SimplePay 2.0 (API v2 auto interface)

Development Documentation

Payment process and development for credit/debit card data management for merchants

05.11.2025

Table of Contents

Document History

Date	Version	Change
12.11.2018	181112	Original issue
12.03.2019	190312	3DS extension
18.08.2020	200818	3DS 2.x supplement
25.03.2021	210325	Updating JSON sample codes
01.07.2025	250701	Corporate branding updates
05.11.2025	251105	AutoPayment endpoints addition, SZÉP card Transaction ID length

1 Short Summary

1.1 SimplePay Online Payment v2

SimplePay is the online payment solution of SimplePay Plc. This documentation was created for the v2 version of the merchant API and the transaction management to be used for SimplePay payments.
It only focuses on payment transaction scenarios in which customers' card details are prompted on the merchant's website.

This documentation only provides information for the above-described special management of card data, but basic know-how and the payment process is not discussed.

Prior knowledge of the information contained in the general documentation is therefore crucial in interpreting the forthcoming information.

The current version of SimplePay's general technical documentation is available in the download section of the merchant admin interface (for both the sandbox and the live systems), or at the following URL:

Hungarian version: https://simplepartner.hu/download.php?target=v21dochu
English version: https://simplepartner.hu/download.php?target=v21docen

This documentation refers to the aforementioned general documentation in all sections when discussing corresponding technical background information.

1.2 Conditions of card data management for merchants

In case of both the standard transaction described in the general documentation and the card registration payment, customers were redirected to SimplePay's secure payment page, where they could provide their card data. The card details were not provided directly on the merchant website; therefore, the merchant webstore was not involved in card data management.
This is a technically simpler process for the merchant, furthermore, the merchant is not burdened by the responsibility arising from the secure management of card data.

On SimplePay's secure payment page, card data is safeguarded through **PCI-DSS** (Payment Card Industry Data Security Standard) compliance.

SimplePay's system is capable of receiving card data from the merchant, therefore enables merchants to manage customer card details. In this case, merchants prompt customers for their card data on the merchant website, then initiate the transaction using SimplePay's specific interface (**auto**). In such cases no redirection takes place, communication is carried out towards the SimplePay system in the background.

However, compared to the previously described process, further requirements have to be met by the merchant in order to use the interface:

The merchant system must achieve audited PCI-DSS compliance, too.
Please don't develop this function if your system does not meet these requirements.

1.3 Payment transaction process

1.3.1 Debit/credit card payment with card data prompt

a. On the merchant's website the customer selects the items to be purchased and the total sum to be paid is calculated.
b. The merchant prompts for the customer's card details.
c. The merchant transfers the transaction data and the card details to SimplePay via the API (**auto**). Upon the API request, the payment is immediately verified by the bank (authorisation). Meanwhile, the customer stays on the webpage of the webstore, no redirection is necessary.
d. The SimplePay system sends a synchronous response to the **auto** request with the result of the authorisation. This step is the logical equivalent of the redirection back to the merchant's website in case of regular payments (**back**).
e. After this, the fraud monitoring and prevention process is carried out in the background. If the system detects no issues, it sends feedback to the website in the background (**ipn**). This concludes the payment transaction. After the IPN message is received, the merchant can complete the order.

1.4 Downloadable Resources

The basic resources necessary for the development can be downloaded from the merchant administration interface of both the sandbox and the live systems.

In both systems, the following content can be found in the left side menu, under "**Downloads**"

- technical documentation
- sample code
- logo package
- financial resources

Documentation and sample code for payments with card data management (**auto**) are available at the following URL:

This documentation in Hungarian:
https://simplepartner.hu/download.php?target=v21autodochu

This documentation in English:
https://simplepartner.hu/download.php?target=v21autodocen

SDK sample code:
https://simplepartner.hu/download.php?target=v21autosdk

The general SimplePay technical documentation in Hungarian:
https://simplepartner.hu/download.php?target=v21dochu

The general SimplePay technical documentation in English:
https://simplepartner.hu/download.php?target=v21docen

General SDK sample code
https://simplepartner.hu/download.php?target=v21sdk

2 Implementation

The **auto** request is expected by the API at the following URL:
Sandbox system: https://sandbox.simplepay.hu/pay/pay/auto/pspHU
Production system: https://securepay.simplepay.hu/pay/pay/auto/pspHU

The transaction data must be sent by POST method to the above URL in a JSON string described in the referenced documentation's "**General message format**" chapter.

Request authentication is discussed in the referenced documentation's "**Validating messages**" chapter.

