Dragonpay Online Payment

Merchant Payout API

Version 1.12 – 31 Mar 2020

Table of Contents

1. About this Document

This document describes the Application Programming Interface (API) between the Dragonpay Payout (PO) System and the Merchant's e-commerce back-end. The PO is responsible for performing mass payout to multiple recipients. Upon completing the request, the merchant's payout postback url will be invoked. An email summary will also be generated.

If you have any questions please do not hesitate to contact **sales@dragonpay.ph**.

2. Intended Audience

The intended audience for this document is technical personnel or programmers with background knowledge of programming and e-commerce. The examples in this document are written in Microsoft C# .NET. However, the programmer is free to implement the interfaces using other programming languages as long as they conform to Web standards such as HTTP GET, Name-Value Pair, and SOAP/XML Web Services calls.

3. Change Log

Version	Date	Changes
0.25	Oct 12, 2015	Removed enrollment module documentation
0.30	Feb 27, 2016	Added return codes for RequestPayoutEx
0.32	Aug 2, 2016	Updated Appendix 3
0.33	Sept 30, 2016	Added email and mobileNo to RequestPayoutEx
0.34	Jan 12, 2017	Added postback url documentation
0.35	Oct 31, 2017	Added Appendix 5
0.50	Jun 1, 2018	Added (H)old status, GetLedgerBalance, GetTxnDetails; corrected GetTxnStatus
0.60	Aug 1, 2018	Added GetProcessors() documentation
1.00	Dec 29, 2018	Added RequestCashPayout()
1.10	Feb 25, 2019	REST/JSON support added
1.11	Sep 11, 2019	Updated Appendix 3 list
1.12	Mar 31, 2020	Document refresh

4. Introduction

Merchants that need to send payments to multiple recipients are faced with the challenge of writing hundreds, if not thousands, of checks or manually depositing to various bank accounts in different banks. This process is very tedious, takes up a lot of man-hours, and is prone to error.

Mass payout solutions simplify this process by allowing Merchants to just send a list of recipients and amounts, or programmatically request the payout, and it will handle everything behind-the-scenes. The recipient can choose their preferred method of receiving the payment whether through bank transfers, mobile payments, or physical cash pickup.

4.1 How does mass payout work?

The system generally follows this pattern:

1. Merchant sends the payout details (bank id, account no, name, amount) to Dragonpay via SOAP or REST
2. Dragonpay handles the payout and notifies Merchant through http postback or merchant can manually check the progress status through the admin portal

The Dragonpay Payout System will perform the actual payment based on its agreed cut-off times with the Merchant. It is not carried out in real-time. Merchant can also query the payout system in real-time to check on the status of a payout request.

5. Payout API

This section of the document describes the Payout (PO) API in detail, covering the various functions used, as well as, codes that can be used to integrate them.

5.1 System Requirements

In order to integrate with the PO, Merchant must fulfill the following prerequisites:

1. Merchant site must be capable of getting the required data from customer (ex. Name, address, email).
2. Merchant site can send http request data to PO system to request a payout.
3. Optionally, Merchant site may have a Postback URL to accept real-time notifications from PO of status changes or rely on email summaries.

Each Merchant is assigned the following:

- merchant id – unique code identifying the Merchant
- password – a unique password for logging in to admin website
- api key – a unique string that is used for API calls

Although this document uses Microsoft .NET conventions, it should be implementable under other operating environments (ex. Linux, PHP, Perl, Java).

5.2 Message Passing (Merchant -> PO)

This section describes how the Merchant will pass a request to the PO for payout requests and vice versa. There is currently one integration model available –the Web Services Model.

5.2.1 SOAP/XML Web Service Model

The Merchant may choose to implement the API using the XML Web Services model. Under this model, the parameters are exchanged directly between the Organization back-end system and PO servers through SOAP calls.

You may use the following URL's as the Web Service entry point.

Web Service Production URL:

```
https://gw.dragonpay.ph/DragonPayWebService/PayoutService.asmx
```

Web Service Test URL:

```
https://test.dragonpay.ph/DragonPayWebService/PayoutService.asmx
```

5.2.1.1 Requesting a Simple Payout

These are the parameters passed by the Merchant to the PO to request for a simple payout with no enrollment involved.

