API Introduction

The LianLian API is organized around REST. Our API has predictable resource-oriented URLs, accepts JSON-encoded request bodies, returns JSON-encoded responses, and uses standard HTTP response codes, authentication, and verbs.

Versioning

When we make backwards-incompatible changes to the API, we release new versions. The version to use is specified in the URL. The current version for the Payments API is v1 such as:

```
https://api....com/payments/v1/...
```

Authentication

The LianLian API uses an API key to authenticate requests. You will receive your `token` from LianLian Global. The `Authorization` header must be included in all requests with the following format:

```
Authorization: Basic {token}
```

Your API keys carry many privileges, so be sure to keep them secure! Do not share your secret API keys in publicly accessible areas such as GitHub, client-side code, and so forth.

All API requests must be made over HTTPS. Calls made over plain HTTP will fail. API requests without authentication will fail.

Preventing replay attacks

A replay attack is when an attacker intercepts a valid payload and its signature, then re-transmits them. To mitigate such attacks, LianLian includes a timestamp in the **LLPAY-Signature** header for requests and for responses. Because this timestamp is part of the signed payload, it is also verified by the signature, so an attacker cannot change the timestamp without invalidating the signature. If the signature is valid but the timestamp is too old, the call will be rejected.

Verifying signatures

The LLPAY-Signature header contains a timestamp and one or more signatures. The timestamp is prefixed by t=, and each signature is prefixed by a scheme. Schemes start with v, potentially followed by an integer. Currently, the only valid signature scheme is v. When the signature changes it will move to v1, v2, ...

```
LLPAY-Signature: t=1492774577,
v=5257a869e7ecebeda32affa62cdca3fa51cad7e77a0e56ff536d0ce8e108d8bd
```

Note that newlines have been added in the example above for clarity, but a real LLPAY-Signature header will be all one line.

Lianlian Pay and clients shall exchange each party’s public key. The LLPAY-Signature will be signed by with the RSA private key.

Step 1: Determine the signature payload

Create a `payload` string which is the following separated by `&`

- HTTP_METHOD: The actual method defined for specific API endpoint `POST`
- URI: The request's URL Path, URI. `/api/mkt/balance`
- REQUEST_EPOCH: The seconds elapsed since 1970/1/1 00:00:00 GMT same as value of `t`
- REQUEST_PAYLOAD: The contents of request body, typically in JSON format `{"currency":"USD"}`
- QUERY_STRING: The query string of the URL, URL Encoded. `attr1%3Dvalue1%26attr2%3Dvalue2`

Put typical GET and POST examples here.

Example POST `payload` (Note there is no trailing `&` when no query string is present):

```
POST&payments/v1/merchants&19879234&{"currency":"USD"}
```

Example GET `payload`:

```
GET&payments/v1/payments/602837&19879234&&amp;currency%3DUSD
```

Step 2: Prepare the `LLPAY-Signature` string

You achieve this by concatenating:

- The timestamp (Seconds elapsed since 1970/1/1 00:00:00 GMT as a string)
- The character `,`
- Use your RSA private key and calculate the string `BASE64(RSAwithSHA256(payload))` from Step 1

Result Signature

LianLian result headers also contain the `LLPAY-Signature` header as described above but signed with Lian Lian Global's private RSA key. You may verify the signature corresponds to the Result body by calculating the signature and comparing it to the unencrypted version of `LLPAY-Signature` using the Lian Lian Global public RSA key. Here are the steps:

Step 1: Determine the signature `payload`

Create a `payload` string which is the following separated by `&`

- Response Timestamp: Extract t from LLPay-Signature and use it as timestamp
- Response Payload: The contents of response body, typically in JSON format `{"currency":"USD"}`

Example `payload`:

```
19879234&{"currency":"USD"}
```

Step 2: Use RSA Verify to compare signatures

Compare the LLPay-Signature received with the `payload` calculated in Step 1 using the Lian Lian Global Public Key using the following algorithm:

```
RSA.verify( LLPAY-Signature,
            '19879234&{" currency":"USD"}',
            Lian Lian Global Public Key )
```

Results

Success

A successful API call returns a conventional HTTP response with status 2xx indicating success. The result objects are standard REST resources. Example:

```
{
  "name": "Merchant Name",
  ...
}
```

or

```
[
  {
    "name": "Merchant Name",
    ...
  },
  {
    "name": "Merchant Name 2",
    ...
  }
]
```

Errors

LianLian uses conventional HTTP response codes to indicate the success or failure of an API request. In general: Codes in the 2xx range indicate success. Codes in the 4xx range indicate an error that failed given the information provided (e.g., a required parameter was omitted, a charge failed, etc.). Codes in the 5xx range indicate an error with LianLian's servers (these are rare).