2.1 auto – transaction start with card data without 3DS

Compared to the start function used for initiating standard payments, the main difference is the presence of the **cardData** field, which holds the card details.

cardData must include the following:
number: the card number, excluding spaces or any other separators
expiry: the card expiry date, excluding spaces or any other separators. The first two digits indicate the month of expiry, the third and fourth digits indicate the year of expiry. If the month value consists only of a single digit, like March (3), the value must be padded with a leading zero, e.g. to 03, in the case of March.
cvc: card CVC/CVV code
holder: cardholder's name

The following card details can be used for sandbox tests:
number: 4908366099900425
expiry: 1021
cvc: 579
holder: v2 AUTO Tester

Starting a call with minimum data, **without 3DS**.

```
{
  "merchant":"PUBLICTESTHUF",
  "orderRef":"101010515417720487444",
  "customerEmail":"sdk_test@simplepay.com",
  "currency":"HUF",
  "total":"100",
  "sdkVersion":"SimplePay_PHP_SDK_2.0.6_190301",
  "salt":"d79ed6c2d9aa3a7f96a9f884f0ee3958",
  "cardData":{
    "number":"4908366099900425",
    "expiry":"1021",
    "cvc":"579",
    "holder":"v2 AUTO Tester"
  },
  "customer":"v2 AUTO Tester",
  "language":"HU",
}
```

The payment system initiates the authorisation for the specified card. The API sends synchronous response with the authorisation result. The response's "**result"** value is 0 if the transaction is successful. All other values indicate a failed authorisation, and the card is not charged.

Response to the request

```
{
   "total":"100.0",
   "salt":"Kf2c7l6t3IJ9H0p90tR2T517x5T7T3Tm",
   "orderRef":"101010515417720487444",
   "merchant":"PUBLICTESTHUF",
   "currency":"HUF",
   "transactionId":"99355853"
}
```

Should the authorisation be successful, the IPN message is sent as described in the referenced documentation.

2.2 New variables linked to 3DS

Strong customer authorisation to be launched on 30 September 2020 requires the transmission of extended transaction data. This is what the variables to be discussed below enable.

For more information on strong customer authentication see Annex 2 to the basic documentation.

2.2.1 threeDSReqAuthMethod

The way buyers are registered in the merchant's system:

possible values:

01: guest
02: registered with the merchant
05: registered with a third-party ID (Google, Facebook, account, etc.)

2.2.2 type

The presence or absence of the buyer needs to be indicated during the transaction. Then it needs to be specified whether the payment is a one-off or a recurring transaction.

A one-off payment may be made in the user's presence and with their active participation (**CIT**) or without these (**MIT**) in the following three ways.
All recurring payments (**REC**) are started by merchants in the absence of the users concerned.

When the buyer is present (Customer Initiated Transaction, **CIT**) and clicks on the button starting the payment process on the merchant's page, the merchant's system contacts the SimplePay system in the background where the payment transaction actually takes place.
The advantage of this is that the buyer will not be redirected to the SimplePay payment page and will not have to provide any additional information, but the payment can be made with only one click.

When the buyer is not present (Merchant Initiated Transaction, **MIT**), payment can be started by the merchant with the card stored in the merchant's system. One example of this is when multiple micro-transactions take place in which actual payment is not started instantaneously: instead, the merchant's system starts payment – of an aggregated amount – once a day with SimplePay.

When the buyer is not present at the time the transaction is started **and** the merchant **debits the card** over a period **at regular intervals** (Recurring, **REC**).
The emphasis here is on regularity, on debiting at equal intervals. Intervals can be monthly, weekly, daily, etc., according to the merchant's business model, as in the case of the collection of regular monthly instalments or monthly tariffs.

When the "**auto**" endpoint is contacted the SimplePay system has no information on which of the above three types the buyer represents on the merchant's page from which the transaction is started; therefore, it notifies this to the card issuer bank solely on the basis of the **type** variable communicated to it.

2.2.3 browser

The parameters of the buyer's browser in the case of transactions started in the buyer's presence (**browser**). This piece of data is also part of **3DS** authentication, which is forwarded to the card-issuing bank.

details of the **browser** array

- **accept**: Value of accept http header
- **agent**: Value of user-Agent http header
- **ip**: IP of browser source
- **java**: whether the browser supports java applets; in javascript: *navigator.javaEnabled()*
- **lang**: browser language; in javascript: *navigator.language*
- **color**: color depth of browser; in javascript: *screen.colorDepth*
- **height** = browser screen height; in javascript: *screen.height*
- **width** = browser screen width; in javascript: *screen.width*
- **tz** = browser time zone; in javascript: *new Date().getTimezoneOffset()*

When the cardholder is present at the transaction (**CIT**) therefore the parameters of their browser need to be handed over. If browser data are lacking, 3DS authentication by the bank may end with an error.

