https://verifone.cloud/docs/online-payments/accepting-card-payments/googlepay
Updated: 12-Nov-2024

Google Pay™

Overview

Checkout can be used to accept Google Pay payments. Google Pay payments are card payments, so they require an Acquiring connection.

This guide requires familiarity with Accepting payments.

Google Pay processing fields

- customer ? (**Required**) Id of a Customer created via the Customer API. A customer object can be created and attached to a Checkout. The customer object can store relevant customer details. Some of these details might be required, depending on the payment method and/or authentication mechanisms (e.g. 3DS)
    - email_addres ? In order to trigger a 3DS transaction with Google Pay, the customer field needs a mandatory email_address parameter attached.
- configurations.google_pay ? (**Required**) Object carrying the parameters required for making an Google Pay payment
    - dynamic_descriptor ? Short text to be displayed on the bank statement of the cardholder. Support varies per Payment Contract.
    - capture_now ? Used for separate authorization and capture
    - account_validation ? Boolean flag to mark transaction as account validation ($0 auth)
    - card ? (**Required**)
        - payment_contract_id ? This id can be found in the portal or given to you by a Verifone employee. It is used to retrieve MID and other merchant configuration necessary for a payment.
        - sca_compliance_level ? Strong Customer Authentication (SCA) compliance level. Determines the rules of 3DS usage.
        - threed_secure ? Used for 3D-Secure payments
        - authorization_type ? Flags transaction as pre/final authorization

Example request

```
{
    "amount": 9988,
    "currency_code": "EUR",
    "entity_id": "{{entity_id}}",
    "configurations": {
        "google_pay": {
```

```
            "dynamic_descriptor": "Sneakers deluxe",
            "card": {
                "sca_compliance_level": "NONE",
                "payment_contract_id": "{{payment_contract_id}}"
            }
        }
    },
    "merchant_reference": "ORDER-4432",
    "return_url": "https://examplemerchant.com/order/1234/return"
}
```

Authorization and capture

Checkout can be used to do a sale (captureNow = true) or to authorize without capturing immediately (captureNow = false). An authorized payment reserves the money in the cardholder's account and allows you to capture the funds at a later stage. Authorizing the payment can be done using Checkout, capturing the payment is done through a separate API call or through Verifone Central.

You cannot capture more or less than has been authorized by the cardholder per transaction.

Example: If a payment has been authorized for 20.00 EUR; it is only possible to perform a full capture for 20.00 EUR.

Requirement

Set the configurations.google_pay.capture_now value to `false`. Setting it to true will immediately capture the payment.

Captures can only be done on transactions with the status AUTHORIZED.

After you have used Checkout to authorize a payment, you will receive the id of the transaction in the transaction_id query string parameter appended to the return_url. This id needs to be referenced for capturing the payment.

MOTO

Google Pay transactions are **not compatible** with Mail or Telephone shopper interactions.

Account verification

Sometimes a merchant wants to ensure that the customer's card details are valid and can be used to make a payment, without making the actual payment at the time. This can be done via a transaction referred to as an Account Verification. Account Verification transactions do not have an amount, so there is no money being blocked or moved from the customer's card. However, processing such transaction can provide the merchant with the knowledge on whether the card is valid, but also indicate CVV validity and AVS information, depending on if it is supported by your acquirer supports this feature. Account verification is also known as 'zero-value'.

Setup

Checkout pages can be set up for account verification. This is done by providing the amount of the Checkout as 0. If configurations.google_pay.capture_now is set to true, it will be overridden with false because it is not possible to capture an account verification payment.

3D-Secure

Account verification can be used in combination with 3D-Secure. Follow the steps in Accepting 3-DSecure payments to configure Checkout for 3D-Secure, then set the `amount` to `0` (zero). No money will be reserved on the cardholder's account and an authentication will be done.

Handling responses

Whenever a Google Pay payment is processed via the Checkout, the responses events would contain additional fields in the **details** object.

Example of successful Google Pay payment via the Checkout:


```
[
    {
        "type": "TRANSACTION_SUCCESS",
        "id": "f2041250-4fc2-4b3a-bc94-651ba099541a",
        "timestamp": "2020-07-08T12:42:37.974Z",
        "details": {
            "id": "927bdeb4-afb9-44ff-9bf2-6348e2080c82",
            "payment_provider_contract":
"8d3c506b-5585-4c1c-8530-2319237f6385",
            "amount": 12322,
            "blocked": false,
            "customer": null,
            "merchant_reference": "ORDER-9633",
            "payment_product": "CARD",
            "status": "AUTHORIZED",
            "authorization_code": "5669  ",
            "created_by": "ffa1ac64-d04c-4af1-9e71-c7aad3c854d5",
            "cvv_result": "0",
            "details": {
                "auto_capture": true
            },
            "reason_code": "0000",
            "rrn": "ORDER-9633",
            "shopper_interaction": "ECOMMERCE",
            "stan": "041796",
            "reversal_status": "NONE",
            "geo_location": [
                51.9336,
                4.4888
            ],
            "city": "Rotterdam",
            "country_code": "NLD",
            "additional_data": {
```

```
                "acquirer_response_code": "00",
                "initiator_trace_id": "041796"
            }
        }
    }
]
```