Web Method: RequestPayoutEx

Parameter	Data Type	Description
apiKey	Varchar(40)	A unique code assigned to Merchant for API
merchantTxnId	Varchar(40)	Unique txnid on the side of merchant referring to this request
userName	Varchar(30)	Account name
amount	Numeric(15,2)	Amount to pay
currency	Char(3)	Currency (currently only PHP is supported)
description	Varchar(180)	Text description as to what this is about
procId	Varchar(4)	Payout channel selected (see Appendix 3)
procDetail	Varchar(40)	Account/mobile no of payout channel
runDate	Date	Date when to execute the payout
email	Varchar(80)	Email address of payout recipient
mobileNo	Varchar(20)	Mobile number of payout recipient

The *RequestPayoutEx()* method will return an integer value of zero (0) if successfully requested. Merchant has to check the Admin portal to see updates on payout status.

Possible return values are listed in Appendix 5.

There is a specialized version of the RequestPayoutEx web service specifically for cash pickup channels.

Web Method: RequestCashPayout

Parameter	Data Type	Description
apiKey	Varchar(40)	A unique code assigned to Merchant for API
merchantTxnId	Varchar(40)	Unique txnid on the side of merchant referring to this request
firstName	Varchar(30)	First name of recipient
middleName	Varchar(30)	Middle name of recipient
lastName	Varchar(30)	Last name of recipient
street1	Varchar(60)	Street address of recipient
street2	Varchar(60)	Additional street address of recipient (optional)
barangay	Varchar(40)	Barangay of recipient's address
city	Varchar(20)	City of recipient's address
province	Varchar(20)	Province of recipient's address
email	Varchar(80)	Email address of recipient
mobileNo	Varchar(20)	Mobile number of recipient
birthDate	Date	Date of birth of recipient (required for TrueMoney)
nationality	Varchar(20)	Nationality of recipient (required for TrueMoney)
amount	Numeric(15,2)	Amount to pay
currency	Char(3)	Currency (currently only PHP is supported)
description	Varchar(180)	Text description as to what this is about
procId	Varchar(4)	Cash payout channel selected (see Appendix 3)
runDate	Date	Date when to execute the payout

The *RequestCashPayout()* method will return an integer value of zero (0) if successfully requested. Merchant has to check the Admin portal to see updates on payout status.

5.2.1.2 Querying Transaction Status

These are the parameters passed by the Merchant to the PO to request for a simple payout with no enrollment involved.

Web Method: GetTxnStatus

Parameter	Data Type	Description
apiKey	Varchar(40)	A unique code assigned to Merchant for API
merchantTxnId	Varchar(40)	Unique txnid on the side of merchant referring to this request

The *GetTxnStatus()* method will respond with a single *status* string:

Parameter	Description
status	The status of the payout transaction. Refer to Appendix 4 for codes.

Aside from the transaction status listed in Appendix 4, *GetTxnStatus()* may also return "E" if the apiKey is invalid; or "U" if the merchantTxnId does not exist.

For more details on error codes due to FAILURE, or reference numbers for SUCCESS or PENDING, please access the web-based administrator page.

5.2.1.3 Checking Ledger Balance

These are the parameters passed by the Merchant to the PO to request for the available balance from the payout ledger.

Web Method: GetLedgerBalance

Parameter	Data Type	Description
apiKey	Varchar(40)	A unique code assigned to Merchant for API

The web method returns a value of type *double* referring to the amount in Philippine Peso (PHP).

5.2.1.4 Getting Available Payout Channels

These are the parameters passed by the Merchant to the PO to request for the available payout channels.

Web Method: GetProcessors

The *GetProcessors()* method expects no parameters and will respond with an array of records with the following structure:

Parameter	Data Type	Description
procId	Varchar(4)	Payout channel selected (see Appendix 3)
shortName	Varchar(15)	Text name of the processor
logo	Varchar(160)	Optional logo
defBillerId	Varchar(80)	Dragonpay source account (internal use only)
status	Char(1)	Channel status (Active or Inactive)
merchantFee	Numeric(10,2)	Fee charged by Dragonpay to Merchant
userFee	Numeric(10,2)	Fee charged by Dragonpay to recipient

5.2.1.5 Retrieving Payout Transaction Details

These are the parameters passed by the Merchant to the PO to request for the details of a Payout transaction.