2.2.4 url + challenge process

In the case of a **CIT** started in the buyer's presence the merchant's system needs to be prepared to respond in case during the 3DS process the card-issuing bank may require the cardholders to identify themselves.

In this case, the SimplePay system specifies an URL (**redirectUrl**) to the merchant in the "**auto**" call response to which it can redirect the buyer (**challenge**).

If the merchant's system redirects the customer to the URL received in the **"redirectUrl"** field, the customer is at a place where additional customer authentication required by the card-issuing bank is performed.

Accordingly, the **challenge** occurs in the browser already. At the end of the process, the system redirects the buyer in the browser to the merchant's website.
It is for this redirection that the SimplePay system needs to know where the merchant's system wants to receive the buyer.
The URL can be entered when the call is made, in the "**url**" variable, or in the "**urls**" array, in accordance with the general documentation's description.

2.3 auto – transaction started with 3DS check on the SimplePay side

2.3.1 Starting transaction in the buyer's presence


```
{
   "salt":"370b5d7f71884cc43783405157d53deb",
   "merchant":"LRMIPSMHUF",
   "orderRef":"101010515976555937097",
   "currency":"HUF",
   "customerEmail":"sdk_test@simplepay.com",
   "sdkVersion":"SimplePay_PHP_SDK_2.0.9_200130:dc597528c6fb3b4c6312d470ed28895e",
   "cardData":{
      "number":"4908366099900425",
      "expiry":"0122",
      "cvc":"123",
```

```
      "holder":"V2 AUTO test"
   },
   "type":"CIT",
   "threeDSReqAuthMethod":"02",
   "browser":{
      "accept":"text\/html,application\/xhtml+xml,application\/xml;q=0.9,image\/webp,image\/apng,*\/*;
q=0.8,application\/signed-exchange;v=b3;q=0.9",
      "agent":"Mozilla\/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit\/537.36 (KHTML, like Gecko) Chro
me\/84.0.4147.105 Safari\/537.36",
      "ip":"94.199.53.96",
      "java":false,
      "lang":"hu-HU",
      "color":24,
      "height":1280,
      "width":720,
      "tz":-120
   },
   "url":"https:\/\/payusdk.simplelabs.hu\/nandi\/v21_200814\/back.php",
   "total":"100",
   "language":"HU",
   "invoice":{
      "name":"SimplePay V2 Tester",
      "country":"hu",
      "state":"Budapest",
      "city":"Budapest",
      "zip":"1111",
      "address":"Address 1"
   }
}
```


Response in the case of a successful call


```
{
   "total":"100.0",
   "salt":"Kf2c7l6t3IJ9H0p90tR2T517x5T7T3Tm",
   "orderRef":"101010515976555937097",
   "merchant":"PUBLICTESTHUF",
   "currency":"HUF",
   "transactionId":"99355855"
}
```


Response if the card issuer wishes to interactively check the buyer during the **challenge**.


```
{
  "salt":"QnQtqSrVu01g331fNkXZBD2YusLzfG5U",
  "merchant":"PUBLICTESTHUF",
  "orderRef":"101010515976555937097",
  "currency":"HUF",
  "total":100.0
  "redirectUrl": "https://sandbox.simplepay.hu/pay/pay/ABCDEFG",
  "errorCodes": <errorCode>[]
}
```

2.3.2 Starting transaction in the buyer's absence

As the customer is not present in the case of a MIT or a REC, in these cases **there is no need for sending browser and url fields that are required for the challenge process.**


```
{
   "salt":"6d4aa6bcb099235639f551c4ce8cee1f",
   "merchant":"LRMIPSMHUF",
   "orderRef":"101010515976593416504",
   "currency":"HUF",
   "customerEmail":"sdk_test@simplepay.com",
   "sdkVersion":"SimplePay_PHP_SDK_2.0.9_200130:dc597528c6fb3b4c6312d470ed28895e",
   "cardData":{
      "number":"4908366099900425",
      "expiry":"0122",
      "cvc":"123",
      "holder":"V2 AUTO test"
   },
   "type":"MIT",
   "total":"100",
   "language":"HU",
   "threeDSReqAuthMethod":"02",
   "invoice":{
      "name":"SimplePay V2 Tester",
      "country":"hu",
      "state":"Budapest",
      "city":"Budapest",
      "zip":"1111",
      "address":"Address 1"
   }
}
```


The answer only specifies whether the debit transaction was successfully carried out or failed – in this case, however, no challenge is possible.