Some 4xx errors that could be handled programmatically (e.g., a validation error) include an error code that briefly explains the error reported.

HTTP Status Table

CODE	DESCRIPTION
200	Success.
400	API Error. Detailed error codes & error message are included in each API method.
401	Unauthorized. No valid API key provided.
403	Forbidden. The API key doesn't have permissions to perform the request.
404	Resource Not Found. The requested resource doesn't exist.
500	Internal Server Error. Contact Lianlian Pay for technical support

Error Result

When a result other than 2xx is returned, an Error Object is returned with further information including Error Code and Message. There are some general error codes that can happen across APIs but most are particular to the API and defined within that API's documentation.

See definition of Error Model in API Reference.

Body

```
{
  "code": "154008",
  "message": "Client has insufficient balance to pay merchant"
 }
```

Request IDs

Each API request has an associated request identifier. You can find this value in the response headers, under `Request-Id`. If you need to contact us about a specific request, providing the request identifier will ensure the fastest possible resolution.

Argument and Field Naming

LianLian arguments and object field names follow standard underscore naming such as:

```
https://api...com/resource/?filter_by="filter"
```

```
{
  "store_name": "My Store",
  "kyc_status": "success"
}
```

Payments (1.0)

Download OpenAPI specification: Download

David Dawson: david.dawson@lianlian.com

The Lianlian Global Payments API is designed for Marketplaces to link their accounts with Lianlian merchant accounts and send them cross currency payments.

Authentication

Balance

Get Client Balance

Returns Total balance of client's account within LianLian in one or more currencies. Not specifying a currency will return the default currencies configured for client.

Typically this is USD & CNY.

QUERY PARAMETERS

`currency`	string

Repeat parameter to specify one 3 letter currency codes to return balance in.

Responses

200 OK

RESPONSE SCHEMA: application/json

`Array [`

`currency` `required`	string 3 letter currency code
`balance`	string Total balance for this currency

`]`

400 Codes:

Code	Description	Client Action	LianLian Action
154011	Currency not supported	Client validates Currency used in request. Supported currency: **USD**, **CNY**	Waits for a new request
999995	Invalid Parameter	Client updates request body values according to error code returned by Lianlian.	Waits for a new request

RESPONSE SCHEMA: application/json

`code` `required`	string Error code as described per API
`message` `required`	string Human readable message of error

GET /balance

Response samples

200 400

Content type

application/json

Copy Expand all Collapse all

```
[
  - {
        "currency": "USD",
        "balance": "4182.28"
    },
  - {
        "currency": "CNY",
        "balance": "78.99"
    }
]
```

Merchants

Start New Merchant Registration

Starts asynchronous process of registering a new merchant with LianLian.

On successful return, if `registration_complete` is `false`, platform shall use `registration_url` to redirect the merchant to in order to begin the LianLian Registration

Experience.

When registration is successfully completed the user will be redirected to the passed in `redirect_url`

REQUEST BODY SCHEMA: application/json

`store_name` required	string Store name as known by the calling platform
`redirect_url` required	string URL Lianlian will redirect to at the end of a successful registration process.
`shop_url`	string URL of merchant store if URL can't be algorithmically produced from merchant_id
`note`	string Note to be used by customer support if needed
`merchant_client_id` required	string Merchant ID as known by the client platform

Responses

200 OK

RESPONSE SCHEMA: application/json

`id`	string LianLian merchant ID. Merchant shall use this ID to contact LianLian for any payment issues. Non-existant if `registraton_complete` is false.
`merchant_client_id` required	string Client's ID for this Merchant
`status` required	string (| Registration Status Enumeration) Enum: `"need_info"` `"success"` `"blocked"` `"ongoing"`

State of merchant registration process. Will be NULL for brand new merchant that hasn't started registration process.