Web Method: GetTxnDetails

Parameter	Data Type	Description
apiKey	Varchar(40)	A unique code assigned to Merchant for API
merchantTxnId	Varchar(40)	Unique txnid on the side of merchant referring to this request

The *GetTxnDetails()* method will respond with the following structure:

Parameter	Data Type	Description
refNo	Varchar(40)	A unique code assigned to Merchant for API
refDate	Date	Timestamp of the original payout request
merchantId	Varchar(20)	Id of the merchant requesting
merchantTxnId	Varchar(40)	Unique txnid on the side of merchant referring to this request
amount	Numeric(15,2)	Amount to pay
currency	Char(3)	Currency (currently only PHP is supported)
description	Varchar(180)	Text description as to what this is about
userId	Varchar(30)	Account name
status	Char(1)	Transaction status (see Appendix 4)
procId	Varchar(4)	Payout channel selected (see Appendix 3)
procDetail	Varchar(40)	Account/mobile no of payout channel
procMsg	Varchar(180)	Messages for transaction specific to procId
runDate	Date	Date when to execute the payout
settleDate	Date	Timestamp when payout was completed
email	Varchar(80)	Email address of payout recipient
mobileNo	Varchar(20)	Mobile number of payout recipient

5.2.2 REST/JSON Model

The Merchant may choose to implement the API using the REST/JSON model. Under this model, the parameters are exchanged directly between the Organization back-end system and PO servers through standard HTTP GET/POST calls using JSON format.

You may use the following URL's as the REST entry point.

REST Production Base URL:

`https://gw.dragonpay.ph/api/payout/merchant/`

REST Test Base URL:

`https://test.dragonpay.ph/api/payout/merchant/`

(Note: IP address whitelisting is required to use these web api's)

Although this document may use Microsoft .NET conventions, it should be implementable under other operating environments (ex. Linux, PHP, Perl, Java).

Merchant must pass its apiKey through the HTTP header using Authorization Bearer.

```
HttpWebRequest request = (HttpWebRequest)WebRequest.Create(apiUrl);
request.Method = "GET";   // or "POST"
request.Headers.Add("Authorization", "Bearer " + apiKey);
request.ContentType = "application/json";
```

5.2.2.1 Requesting a Simple Payout

These are the parameters passed by the Merchant to the PO to request for a simple payout with no enrollment involved. Merchant must pass the Dragonpay-assigned *merchantid* in the url path.

REST endpoint : v1/{merchantid}/post
HTTP method: POST

Parameter	Data Type	Description
TxnId	Varchar(40)	Unique txnid on the side of merchant referring to this request
FirstName	Varchar(30)	First name of the beneficiary
MiddleName	Varchar(30)	Middle name of the beneficiary
LastName	Varchar(30)	Last name of the beneficiary
Amount	Numeric(15,2)	Amount to pay
Currency	Char(3)	Currency (currently only PHP is supported)
Description	Varchar(180)	Text description as to what this is about
ProcId	Varchar(4)	Payout channel selected (see Appendix 3)
ProcDetail	Varchar(40)	Account/mobile no of payout channel
RunDate	Date	Date when to execute the payout
Email	Varchar(80)	Email address of payout recipient
MobileNo	Varchar(20)	Mobile number of payout recipient
BirthDate	Date	Date of birth of beneficiary (optional for bank)
Nationality	Varchar(20)	Nationality of beneficiary (optional for bank)
Address	UserAddress	Postal address of beneficiary (optional for bank)

The *UserAddress* type has the following structure:

Parameter	Data Type	Description
Street1	Varchar(60)	Street address
Street2	Varchar(60)	Street address
Barangay	Varchar(40)	Barangay name
City	Varchar(20)	City name
Province	Varchar(20)	Province name (ex. Metro Manila)
Country	Char(2)	ISO Country code (ex. PH)

The *Post* method will return an Error 401 if wrong apiKey is provided or the following structure if a valid apiKey is provided:

Parameter	Data Type	Description
Code	Integer	0 if successful, else error. See Appendix 5.
Message	String	Payout Reference No if successful, else error message.

Merchant has to check the Admin portal to see updates on payout status.

Sample JSON body message

```
{
    "TxnId": "20190225008",
    "FirstName": "Robertson",
    "MiddleName": "Sy",
    "LastName": "Chiang",
    "Amount": "1000",
    "Currency": "PHP",
    "Description": "Testing JSON payout",
    "ProcId": "CEBL",
    "ProcDetail": "dick@dragonpay.ph",
    "RunDate": "2019-02-26",
    "Email": "dick@dragonpay.ph",
    "MobileNo": "09175281679",
    "BirthDate": "1970-11-17",
    "Nationality": "Philippines",
    "Address":
    {
       "Street1": "123 Sesame Street",
       "Street2": "Childrens Television Workshop",
       "Barangay": "Ugong",
       "City" : "Pasig",
       "Province": "Metro Manila",
       "Country": "PH"
    }
}
```

Sample successful JSON response

```
{
    "Code": 0,
    "Message": "ABCD1234"
}
```

Sample failed JSON response

```
{
    "Code": -2,
    "Message": "Cannot be voided anymore"
}
```

5.2.2.2 Querying Transaction Status

See Section 5.2.2.5 for retrieving Payout Transaction Details. The *Status* is returned as one of the fields.