2.4 auto – with 3DS check using the merchant's MPI

If the merchant system has its own 3D secure (v1) solution (MPI component) and identification was carried out prior to the transaction, its results can be attached to the transaction request which will be forwarded to the authorisation system.

3D secure data:
xid: XID, unique ID generated for the identification request
eci: ECI (e-commerce indicator) the security level of the transaction which can be received in the form returned by the MPI)
cavv: CAVV/AAV/AEV, a cryptogram verifying identification or an attempted identification, in raw format

Starting a call with the merchant side 3DS check data

```
{
  "merchant":"PUBLICTESTHUF",
  "orderRef":"101010515417720487444",
  "customerEmail":"sdk_test@simplepay.com",
  "currency":"HUF",
  "total":"100",
  "sdkVersion":"SimplePay_PHP_SDK_2.0.6_190301",
  "salt":"d79ed6c2d9aa3a7f96a9f884f0ee3958",
  "cardData":{
    "number":"4908366099900425",
    "expiry":"1021",
    "cvc":"579",
    "holder":"v2 AUTO Tester"
  },
  "3DSecExternal":{
    "xid":"01234567980123456789",
    "eci":"01",
    "cavv":"ABCDEF"
  },
  "customer":"v2 AUTO Tester",
  "language":"HU",
}
```

2.5 SZÉP card acceptance

Payment with SZÉP Cards—issued by OTP—may also be initiated with the auto endpoint. In this case, the system can recognise the SZÉP Card on the basis of the card BIN (the first 6 characters of the card number) and send it on accordingly for debiting.

When the service was available the earlier SZÉP Card pockets were merged already; therefore, **the pocket—whose indication used to be a mandatory requirement—no longer needs to be specified and submitted**.

In case the merchant system wishes to send the pocket, it can be received by the API. In such cases the data should be sent in the **pocket** variable. Due to the merger only the value "09" of the previous value set is currently applied here, while the Aktív Magyarok pocket, created in 2025, can be identified with "08". If any other value is sent, it will not be taken into account by the API.

```
{
   "pocket":"09"
}
```

The transaction's statuses are the same as that of normal bank card payment.

SZÉP card payment can only work on merchant accounts separated from bank card payments. Consequently, transactions on the live and sandbox systems **require a merchant account configured to accept SZÉP cards only**.

2.6 SZÉP Card merchant's transaction identifier length

Unlike what is described in the SimplePay API documentation, in the case of TSP (i.e. SZÉP Card payment issued by OTP), the merchant transaction identifier must comply with the following:

- length: maximum 32 characters
- content: alphanumeric string

SZÉP Card payment, special attention must be paid to this!

2.7 SZÉP Card test in sandbox environment

Test transaction with SZÉP Card can be initiated on Sandbox with pre-set test card. Click on the card icon to select the card you wish to use for payment on the payment form. The following needs to be selected in the case of a SZÉP card:

number: 6101324296690851
expiry: 0926
holder: Sandbox Test

2.8 SZÉP Card and 3DS

3DS does not apply in the case of SZÉP card acceptance. There being no strong customer authentication, the challenge process cannot take place either.

2.9 Communication

The customer must be given clear information on the authorisation results.
In the case of a CIT such information needs to be placed/presented as specified in the "back – (information on the merchant's website)" chapter of the document concerned.

Compared to standard payments, timeout and transaction cancellation on the payment page are not a factor during **auto** payments, as in such scenarios card details are not provided on the SimplePay payment page, but on the merchant website.

Therefore, it is only necessary to display transaction information in the following scenarios

- successful payment
- failed payment

2.10 IPN

The IPN indicates the end of a successful transaction. It is not affected by the transaction initiation method, therefore it is received and processed the same way as described in the referenced documentation's "**IPN (Notification on performability)**" chapter.

3 PHP SDK

Setup and usage of the PHP SDK is discussed in detail in the referenced documentation. To use the **auto** function, the basic SDK has to be supplemented with the following downloadable sample code kit
https://simplepartner.hu/download.php?target=v2autosdk

Contents of the supplementary kit have to be extracted to the SDK main directory. Compared to the basic setup, no additional steps are required.

