Peach Payments Masterpass Integration Guide

Overview

Masterpass is a digital wallet payment method that allows your customers to quickly and securely pay online without having to manually enter their payment credentials. This streamlined purchase flow provides customers the ability to select masterpass on your checkout page, select a payment method and verify their payment all through their Masterpass wallet. Once the customer has verified their purchase, you will be able to complete the transaction and fulfill the customer experience.

This document will provide merchant partners with instructions on how to integrate Masterpass as a payment option on checkout pages for web using the Peach Payments REST API.

Experience

The checkout experience streamlines your payment experience with a quick way for consumers to check out, without having to enter sensitive card details on your website.

1) Merchant partners will display a masterpass payment button on their checkout page when a customer is ready to compete payment of their purchase.
2) When a customer selects masterpass the merchant will be redirect the customer to a payment page hosted by Peach Payments. In the redirect flow, the merchant will securely send an API POST request to Peach Payments with their Masterpass authentication credentials along with the purchase details.
3) The hosted payment page will page will display to the customer the purchase details with a QR code image and pin code to complete and verify the purchase.
4) The customer will use their Masterpass digital wallet to select their payment method and verify the purchase transaction by scanning the QR code or entering the pin code displayed on the hosted payment page.
5) Once successful, the customer will be redirected back to your checkout page with a confirmation of the transaction for you to complete and confirm the purchase. The redirect will include an API GET request to retrieve the complete details of the completed transaction.
6) If a valid webhook is enabled, the merchant partner will receive a response to their server endpoint with the details of the completed transaction.
7) The customer will receive a purchase confirmation from their Masterpass digital wallet.

Brand Guidelines

When displaying the Masterpass purchase option on the merchant checkout page, refer to the Masterpass branding requirements.

https://developer.mastercard.com/page/masterpass-branding-requirements

Masterpass assets are available for merchant use.

https://developer.mastercard.com/page/masterpass-digital-assets

Getting Started

To get started, merchant partners will need to setup a Masterpass account with their bank. To get set up, reach out to the Peach Payment team, `support@peachpayments.com`.

The Peach Payments team will initially provision the merchant partner with test environment access to begin integration and payments testing.

Once a merchant partner has completed the Masterpass integration and has successfully tested purchase transactions, the merchant integration will be reviewed by the Peach Payments team. Email our team when the Masterpass integration is ready for review `support@peachpayments.com`. The review process will take up to two (2) days.

API Reference

Hosts

Test: `https://testapi.peachpayments.com/v1`

Live: `https://api.peachpayments.com/v1`

Postman Collection:
`https://www.getpostman.com/collections/aa1179d07e84eb4a0fa5`

Security / Authentication

All requests must be made over SSL.

Test and Live environment credentials are generated through the Peach Payments Console. The `userid` and `password` will authorize each API request. The `entityid` associates the merchant channel entity with the Masterpass integration. During account setup the Peach Payments team will assign the desired merchant channel and share the required `entityid` .

`Console.peachpayments.com`
(coming soon)

Authentication Parameters	**Description**	**Format**	**Required**
authentication.userId	The userId for the entity. Required to authenticate a server-to-server request	AN32 [a-f0-9]{32}	Required
authentication.password	The password for the userId. Required to authenticate a server-to-server request	AN32 [a-zA-Z0-9]{8, 32}	Required
authentication.entityId	The entity for the request. By default this is the channel's ID. It can be the division,	AN32 [a-f0-9]{32}	Required

	merchant or channel identifier.		

Payments API

To accept Masterpass as a payment method add the Masterpass payment option to the checkout process on the merchant website. When the Masterpass payment option is selected in the checkout experience by the customer, the merchant will initiate a POST request over HTTPS to the `/v1/payments` endpoint to send the initial payment request. The response from this POST request will include a `redirect.url` which the merchant will redirect the shopper to verify and complete their purchase using their Masterpass digital wallet. Once the customer has verified and completed their purchase they will be redirected back to the merchant's `shopperResultUrl`. In the redirect, the merchant will receive a GET parameter `resourcePath` that will be used to retrieve the transaction status. Merchants will now be able to complete the customer purchase if successful or to determine the next steps depending on the result.

Send an initial payment

Perform a server-to server POST request to the `/v1/payments` endpoint to create the payment `id` and to generate a `redirect.url` to redirect the customer to the Peach Payments hosted payment page to complete the purchase. This POST request will contain the authentication parameters, purchase parameters and any custom parameters that a merchant can optionally send.

