ThePay payment gate data API

Obsah

General API description

Data API is designated for downloading data from ThePay payment gate systems. It gives you access to information available in merchant administration through API - data about available payment methods, payments, payment statuses, accountings etc.

By using this API you can more automate processes in your application, synchronize data, increase payment process security, or implement uncommon requirements beyond those that are provided by our component.

API is available in two versions different in communication format:

1. SOAP web service - WSDL for production environment is available at https://www.thepay.cz/gate/api/data.wsdl, WSDL for development environment at https://www.thepay.cz/demo-gate/api/data-demo.wsdl

2. JSON - production URL https://www.thepay.cz/gate/api/data/, development URL https://www.thepay.cz/demo-gate/api/data/

Support for data API is included in our component. If you use it, you don't have to be concerned about communication - you will just use methods and classes provided by component.

Actual version - v 1.3

Actual API version supports these methods *getPaymentMethods*, *getPaymentState*, *getPayment, GetPayments* a *getPaymentInstructions*. SOAP and JSON formats are available.

Other methods will be implemented later. If you are interested in some unimplemented method, feel free to contact us, we comply with your wishes.

Common parameters used in most method calls

merchantId

Every request and response contain parameter *merchantId,* unique ID of merchant, that is same as payment gate ID.

accountId

Unique merchant account's ID, same as *accountId* for payment gate.

paymentId

Unique ID in ThePay system. During the getting back on payment gate (or during the receiving asynchronous payment notification) you can get the parameter *paymentId* – alternatively is possible to gain it through data API by GetPayments method. This ID you can also find in payment gate's administration interface.

Signature

Every single request and response includes signature for request verification. Request is SHA 256 hash from request parameters and merchant's account number.

Signature coding rules:

- In general, hash calculating form is sha256("merchantId=1&parametr1=parameter value&parametr2=...&password=12345")
- If parameter value is primitive data type, than his value will be appended to signature string
- If parameter value is object, than his SHA 256 hash (created by same algorithm which create whole request) will be used, but without password. Password will be used just once – during the final hash of whole request calculation.
- If parameter value is array, than his components will be concatenated by vertical line, |. For example string from array *[12, 34, 45]* will be *param1=12|23|*45. String for array of objects will be hashes of these objects concatenated with vertical line as separator.
- As password will be used communication password. You can find it in administrative interface, section merchant settings. Password for data API is different from payment gate password. Payment gate password is unique for merchant account, which means combination merchantId-accountId, while in data API merchant accounts are not distinguished.

Implementation component

For communication via API is possible to use classes prepared in implementation component, so you don't have to be concerned about communication, signature counting and received data – objects conversion either.

Individual operations are called through methods in class TpDataApiHelper (/helpers/TpDataApiHelper.php).

Classes that represent response and objects used in them you can find in folder /dataApi/. Object model and individual class parameter fits to API structure defined in WSDL document and to terms mentioned in this document.

API methods

getPaymentMethods

Return available payment methods along with their ID and names. Depending on input parameters this method returns either only methods which are currently available for merchant account or all methods.

Request

- **merchantId** int
- **accountId** int
- **onlyActive** boolean *(default true)* - whether will be returned only currently available methods, or all existing methods.
- **signature** string

Response

- **merchantId** int
- **accountId** int
- **methods** object[] – payment method's field
  - **id** int – unique method's identificator
  - **name** string – payment method's name
  - **active** boolean – whether is method available for merchant's account
- **signature** string

Payment method's logos

Payment method's logos are available in several sizes in this folders:

https://www.thepay.cz/gate/images/logos/public/tight/ - logos with transparent backgrounds equivalents to logo's size (logos are not in same size)
https://www.thepay.cz/gate/images/logos/public/[sirka]x[vyska]/ - rectangles in sizes [width]x[height]px with white background
Available logo's sizes:

- 86x86
- 100x67
- 100x100
- 150x100
- 150x150
- 200x150
- 200x200
- 209x127

Logo's name is [ID].png, where [ID] is payment method's ID returned by *getPaymentMethods* function. For example https://www.thepay.cz/gate/images/logos/public/86x86/11.png

If you need logos in different sizes, feel free to contact us, we comply with your wishes.

getPaymentState

Checks payment status according to ID. Similar to *getPayment* method in very simply way; designated to check payments in ThePay system.