One of:

Value	Description
ongoing	Registration ongoing
need_info	Merchant must communicate with Lianlian
success	Ready for delivery money to merchant
blocked	User has been blacklisted from receiving payments

`registration_url`

string
URL to redirect merchant to in order to start registration on LianLian platform.

Non-existent if `registration_complete` is true.

`shop_url`

string
URL of Merchant's shop. For some clients this URL is auto-generated from `client_id`

`note`

string
Human readable description of Merchant

`registration_complete`
`required`

boolean
State of merchant registration process. One of:

Value	Description
true	Merchant registration complete, platform can issue payment to this merchant
false	Merchant registration not complete yet, platform cannot issue payment to this merchant

400 Codes:

Code	Description	Client Action	LianLian Action
999995	Invalid Parameter	Client updates request body values according to error code returned by Lianlian.	Waits for a new request

RESPONSE SCHEMA: application/json

`code` required	string Error code as described per API
`message` required	string Human readable message of error

POST `/merchants`

Request samples

Payload

Content type
application/json

Copy Expand all Collapse all

```json
{
    "store_name": "string",
    "redirect_url": "string",
    "shop_url": "string",
    "note": "string",
    "merchant_client_id": "string"
}
```

Response samples

200 400

Content type

application/json

Example

Example 1

Copy Expand all Collapse all

```
{
    "shop_url": "https://www.wish.com/merchant/wish001",
    "note": "rft",
    "registration_complete": true,
    "status": "success",
    "id": "8866003800000564",
    "merchant_client_id": "wish001"
}
```

Get Merchant

Get information about the merchant - in particular their status on the LianLian platform.

PATH PARAMETERS

`merchant_client_id` required	string

Responses

200 OK

RESPONSE SCHEMA: application/json

`id`

string
LianLian merchant ID. Merchant shall use this ID to contact LianLian for any payment issues.

Non-existant if `registraton_complete` is false.

`merchant_client_id`
required

string
Client's ID for this Merchant

`status`
required

string (| Registration Status Enumeration)
Enum: `"need_info"` `"success"` `"blocked"` `"ongoing"`
State of merchant registration process. Will be NULL for brand new merchant that hasn't started registration process.

One of:

Value	Description
ongoing	Registration ongoing
need_info	Merchant must communicate with Lianlian
success	Ready for delivery money to merchant
blocked	User has been blacklisted from receiving payments

`registration_url`

string
URL to redirect merchant to in order to start registration on LianLian platform.

Non-existent if `registration_complete` is true.

`shop_url`

string
URL of Merchant's shop. For some clients this URL is auto-generated from `client_id`

`note`

string
Human readable description of Merchant

`registration_complete`
required

boolean
State of merchant registration process. One of:

Value	Description

true	Merchant registration complete, platform can issue payment to this merchant
false	Merchant registration not complete yet, platform cannot issue payment to this merchant

— **404** Not Found

GET `/merchants/merchant_client_id/{merchant_client_id}`

Response samples

200

Content type

application/json

Copy Expand all Collapse all

```
{
    "shop_url": "https://www.wish.com/merchant/wish001",
    "registration_complete": true,
    "id": "8866003800000564",
    "status": "success",
    "merchant_client_id": "wish001"
}
```

Payments

Send Payment to Merchant

Initiates a payment to specific merchant.

This call is Idempotent if sent with the same payment_client_id and last payment request was successfull. In this case the additional HTTP Response Header `Request-Repeated: true` will be returned. If the original request failed, a new payment request will retry the payment.

Steps within API:

- **Money Debited** from client's account can fail from insufficient funds
- **Verify Merchant Registration** information is valid If Merchant is unknown, call fails If Merchant status is `blocked`, call fails
- **Send Money** to merchant If any issues arise, LianLian customer support helps fix issues and finalize transfer

This call returns HTTP Status 200 on success and returns the payment object including Lianlian internal payment status. Success guarantees the money has been debited from the client account not that the actual payment has completed as it takes too long to be done synchronously and will be completed within a few days typically. It does guarantee, however, that the payment will eventually happen.

