
  citconInstance = clientInstance;
  clientInstance.mount('#citcon-client-container',{
    classname: 'payment-method-select-component',
    paymentMethods: paymentMethodArray,
    selectedPaymentMethod: defaultPaymentMethod,
  }, sdkUIDidInitialized).then(function(instance) {

    console.log('instance ...' + JSON.stringify(instance));
    //Do Events Register after WebSDK core init complete
    registerEvents();
  }).catch(error=>{
    console.log(' mount error:' + JSON.stringify(error));
  });
}).catch(error=>{
  console.log(' Init SDK error:' + JSON.stringify(error));
});
function sdkUIDidInitialized(e){

  console.log(' sdk ui initialized ..... ' + JSON.stringify(e));
}
```

## Step 4: Register event listeners

Supported events include the following:

### 1.payment-method-selected

This event is triggered when consumer clicks on the Radio Button



☐ **Payment Option**  
**Paypal**

☒ **Payment Option**  
**Venmo**





### 2.payment-method-submitted

This event is triggered when consumer clicks the “Pay Now” button

### 3.payment-status-changed

This event is triggered when payment is completed. It calls back the listener with payment results.

Sample Return Object in JSON:

Success:

```
{
  "status": "success",
  "data": {
    "object": "charge",
    "id": "da949a90cb1011ecb9a3550ac676a90f",
    "reference": "0wqd3i0s5p8483q62f5x",
    "amount": 180,
    "amount_captured": null,
    "amount_refunded": null,
    "currency": "USD",
    "time_created": 1651603379000,
    "time_captured": null,
    "auto_capture": false,
    "status": "authorized",
    "country": "US",
    "payment": {
      "method": "paypal"
    }
  },
  ...
}
```

fail:

```
{
  "status": "fail",
  "data": {
    "code": "xxxx",
    "message": "xxxxx",
  }
}
```

```
}
```

### Sample Code:

```
function registerEvents(){

    citconInstance.on('payment-method-selected', function (e) {
        console.log('Inside `payment-method-select` ..... ' + JSON.stringify(e));
        //Do you code here
    });
    citconInstance.on('payment-method-submitted', function(e) {
        // UI Event
        // For those payment method Popup Or Redirect, Sucha as Paypal, Venmo
        // merchant handles selection of payment methods through Citcon's UI
        console.log('....paynow..click.....');
        $("body").addClass("loading");
        const requestOptions ={
            payment: {
                chargeToken:chargeToken,
                countryCode:$("#country").val(),
                transactionReference: transaction_reference
            },
            billing_address: {
                street: $("#address").val(),
                street2:$("#address2").val(),
                city: $("#txtCity").val(),
                state: $("#state").val(),
                zip: $("#zip").val(),
                country: $("#country").val()
            },
            consumer:{
                id:"18000",
                reference: "consumer_test_1",
                firstName: $("#firstName").val(),
                lastName: $("#lastName").val(),
```

```

        phone: $("#phone").val(),
        email: $("#email").val(),
    }
}

citconInstance.onPaymentMethodSubmitted(e.paymentMethod,requestOptions).then(rest=>{
    console.log('pay now click, return..' + JSON.stringify(rest));
}).catch(error=>{
    $("#body").removeClass("loading");
    console.log('pay now click, error..' + JSON.stringify(error));
});

});
//on payment status change, this is the event for charge result
//status: status, status will return "success" or "failed"
//retObj: retObj
citconInstance.on('payment-status-changed', function(e) {
    const status = e.status;
    const res = e.retObj;

    console.log('payment-status-change status..' + status + JSON.stringify(res));
    if(status == 'success'){
        //payment success
        // do your code here....
    }else{
        $("#body").removeClass("loading");
        if (res.code === 'VENMO_CANCELED') {
            console.log('App is not available or user aborted payment flow');
        } else if (res.code === 'VENMO_APP_CANCELED') {
            console.log('User canceled payment flow');
        } else {
            console.log('An error occurred:', res.message);
        }
    }
}
});
}

```

## Additional functions

## Charge inquiry

You can use this function to inquire charge status

Sample code

```
$("#body").addClass("loading");
console.log('inquire order... id=' + transactionId);
citconInstance.inquire(transactionId).then(resp=>{
    console.log('inquire...return' + JSON.stringify(resp));
    $("#body").removeClass("loading");
}).catch(error=>{
    console.log('inquire order error...' +
JSON.stringify(error));
    $("#body").removeClass("loading");
});
```

## Parameters

### Init SDK

Sample code:

```
//init sdk
const configObj = {
    accessToken: access_token,
    environment: 'prod', //dev/qa/uat/prod,
    debug:true,
    consumerID:"18000"
};

console.log(' citconpay...' + JSON.stringify(citconpay));
```



```
citconpay.client.core(configObj).then( clientInstance=>{  
...  
}
```

## Parameters

Parameter	Required	Description
accessToken	M	This is the access token which create by backend code at step 2, use to communication with UPI API
environment	O	This is a enum , values will be uat /prod, default is prod
debug	O	This is whether show log . suggest set to true during developing. and remove it when live. default is false
consumerID	M	This is the consumer ID which store in Merchant's database

## Submit Payment

This is for event payment-method-submitted.

Sample Code will be

```
citconInstance.on('payment-method-submitted', function(e) {
  console.log('....paynow..click....');
  $("body").addClass("loading");
  const paypalOptions = {
    payment: {
      totalAmount: parseInt( $("#txtAmount").val()),
      currency:$("#currency").val(),
      countryCode:$("#country").val(),
      transactionReference: transaction_reference,
      chargeToken:chargeToken,
      autoCapture: true
    },
    billing_address: {
      street: $("#address").val(),
      street2:$("#address2").val(),
      city: $("#txtCity").val(),
      state: $("#state").val(),
      zip: $("#zip").val(),
      country: $("#country").val()
    },
    consumer:{
      id:"18000",
      reference: "consumer_test_1",
      firstName: $("#firstName").val(),
      lastName: $("#lastName").val(),
      phone: $("#phone").val(),
      email: $("#email").val(),
    }
  }
  switch(e.paymentMethod){
    case 'venmo':
    case 'paypal':

citconInstance.onPaymentMethodSubmitted(e.paymentMethod,paypalOptions).then(rest=>{
  console.log('pay now click, return..' + JSON.stringify(rest));
}).catch(error=>{
  $("body").removeClass("loading");
  console.log('pay now click, error..' + JSON.stringify(error));
```

```
});  
break;  
}  
  
});
```

Parameters

Parameter	Required	Description
payment.totalAmount	M	Total amount of the order, In Cents, such as 100 is \$1.00
payment.currency	M	Currency Code, will be USD, CAD
payment.countryCode	M	Country Code, This is ISO2, like US, CA
payment.transactionReference	M	This is the transaction reference
payment.chargeToken	M	This is the charge token which create by backend at step 2

payment.autoCapture	O	Whether need auto capture
billing_address.street	O	Billing address street address
billing_address.street2	O	Billing address street address2, like Apartment number,..
billing_address.city	O	Billing address city name
billing_address.state	O	Billing address state or province name
billing_address.zip	O	Billing address zip or post code
billing_address.country	O	Billing address country code, this also ISO2, ex. US.
consumer.id	M	This is the consumer ID which store in Merchant's database, the same as Init SDK

consumer.reference	M	This is the consumer reference
consumer.firstName	O	This is the consumer first name
consumer.lastName	O	This is the consumer last name
consumer.phone	O	This is the consumer phone number
consumer.email	O	This is the consumer email address