Request

- **merchantId** int
- **paymentId** int
- **signature** string

Response

- **merchantId** int
- **state** int – payment status
- **signature** string

Possible payment statuses

1 – waiting for payment, payment wasn't completed
2 – payment was successfully completed
3 – payment wasn't completed, payment was canceled by customer
4 – some error occurred
6 – payment is partly paid
7 – payment is waiting for confirmation from payment system
8 – payment was cancelled and the money has been returned to customer
9 – amount is blocked on customer's account (in case of payments by card)

getPayment

Gets payment information by ID. Return all available information about payment. Returned values fits to data sent by merchant during the creating payment, actual payment status and used payment method.
More descriptions of parameters you can find in implementation manual for payment component.

Tip: If you need to find payment in accordance to *merchantData*, use method *getPayments*.

Request

- **merchantId** int
- **paymentId** int
- **signature** string

Response

- **merchantId** int
- **payment** payment – payment information
  - **id** int – unique payment ID
  - **account** int – merchant account ID
  - **state** int – payment status (possible states – see method *getPaymentState*)
  - **createdOn** datetime – payment creation date
  - **finishedOn** datetime – date when payment was paid
  - **canceledOn** datetime – payment cancelation date

    - **payOff** int – settlement ID; in this settlement payment was paid to merchant
    - **payOffCancel** int – settlement ID; in the course of this settlement cancelled payment was charged
    - **value** decimal – payment value
    - **receivedValue** decimal – received amount; used by partly paid payments
    - **currency** int – currency ID
    - **fee** decimal – transaction fee
    - **description** string – description for customer
    - **merchantData** string – merchant's data
    - **paymentMethod** id – payment method ID
    - **specificSymbol** string – specific symbol used in case of permanent payment
    - **merchantSpecificSymbol** string – merchant specific symbol
    - **accountNumber** string – sender's (owner's) account number
    - **accountOwnerName** string – sender's (owner's) account name
    - **returnUrl** string – return URL
    - **permanentPayment** int – permanent payment ID
    - **deposit** boolean – whether should be amount charged from account
    - **recurring** boolean – if payment is marked as recurring
    - **ip** string – customer's IP address
    - **customerEmail** string – customer's e-mail address
    - **fik** string – EET - fiscal identification code (FIK)
    - **bkp** string - EET – security code of taxpayer (BKP)
    - **pkp** string – EET - signature code of taxpayer (PKP)
    - **receiptUrl** string – EET – web EET receipt URL, if merchant have web receipts active
    - **firstSuccess** boolean - EET – if receipt was accepted by EET system on first attempt or not
- **signature** string

getPaymentInstructions

Gets instructions for paying of specific payment, i.e. link to payment gate in case of online payment, or data in case of offline payment.

Parameters *accountNumberPrefix*, *accountNumber*, *bankCode*, *vs*, *ss* a *ebankingUrl* are returned only in case of unpaid payments through offline payment methods.

Parameter *methodChangeUrl* is returned only in case of unpaid payments.

Parameters *scCode* a *scBarcodeUrl* are filled only in case of unpaid payment through SuperCash method.

Tip: If you need to find payment instruction according to *merchantData*, first of all use *getPayments* method to get *paymentId*.