Payment POST Parameters

Parameter	Description	Format	Required
authentication.userId	The userId for the entity. Required to authenticate a server-to-server request	AN32 [a-f0-9]{32}	Required
authentication.password	The password for the userId. Required to authenticate a server-to-server request	AN32 [a-zA-Z0-9]{8,32}	Required
authentication.entityId	The entity for the request. By default this is the channel's ID. It can be the division, merchant or channel identifier.	AN32 [a-f0-9]{32}	Required
merchantTransactionId	Merchant-provided reference number, should be unique for your transactions.	AN255 [\s\S]{8,255}	Required
amount	Indicates the amount of the payment request. The dot is used as decimal separator.	N10.N2 [0-9]{1,10}(\.[0-9]{2})?	Required

paymentBrand	The payment brand of the request. For Masterpass this will be set to `MASTERPASS`	AN32 [a-zA-Z0-9_] {1,32}	Required
paymentType	The payment type for the request. You can send payment requests with one of the following types: `DB`, `CD`, `RV` or `RF`.	A2	Required
currency	The currency code of the payment request amount.	A3 [A-Z]{3}	Required
shopperResultUrl	This URL will receive the result of an asynchronous payment. Must be sent URL encoded.	AN2048 [\s\S]{6,2048}	Conditional
customParameters[*name*]	A name value pair used for sending custom information.	name: AN64 [a-zA-Z0-9\._]{3,64} value: AN2048 [\s\S]{0,2048}	Optional
merchantInvoiceId	Merchant-provided invoice number, should be unique for your transactions. This identifier is not sent onwards.	AN255 [\s\S]{8,255}	Optional

JSON Response


```
{
        "connectorTxID1": "29740001047",
        "customParameters": {},
        "paymentBrand": "MASTERPASS",
        "currency": "ZAR",
        "descriptor": "Example Merchant",
        "merchantTransactionId": "8p829494638ef34e0146fcd847001eoe",
        "result": {
                "code": "000.100.110",
                "description": "Request successfully processed in 'Merchant in
Integrator Test Mode"
        },
        "redirect": {
                 "url": "https://testapi.ppay.io/v1/verify/",
                 "parameters": [
                      {"name": "customer",
                                "value": "7064417334"},
                      {"name": "amount",
                               "value": "50.00"},
                      {"name": "connector",
                              "value": "MASTERPASS"},
                      {"name": "currency",
                              "value": "ZAR"},
                      {"name": "transaction",
                             "value": "eefeb77a41df481peace7bf6f52e80a49”}
                ],
```

```
                "method": "POST"
            },
            "amount": "50.00",
            "shopperResultUrl": "http://merchant.example.com",
        },
        "id": "eefeb77a41df481peace7bf6f52e80a49",
        "paymentType": "DB",
        "timestamp": "2018-07-30T12:18:19.453777Z",
            "resultDetails": {
            "expiryDate": "2018-07-30T10:31:31.864000Z",
            "ExtendedDescription": "Payment process started.",
            "code": "https://masterpasstest.oltio.co.za/pluto/public/qr/495008905",
            "AcquirerResponse": "PENDING"
            }
}
```


Parameter	Description	Format	Required
connectorTxID1	Masterpass transaction reference identifier.		Required
customParameters[*name*]	A name value pair used for sending custom information.	name: AN64 [a-zA-Z0-9\._]{3,64} value: AN2048 [\s\S]{0,2048}	Optional
paymentBrand	The payment brand of the request. For Masterpass this will be set to `MASTERPASS`	AN32 [a-zA-Z0-9_] {1,32}	Required
currency	The currency code of the payment request's amount.	A3 [A-Z]{3}	Required
descriptor	Currenlty responds with null.		
merchantTransactionId	Merchant-provided reference number, should be unique for your transactions. Some receivers require this ID. This identifier is often used for reconciliation.	AN255 [\s\S]{8,255}	Conditional
result	The unique code that indicates the result status of the request. See the Result Codes for more detailed information.	AN11 [0-9\.]{2,11}	Required
redirect.url	URL the the shopper must be redirected to in order to proceed.	AN2048 [\s\S]{6,2048}	Conditional
redirect.parameters[n].name	List of parameter names for the redirect.url. The corresponding parameter value is the same parameter number ending with	AN255 [\s\S]{1,255}	Conditional