Example of failed Google Pay payment via the Checkout:


```
[
    {
        "type": "TRANSACTION_FAILED",
        "id": "f2041250-4fc2-4b3a-bc94-651ba099541a",
        "timestamp": "2020-07-08T12:42:37.974Z",
        "details": {
            "id": "927bdeb4-afb9-44ff-9bf2-6348e2080c82",
            "payment_provider_contract":
"8d3c506b-5585-4c1c-8530-2319237f6385",
            "amount": 12322,
            "blocked": false,
            "customer": null,
            "merchant_reference": "ORDER-9633",
            "payment_product": "CARD",
            "status": "FAILED",
            "created_by": "ffa1ac64-d04c-4af1-9e71-c7aad3c854d5",
            "cvv_result": "0",
            "details": {
                "auto_capture": true
            },
            "shopper_interaction": "ECOMMERCE",
            "stan": "041796",
            "reversal_status": "NONE",
            "geo_location": [
                51.9336,
                4.4888
            ],
            "city": "Rotterdam",
            "country_code": "NLD",
            "additional_data": {
                "acquirer_response_code": "00",
                "initiator_trace_id": "041796"
            }
        }
    }
]
```


When a transaction has been declined the events would look like this:


```
[
    {
        "type": "TRANSACTION_DECLINED",
        "id": "f2041250-4fc2-4b3a-bc94-651ba099541a",
        "timestamp": "2020-07-08T12:42:37.974Z",
        "details": {
            "id": "927bdeb4-afb9-44ff-9bf2-6348e2080c82",
            "payment_provider_contract":
```

```
"8d3c506b-5585-4c1c-8530-2319237f6385",
            "amount": 12322,
            "blocked": false,
            "customer": null,
            "merchant_reference": "ORDER-9633",
            "payment_product": "CARD",
            "status": "DECLINED",
            "created_by": "ffa1ac64-d04c-4af1-9e71-c7aad3c854d5",
            "cvv_result": "0",
            "details": {
                "auto_capture": true
            },
            "shopper_interaction": "ECOMMERCE",
            "stan": "041796",
            "reversal_status": "NONE",
            "geo_location": [
                51.9336,
                4.4888
            ],
            "city": "Rotterdam",
            "country_code": "NLD",
            "additional_data": {
                "acquirer_response_code": "05",
                "initiator_trace_id": "041796"
            }
        }
    }
]
```


To ensure that the redirection request was not tampered with, always check that the transactionId received as query parameter in the redirection matches the transactionId property of the retrieved Checkout. If those are not matching, this is indication of either an incorrect integration, that the redirection to your returnUrl did not originate from Verifone or it was tampered with.

You can now store the transactionId value together with the order 1234 in your system to link the two together.

Scenarios

The table below describes the different outcomes of a Checkout. Examples for reproducing the scenarios can be found in the section below it. A full list error codes are available.

Description	3D-Secure Configured	Result	Merchant action
Failed transaction*	N	Redirect: checkout_id={checkoutId} & transaction_id={transactionId} & errorCode=123	Unsuccessful payment (technical reason), do not display order confirmation.
Failed 3D-Secure	Y	Redirect: checkout_id={checkoutId} & authentication_id={authenticationId} & errorCode=140	3DS failed due to technical reason, do not display order confirmation.

Description	3D-Secure Configured	Result	Merchant action
Declined transaction	N	Redirect: checkout_id={checkoutId} & transaction_id={transactionId} & error_code=121	Payment reached the acquirer, but was declined (e.g. Insufficient Funds), do not display order confirmation.
Successful transaction **Test card**: 4000000000001091, expiry: 01/any, cvv: any	N	Redirect: checkout_id={checkoutId} & transaction_id={transactionId}	Display order confirmation.
Successful 3D-Secure and transaction. **Test card**: 4000000000001091, expiry: 01/any, cvv: any	Y	Redirect: checkout_id={checkoutId} & transaction_id={transactionId} & authentication_id={authenticationId}	Display order confirmation.
Customer visits the URL of an already completed Checkout	-	Redirect: checkoutId={checkoutId} & error_code=168	Display corresponding message to customer. Checkout is completed whenever there was a single successful payment processed through it.
Customer visits the URL of an expired Checkout	-	Redirect: checkout_id={checkoutId} & error_code=169	Display corresponding message to customer. Checkout is expired whenever the expiryTime is reached
Customer visits the URL of a Checkout which has reached the maximum of failed payment attempts	-	Redirect: checkout_id={checkoutId} & error_code=165	Display corresponding message to customer. Payments through a single Checkout can be attempted up to 3 times unsuccessfully.
Form validation errors / Google Pay service failures on the Checkout page	-	Displays error alert to Customer on the page	Customer is prompted to correct their form input and retry the payment or try using alternate card or payment method

*** Failed transaction - Depending on which step in the payment process failed, the transaction_id might not always be present as the query parameter