On failure the API returns HTTP Status 400 with an Error object specifying the error code why the payment failed.

REQUEST BODY SCHEMA: application/json

`merchant_client_id` required	string Client's Merchant ID to make payment to.
`payment_client_id` required	string Client's Payment ID for this payment request. **Must be unique and is usually a UUID**
`currency` required	string Enum: `"CNY"` `"USD"` 3 letter code of payment currency
`amount` required	string Amount to pay merchant
`note`	string

Additional information to help track payment

Responses

200 OK

RESPONSE SCHEMA: application/json

`id`
required

string
LianLianment ID

`payment_client_id`
required

string
Client's Payment ID for this payment request. **Must be unique and is usually a UUID**

`merchant_id`
required

string
Lianlian's Merchant ID receiving payment

`merchant_client_id`
required

string
Client's Merchant ID receiving payment

`status`

string (| Payment Status Enumeration)
Enum: `"need_info"` `"success"` `"ongoing"`
This status indicates the stage payment is at within Lianlian systems. One of:

Value	Description
ongoing	Payment ongoing normally
need_info	Merchant must communicate with Lianlian
success	Payment complete

`currency`
required

string
Enum: `"CNY"` `"USD"`
3 letter code of currency used

`amount`

string

required

Amount of payment in `currency`

note

string

Human readable note passed into POST /payment

400 Error - payment not processed

Code	Message	Client Action	LianLian Action
154008	Client has insufficient balance to pay merchant	Client to wire more funds to LianLian account	Waits for funds
154011	Currency not supported	Client validates Currency used in request. Supported currencies: **USD**, **CNY**	Waits for a new request
154016	Merchant is blocked from receiving payments	Suggest merchant to contact LianLian for details	Is typically having offline discussion with merchant
999995	Invalid Parameter	Client updates request body values according to error code returned by LianLian.	Waits for a new request

RESPONSE SCHEMA: application/json

code
required

string

Error code as described per API

message
required

string

Human readable message of error

POST /payments

Request samples

Payload

Content type

application/json

Copy Expand all Collapse all

```
{
    "merchant_client_id": "string",
    "payment_client_id": "string",
    "currency": "CNY",
    "amount": "string",
    "note": "string"
}
```

Response samples

200 400

Content type

application/json

Copy Expand all Collapse all

```
{
    "status": "ongoing",
    "currency": "USD",
    "amount": "123.30",
    "note": "USD",
    "merchant_id": "8866003600000853",
    "id": "2020012026476843",
    "payment_client_id": "wish001",
    "merchant_client_id": "test1021"
}
```

Get Payment

The API returns HTTP Status 200 with the related payment object if the payment was successful.

The API returns HTTP Status 404 if the `POST /payments` call returned an error as the **payment_client_id** is not in the system and not recognized.

PATH PARAMETERS

`payment_client_id` required	string

Responses

200 OK

RESPONSE SCHEMA: application/json

`id` required	string LianLianment ID
`payment_client_id` required	string Client's Payment ID for this payment request. **Must be unique and is usually a UUID**
`merchant_id` required	string Lianlian's Merchant ID receiving payment
`merchant_client_id` required	string Client's Merchant ID receiving payment
`status`	string (| Payment Status Enumeration) Enum: `"need_info"` `"success"` `"ongoing"` This status indicates the stage payment is at within Lianlian systems. One of:

Value	Description
ongoing	Payment ongoing normally

need_info	Merchant must communicate with Lianlian
success	Payment complete

`currency` required	string Enum: `"CNY"` `"USD"` 3 letter code of currency used
`amount` required	string Amount of payment in `currency`
`note`	string Human readable note passed into POST /payment

— 404 Not Found

GET `/payments/payment_client_id/{payment_client_id}`

Response samples

200

Content type

application/json

Copy Expand all Collapse all

```
{
    "status": "success",
    "currency": "USD",
    "amount": "123.30",
    "note": "rft",
    "merchant_id": "8866003600000853",
    "id": "2020011923354002",
    "payment_client_id": "wish001",

    "merchant client id": "test1021"
```

```
           _     _
}
```