	.value like described in the line below.		
redirect.parameters[n].value	The parameter values corresponding to the names as described above.	AN255 [\s\S]{1,255}	Conditional
amount	Indicates the amount of the payment request. The dot is used as decimal separator.	N10.N2 [0-9]{1,10}(\.[0-9]{2})?	Required
shopperResultUrl	This URL will receive the result of an asynchronous payment. Must be sent URL encoded.	AN2048 [\s\S]{6,2048}	Conditional
id	The identifier of the checkout request that can be used to reference the payment later. You get this as the field id of a checkout's response and then should use it as the {id} part in subsequent steps.	AN48 [a-zA-Z0-9.\-]{32,48}	Required
paymentType	The payment type for the request. Types: `DB`, `RV` or `RF` are supported.	A2	Required
timestamp	The timestamp the response has generated.	date yyyy-MM-dd hh:mm:ss	Required
resultDetails	A container for name value pair used for enriching the response with Masterpass response details.	name: AN64 [a-zA-Z0-9\._]{3,64} value: AN2048 [\s\S]{0,2048}	Conditional

Redirect the shopper

The next step is to redirect the customer to the Peach Payments hosted payment page to complete their purchase . To do this you must parse the `redirect` object in the payments response which contains the `redirect.url` and `redirect.parameters`. These parameter will be sent via HTTP POST to the `redirect.url`. During this time the transaction is pending

On the Peach Payments hosted payment page, the customer will be able to use their Masterpass digital wallet to scan a QR code image or enter a pin code to complete and verify the purchase. **The customer will have 30 minutes to complete the transaction.**

Sample code for redirect

```
<form name="submitForm" action="https://testapi.ppay.io/v1/verify/ >
```

```
<input type="hidden"
        name="customer"
        value="08063317664">
<input type="hidden"
        name="connector"
        value="MASTERPASS">
<input type="hidden"
       name="amount"
       value="50.00">
<input type="hidden"
       name="currency"
       value="ZAR">
<input type="hidden"
        name="transaction"
        value="eefeb77a41df481peace7bf6f52e80a49">
<noscript>
<input type='submit' name='submitButton'/>
</noscript>
</form>
```

Get the payment status

Once the purchase has been processed, the customer is redirected to the `shopperResultUrl` along with a GET parameter resourcePath.

IMPORTANT: The baseUrl must end in a "/", e.g. "`https://testapi.ppay.io/`".
To get the status of the payment, you should make GET request to the baseUrl + resourcePath, including your authentication parameters.

This endpoint can also be queried to check the payment status during the time which the customer has been redirected to the payment verification page. Note that only two requests per minute is allowed.


```
{
    "resultDetails": {
        "ExtendedDescription": "The transaction has completed.",
        "AcquirerResponse": "SUCCESS",
        "expiryDate": "2018-11-02T13:02:22.042000Z",
        "code": "https://masterpasstest.oltio.co.za/pluto/public/qr/0130388269"
    },
    "paymentBrand": "MASTERPASS",
    "result": {
        "description": "Request successfully processed in 'Merchant in Integrator
Test Mode'",
        "code": "000.100.110"
    },
    "id": "823019eb8171424f9229ebbb01b3fa62",
    "card": {
        "expiryYear": null,
        "last4Digits": "0106",
        "expiryMonth": null,
        "holder": "Debit",
        "bin": "500100"
```

```
    },
    "customParameters": {},
    "paymentType": "DB",
    "currency": "ZAR",
    "timestamp": "2018-11-02T12:34:28.650452Z",
    "merchantTransactionId": "MasterTest1",
    "amount": 50
}
```

Refunds

A refund is performed against a previous payment, referencing its payment `id` by sending POST request over HTTPS to the `/v1/payments` endpoint. The POST request will contain the authentication parameters, a payment type, refund amount, refund and currency. The `paymentType` parameter will be set to `RF`.

Parameter	Description	Format	Required
authentication.userId	The userId for the entity. Required to authenticate a server-to-server request	AN32 [a-f0-9]{32}	Required
authentication.password	The password for the userId. Required to authenticate a server-to-server request	AN32 [a-zA-Z0-9]{8,32}	Required
authentication.entityId	The entity for the request. By default this is the channel's ID. It can be the division, merchant or channel identifier. Division is for requesting registrations only, merchant only in combination with channel dispatching, i.e. channel is the default for sending payment transactions.	AN32 [a-f0-9]{32}	Required
amount	Indicates the amount of the payment request. The dot is used as decimal separator.	N10.N2 [0-9]{1,10}(\.[0-9]{2})?	Required
paymentType	The payment type for the request. You can send payment requests with one of the following types: `RF`.	A2	Required
currency	The currency code of the payment request's amount.	A3 [A-Z]{3}	Required

JSON Response


```
{
    "currency": "ZAR",
    "id": "eefeb77a41df481peace7bf6f52e80a49",
    "amount": "100.00",
    "result": {
        "description": "Request successfully processed in 'Merchant in Integrator Test Mode'",
        "code": "000.100.110"
    },
    "timestamp": "2018-08-01T08:24:36.308493Z",
    "paymentType": "RF"
}
```


Webhooks

Peach Payments webhooks
https://peachpayments.docs.oppwa.com/tutorials/webhooks

Webhooks are HTTP callbacks that notify you of all events you subscribed to for an entity.