3.1 auto – transaction initiation with card data transfer

Sample code in the SDK: **auto.php**

The request initiation can be implemented as follows.

```
//Import config data
require_once 'src/config.php';

//Import SimplePayment class
require_once 'src/SimplePayV21.php';
require_once 'src/Auto.php';

$trx = new SimplePayAuto;

$currency = 'HUF';
$trx->addData('currency', $currency);

//add config data
$trx->addConfig($config);

//ORDER PRICE/TOTAL
$trx->addData('total', '100');

$trx->addData('orderRef', '101010515417720487444');

$trx->addData('customer', 'v2 AUTO Tester');

// customer's email
$trx->addData('customerEmail', 'sdk_test@simplepay.com');

// LANGUAGE
$trx->addData('language', 'HU');

//start transaction in SimplePay system
$trx->runAuto();
```

The response to the request is located in the array found in the $trx->transaction variable

```
Array
(
    [responseBody] => {"total":"100.0","salt":"Kf2c7l6t3IJ9H0p90tR2T517x5T7T3Tm","orderRef":"101010515
417720487444","merchant":"LRMIPSMHUF","currency":"HUF","transactionId":"99355853"}
    [responseSignature] => FYB52Ws+p5PTDhctyg81Mf4ztP4Z4xmGzl1fsEjn9jsnpP8yXvBOnjyeoTcqz9Y6
    [responseSignatureValid] => 1
```

```
    [total] => 100.0
    [salt] => Kf2c7l6t3IJ9H0p90tR2T517x5T7T3Tm
    [orderRef] => 101010515417720487444
    [merchant] => LRMIPSMHUF
    [currency] => HUF
    [transactionId] => 99355853
)
```

3.2 auto – initiating 3DS transactions with card details

Transactions initiated with the SDK **auto.php** file can be supplemented with external 3D Secure data as in the following.

```
//optional 3DS data
$trx->addGroupData('3DSecExternal', 'xid', '01234567980123456789');
$trx->addGroupData('3DSecExternal', 'eci', '01');
$trx->addGroupData('3DSecExternal', 'cavv', 'ABCDEF');
```

3.3 Communication

Communication details can be found in chapter **2.5**.

3.4 IPN

The IPN request is described in chapter **2.10**.

4 Logos and information pages

The SimplePay logo must be displayed in a permanently visible section of the part of the service establishment's site (e.g. in the footer), or during the payment method selection for the transaction.
The logo must be placed and displayed as described in the referenced documentation's **"Logos and information pages"** chapter.

5 Data Transmission Declaration

As the merchant transfers the order/customer data to a third party, the **customer must** explicitly **accept the data transfer declaration**.
The declaration has to be displayed, and accepted by customers, as described in the referenced documentation's "**Data Transmission Declaration**" chapter.

6 Testing

Every webstore is tested by SimplePay before deployment.

Tests are conducted as described in the referenced documentation's "**Testing**" chapter. Besides the key principles, and the tested and untested elements described there, special attention should be paid to the following.

6.1 Mandatory test elements

6.1.1 PCi-DSS

- The merchant system is compliant with PCI-DSS requirements, as certified by the **Attestation of Compliance** (AOC) issued by the certification body

6.1.2 SSL certification

- The merchant system uses SSL certification

6.1.3 Successful transaction

- The transaction runs appropriately until the end
- The transaction-specific communication is displayed (please see the referenced documentation's **back** communication for **successful** payments)
- Reception of IPN messages and proper response (see IPN handling in the referenced documentation)

6.1.4 Failed transaction

- The transaction runs appropriately until the end
- The transaction-specific communication is displayed (please see the referenced documentation's **back** communication for **failed** payments)

6.1.5 Display of the SimplePay Logo

- The SimplePay logo is displayed as described in chapter 4

6.1.6 Data Transmission Declaration

- The necessary declaration is displayed before starting the transaction, as described in chapter **5**

7 Support

For further information or technical support, please contact us at itsupport@simplepay.com.
For faster processing, please provide us with the data identifying the problem or your inquiry.

Transaction
If you have questions regarding transactions, please provide us with the **SimplePay** identifier of the payment. The **identifier is composed of nine digits**. in the case of sandbox, it is of the **1xxxxxxxx** format while in a live case it is of the **5xxxxxxxx** format.

Interface
Whether the transaction was started using V1 or V2 API.

Merchant account
Regarding technical configurations, please provide the identifier of the **merchant's account** within the SimplePay system. The identifier is the account's **MERCHANT value**.

Payment system
The system your inquiry relates to. **Sandbox system** exclusively for tests, and **Live system** for live payment transactions.

Deployment
In case of deployment tests, please contact us via itsupport@simplepay.com providing the following

- under which contracted domain name the testable payment was developed
- which account is in use (MERCHANT)
- where we can access the testable system