5.2.2.3 Checking Ledger Balance

These are the parameters passed by the Merchant to the PO to request for the available balance from the payout ledger. Merchant must pass the Dragonpay-assigned *merchantid* in the url path.

REST endpoint : v1/{merchantid}/balance
HTTP method: GET

The web method returns a value of type *double* referring to the amount in Philippine Peso (PHP).

5.2.2.4 Getting Available Payout Channels

These are the parameters passed by the Merchant to the PO to request for the available payout channels.

REST endpoint : v1/processors
HTTP method: GET

The web method expects no parameters and will respond with an array of records with the following structure in JSON format:

Parameter	Data Type	Description
procId	Varchar(4)	Payout channel selected (see Appendix 3)
shortName	Varchar(15)	Text name of the processor
logo	Varchar(160)	Optional logo
billerId	Varchar(80)	Dragonpay source account (internal use only)
status	Char(1)	Channel status (Active or Inactive)
merchantFee	Numeric(10,2)	Fee charged by Dragonpay to Merchant
userFee	Numeric(10,2)	Fee charged by Dragonpay to recipient

5.2.2.5 Retrieving Payout Transaction Details

These are the parameters passed by the Merchant to the PO to request for the details of a Payout transaction. Merchant must pass the Dragonpay-assigned *merchantid* in the url path along with the unique transaction id (TxnId) passed during the original payout request.

REST endpoint : v1/{merchantid}/{txnid}
HTTP method: GET

The web method will respond with the following structure:

Parameter	Data Type	Description
refNo	Varchar(40)	A unique code assigned to Merchant for API
refDate	Date	Timestamp of the original payout request
merchantId	Varchar(20)	Id of the merchant requesting
merchantTxnId	Varchar(40)	Unique txnid on the side of merchant referring to this request
amount	Numeric(15,2)	Amount to pay
currency	Char(3)	Currency (currently only PHP is supported)
description	Varchar(180)	Text description as to what this is about
userId	Varchar(30)	Account name
status	Char(1)	Transaction status (see Appendix 4)
procId	Varchar(4)	Payout channel selected (see Appendix 3)
procDetail	Varchar(40)	Account/mobile no of payout channel
procMsg	Varchar(180)	Messages for transaction specific to procId
runDate	Date	Date when to execute the payout
settleDate	Date	Timestamp when payout was completed
email	Varchar(80)	Email address of payout recipient
mobileNo	Varchar(20)	Mobile number of payout recipient

5.2.2.6 Voiding Payout Transaction

These are the parameters passed by the Merchant to the PO to request for the voiding of a pending Payout transaction. Merchant must pass the Dragonpay-assigned *merchantid* in the url path along with the unique transaction id (TxnId) passed during the original payout request.

REST endpoint : v1/{merchantid}/{txnid}/void
HTTP method: GET

The web method will respond with the following structure:

Parameter	Data Type	Description
Code	Integer	0 if successfully voided, else error.
Message	String	General message

5.2.2.7 Retrieving Payout Transactions for Date Range

These are the parameters passed by the Merchant to the PO to request for the voiding of a pending Payout transaction. Merchant must pass the Dragonpay-assigned *merchantid* in the url path along with the unique transaction id (TxnId) passed during the original payout request.

REST endpoint : v1/{merchantid}/transactions/{startdate}/{enddate}
HTTP method: GET

(Note: the *startdate* and *enddate* parameters would be in the format *yyyy-mm-dd*)

The web method will respond with an array of records following the transaction detail structure in 5.2.2.5.

5.2.2.8 Retrieving Payout Ledger Transactions for Date Range

These are the parameters passed by the Merchant to the PO to request for the voiding of a pending Payout transaction. Merchant must pass the Dragonpay-assigned *merchantid* in the url path along with the unique transaction id (TxnId) passed during the original payout request.

REST endpoint : v1/{merchantid}/ledger/{startdate}/{enddate}
HTTP method: GET

(Note: the *startdate* and *enddate* parameters would be in the format *yyyy-mm-dd*)

The web method will respond with an array of the following structure:

Parameter	Data Type	Description
merchantId	Varchar(20)	Id of the merchant requesting
refDate	DateTime	Timestamp of the ledger entry
txnType	Varchar(20)	Transaction Type code (see Appendix 6)
description	Varchar(180)	Text description as to what this entry is about
amount	Numeric(15,2)	Amount (+ or - depending on type)
balance	Numeric(15,2)	Running balance after this entry

5.3 Receiving Postback Notice

When payment processing has completed, the Payout System will invoke the Merchant's registered postback URL's via HTTP GET and pass along the parameters below.