Configuration

Transaction webhooks can be set up and managed for both Test and Live environment, from within the Peach Payments Console.

Console.peachpayments.com
(coming soon)

Integration

The decrypted notification body contains the type of notification and its payload

```
{
    "payload": [content],
    "type": [notification_type]
}
```

Parameter	Description	Format	Required
type	Type of the notification (this is always set to PAYMENTS)	PAYMENTS	Required
payload	Content of the notification	JSON	Required

Example: Payment

```
{
'payload': {
   'amount': 50.0,
   'authentication': {'entityId': 'd4b10f6465db11e7b43e0245ffcc0db9'},
   'card': {
        'Bin': '500100',
        'expiryMonth': 'null',
        'expiryYear': 'null',
        'holder': 'Debit',
        'last4Digits': '0106'},
 'currency': 'ZAR',
 'customParameters': {
        'SHOPPER_promoCode': 'AT052'},
 'id': 'eefeb77a41df481peace7bf6f52e80a49',
 'paymentBrand': None,
 'paymentType': 'DB',
 'presentationAmount': 50.0,
 'presentationCurrency': 'ZAR',
 'result': {
        'code': '000.000.000',
        'description': 'Transaction succeeded'},
 'timestamp': '2018-08-02 13:47:10.953154+00:00'},
'type': 'PAYMENTS'
}
```

Encryption

The content of the notification are encrypted to protect data from fraud attempts.

Parameter	**Description**
Encryption algorithm	AES
Key	[secret of listener] 64 character-long hexadecimal string in configuration (Provided by Peach Payments)
Key length	256 bits (32 bytes)
Block mode	GCM
Padding	None
Initialization Vector	In HTTP Header (X-Initialization-Vector)
Authentication Tag	In HTTP Header (X-Authentication-Tag)

An example of an encrypted notification:

HEADERS

X-Authentication-Tag: 9CDE76F1E157428D3CFF8CBCB5E5BFED

Accept-Encoding: gzip

Cf-Connecting-Ip: 213.131.241.51

X-Dynatrace: FW1;-1;-2107993456;7056;15;-2107993456;7056;3

X-Initialization-Vector: F0B3A54CC4043F28BE7CA326

Content-Type: application/json; charset=UTF-8

X-Request-Id: 67c70736-b4b6-4369-84a2-ecaf00e7f0ec

Content-Encoding: UTF-8

Cf-Ray: 3a86a01909170f75-FRA

Via: 1.1 vegur

Connect-Time: 0

Cf-Ipcountry: DE

User-Agent: Apache-HttpClient

Content-Length: 206

Total-Route-Time: 0

Cf-Visitor: {"scheme":"https"}

Connection: close

Host: requestb.in (This depends on your webhook url)

RAW BODY
{"encryptedBody":"12C148F1C9C1AEA110A5E9DAC8EABDF994BA6314387C2B3ABB29CF37A64"}

Responding to Notifications

When your service receives a webhook notification, it must return a 200 HTTP status code. Otherwise, the webhook service considers the notification delivery as failed, and will retry to send the notification later.

Protocol Details:
Protocol: HTTPS (HTTP is allowed in test systems only)
HTTP method: POST
Content Type: text(text/plain)

Decryption

The decryption tool/method is dependent on the programming language you are integrating with. To view your programming language's decryption example, please follow this link:
https://peachpayments.docs.oppwa.com/tutorials/webhooks/decryption-example

Result Codes

The result codes are part of the response body's JSON (field result) containing a code and a description explaining the code.