Request

- **merchantId** int
- **paymentId** int
- **signature** string

Response

- **merchantId** int
- **paymentInfo** paymentInfo – payment information
    - **isOffline** boolean – whether is payment method online or offline
    - **paymentPageUrl** string – URL of page with payment instructions (offline methods) or with redirecting to bank payment gate (online methods). If payment is paid, customer will be redirected to page with payment status information.
    - **paymentInfoUrl** string – URL of page with payment information
    - **methodChangeUrl** string - URL of page where customer can change payment method.
    - **value** decimal – payment value
    - **accountNumberPrefix** string – account number prefix
    - **accountNumber** string - account number
    - **bankCode** string – bank code
    - **vs** string – payment reference number (variable symbol)
    - **ss** string - specific symbol
    - **ebankingUrl** string – URL of internet banking
    - **scCode** string – SuperCash code
    - **scBarcodeUrl** – URL with SuperCash bar code
- **signature** string

GetPayments

Gets merchant’s payments list filtrated and paged by entered parameters.

Request

- **merchantId** int
- **searchParams** paymentsSearchParams – parameters for payment (analogical to filter in administration)
    - **accountId** int[] – merchant's account ID
    - **state** int[]- payment status
    - **currency** int[] – payment currency
    - **valueFrom** decimal – minimal payment value
    - **valueTo** decimal – maximal payment value
    - **createdOnFrom** datetime – created on from date
    - **createdOnTo** datetime – created on to date
    - **finishedOnFrom** datetime – finished on from date
    - **finishedOnTo** datetime – finished on to date
    - **accounting** int[] – settlement in which payment was paid to merchant
    - **description** string – payment description
    - **merchantData** string – merchant’s data
    - **method** int[] – payment method ID
    - **specificSymbol** string - specific symbol

    All attributes are optional, whole parameter *searchParams* either.

    More values in one parameter are connected by operator OR.

Individual parameters are connected by operator AND, i.e. payments fittings to all requirements will be found.

Interval <valueFrom, valueTo> is closed from both sides, so amounts in the endpoints of the interval are fitting too.

Interval <createdOnFrom, createdOnTo) and <finishedOnFrom, finishedOnTo) is closed from the left side. In case of looking for all payments during the month, use format 1.1.2016 00:00:00 as the lowest level and 1.2.2016 00:00:00 as the highest level.

Date and time values are transferred in format ISO 8601, seconds has to be whole number; it is not possible to transfer timestamp where are seconds and then another part of numbers after decimal point.

Strings values *description*, *merchantData* and *specificSymbol;* are searched with command LIKE '%...%', searching is not case sensitive.

- **pagination** object – pagination settings
  - **page** int *(default 0)* – required page, numbering from 0
  - **itemsOnPage** int *(default 50)* – numbers of items on page
- **ordering** object – ordering setting
  - **orderBy** string *(default "id")* – payment parameter (correspond to parameter of *payment* object); according to this parameter all payments will be ordered
  - **orderHow** string *(default "DESC")* – ordering form ASC/DESC
- **signature** string

Response

- **merchantId** int
- **payments** payment[] – field for payments that were found. Individual payments are in same format as *payment* in response of method *getPayment*.
- **pagination** object – pagination information
  - **page** int – actual page number, numbering from 0
  - **itemsOnPage** int - numbers of items on page
  - **totalPages** int – total number of pages
- **signature** string

SetPaymentMethods

Set payment methods available for merchant's account.

Request

- **merchantId** int
- **accountId** int
- **type** string *(all|whitelist)* – if all payment methods should be available (*all*) or only selected ones (*whitelist*).
- **paymentMethods** int[] – IDs of payment methods, which should be available. If *type=whitelist* field is required and must contains at least one ID. If *type=all*, field is not requiered and it's content is ignored.
- **signature** string

Response

- **merchantId** int

- **accountId** int
- **status** string *(OK|ERROR)* – result. OK = successfully set, ERROR = error during processing, setting was not saved
- **signature** string

GetAccountings

Gets list of settlement with filtration and pagination according to parameters

GetAccounting

Gets settlement details according to settlement ID.

GetActualState

Actual merchant account balance and next settlement pay-day

GetAccounts

List of merchant accounts in ThePay system.

GetPaymentStates

List of payment states that are used.