Parameter	Description
refNo	A common reference number identifying this specific transaction from the PS side
merchantTxnId	A unique id identifying this specific transaction from the merchant side
status	The result of the payment. Refer to Appendix 4 for codes.
message	Additional payment processing information
digest	A sha1 checksum digest of the parameters along with the merchant pwd.

An HTTP GET from the Payout System may look something like this:

```
http://www.abcstore.com/Postback.aspx?refNo=ABCD1234&merchantTxnId=1234&
status=S&message=72843747212&digest=a4b3d08462......
```

The digest is computed using the SHA1 algorithm. Below is a sample code showing how to generate the SHA1 digest using C# .NET:

```
String digest = GetSHA1Digest(String.Format("{0}:{1}:{2}:{3}:{4}",
       Request["merchanttxnid"].ToString(),
       Request["refno"].ToString(),
       Request["status"].ToString(),
       Request["message"].ToString(),
       Application["merchantPwd"].ToString()));
```

Below is a sample implementation of SHA1 using C# .NET:

```
public static string GetSHA1Digest(string message)
{
       byte[] data = System.Text.Encoding.ASCII.GetBytes(message);

       System.Security.Cryptography.SHA1 sha1 = new
              System.Security.Cryptography.SHA1CryptoServiceProvider();
       byte[] result = sha1.ComputeHash(data);

       System.Text.StringBuilder sb = new System.Text.StringBuilder();
       for(int i=0; i<result.Length; i++)
              sb.Append(result[i].ToString("X2"));

       return sb.ToString().ToLower();
}
```

Appendix 1 – Currency Codes

Code	Description
PHP	Philippine Peso
USD	US Dollar

Appendix 2 – Error Codes

Code	Description
000	Success
102	Incorrect secret key
103	Invalid reference number
104	Unauthorized access
106	Currency not supported
107	Transaction cancelled
108	Insufficient funds
109	Transaction limit exceeded
110	Error in operation
111	Invalid parameters
201	Invalid Merchant Id
202	Invalid Merchant Password

Appendix 3 – Payout Processor Codes

Code	Description
AUB	Asia United Bank CA/SA
BDO	Banco de Oro CA/SA
BPI	BPI CA/SA
BFB	BPI Family Bank
CBC	Chinabank CA/SA
EWB	EastWest CA/SA
LBP	Landbank CA/SA
MBTC	Metrobank CA/SA
PNB	PNB individual CA/SA
RCBC	RCBC CA/SA, RCBC Savings Bank CA/SA, RCBC MyWallet
RSB	Robinsons Bank CA/SA
SBC	Security Bank CA/SA
UBP	Unionbank CA/SA, EON
UCPB	UCPB CA/SA
CEBL	Cebuana Lhuillier Cash Pick-up
LBC	LBC Cash Pick-up
PLWN	Palawan Pawnshop Cash Pick-up (reserved)
PRHB	PeraHub Cash Pick-up
RCBP	RCBC/RCBC Savings Bank Cash Pick-up (reserved)
RDP	RD Pawnshop Cash Pickup (reserved)
TRMY	TrueMoney Cash Pick-up (reserved)
BITC	Coins.ph Wallet (reserved)
GCSH	Gcash
SMRT	Smart Money (reserved)
MAY	Maybank
SBA	Sterling Bank of Asia
DBP	Development Bank of the Philippines (reserved)
PBCM	Philippine Bank of Communications
PSB	Philippine Savings Bank
PVB	Philippine Veterans Bank
BOC	Bank of Commerce
CBCS	Chinabank Savings Bank
CTBC	Chinatrust
PYMY	Smart PayMaya

Appendix 4 – Payout Transaction Status Codes

Code	Description
S	Successfully completed
F	Failed
P	Pending
H	On hold
G	In progress
V	Voided

Appendix 5 – RequestPayoutEx Result Codes

Code	Description
0	Successfully completed
-1	General error
-2	(reserved)
-3	(reserved)
-4	Unable to create a payout transaction
-5	Invalid payout account details
-6	Cannot accept a pre-dated run date
-7	Amount limited exceeded
-8	Similar transaction id already exists
-9	Server IP access is not allowed
-10	Payout account is blacklisted
-11	Payout account is not enrolled for bank
-12	Invalid API Key

Appendix 6 – Payout Transaction Type Codes

Code	Description
A	Adjusting Entry
P	Payout
T	Top-up
F	Service Fee