Result Code	Description
000.000.000	Transaction successfully processed in LIVE system
000.100.110	Transaction successfully processed in TEST system
100.396.101	Cancelled by user
100.396.104	Uncertain status - probably cancelled by user
600.200.500	Invalid payment data. You are not configured for this currency of sub type
200.300.404	Parameter in the incorrect format (The returned description will give details on which parameter is in the incorrect format)
600.200.400	Unsupported Payment Type (only DB is allowed as a payment type)
800.900.300	Invalid authentication information (Authentication password is incorrect)
800.900.201	Unknown channel (Entity Id in payment request is incorrect)

HTTP Status Codes

200	Successful
307	Temporary redirect
400	bad request. This might either point to e.g. invalid parameters or values sent. It's also returned if the payment failed e.g. because the acquirer declined.
403	incorrect authentication information, e.g. one of the authentication.* parameters is wrong - please check them or contact us for correct parameters
404	requested resource or endpoint is not found. I.e. endpoint/url doesn't exist. This can also be caused by typos like POST

	/v1/paymnets instead of payments or wrong IDs like GET /v1/payments/{id} where no payment with {id} exists.

Rate Limits

There is a limit two (2) requests per minute on the number of requests that a merchant can make to the purchase status API.

Testing

When a merchant is ready to test and integrate Masterpass a email will need to be sent to `support@peachpayments.com` who will then setup the merchant with a Test environment for Masterpass..

The Peach Payments team will provide API authentication credentials for the Test environment to the merchant to begin the integration and testing.

Test Wallet

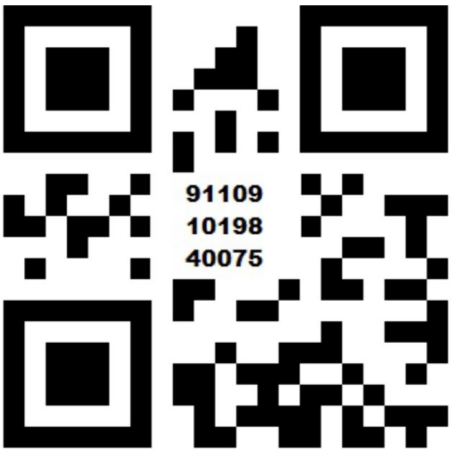
While testing, merchants will be able to access a test Masterpass digital wallet to simulate and test the complete end-to-end checkout experience. The test digital wallet can be accessed from within the existing Masterpass digital wallet app on Android and iOS, download links are listed below. For testing on Test the digital wallet app will need to be set to Test mode. Scan the code below to switch the Masterpass digital wallet app between Test and Live.

Android

- https://play.google.com/store/apps/details?id=com.oltio.nedbank.masterpass
- https://play.google.com/store/apps/details?id=com.oltio.standard.bank.masterpass
- https://play.google.com/store/apps/details?id=com.oltio.absa.bank.masterpass
- https://play.google.com/store/apps/details?id=com.oltio.capitec.masterpass

iOS

- https://itunes.apple.com/za/app/nedbank-masterpass/id1015918517
- https://itunes.apple.com/za/app/standard-bank-masterpass/id895028482
- https://itunes.apple.com/za/app/masterpass-from-absa/id993577609
- https://itunes.apple.com/za/app/masterpass-from-absa/id993577609
- https://itunes.apple.com/gb/app/capitec-masterpass/id1084707614



Test Cards

Test cards will need to be added to the Masterpass digital wallet in order to test transactions in the Test environment. For each test card below, add 4 more digits to the card ranges below and they will return the desired response. Please note that many other testers use these number ranges so if you receive the card relink screen on the app simply cancel and try another last 4 digits.

These below test cards will test only the below mentioned scenarios and will only work once the Masterpass digital wallet is switched to Test.

Debit Card
Any 18-digit card starting with:
50010001000105 with any bank PIN = 00 response (success)

Any 18-digit card starting with:
50010001000101 with any bank PIN = 51 response (insufficient funds)

Any 18-digit card starting with:
50020001000103 with any bank PIN = 91 response (issuer or switch inoperative/bank unavailable)

Credit Card
Any 18-digit card starting with:
50020001000105 with any bank PIN = 00 response (success)

Any 18-digit card starting with:
50020001000101 with any bank PIN = 51 response (insufficient funds)

Any 18-digit card starting with:
50020001000103 with any bank PIN = 91 response (issuer or switch inoperative/bank unavailable)

View Test Transactions

To view test transaction
Console.peachpayments.com
(coming soon)

Manage Test Webhooks

Merchants are able to set their transaction webhooks for test transaction from the Test environment within the Peach Payments Console.

Console.peachpayments.com
(coming soon)

Revision History

Updated Aug 13, 2018
Updated November 2, 2018
Updated May 15, 2019 - Change to API Endpoint Base URL’s