Test API Server

URL: <https://api.gplatform.org/sol/api>

Requests are made over HTTP (HTTPS) protocol version 1.1. Regardless of the method called by the API, the HTTP request type is always POST. In the body of the request, data is transmitted in json format (encoded in UTF-8).

The request header must contain:

1. Host - address and port where the request goes
2. Content-Type: application / json; charset = utf-8
3. Cache-Control: no-cache
4. Pragma: no-cache
5. Content-Length: the length of the transmitted data

The request body must contain:

1. Json data

              Sample request data:

{

  "method": "find\_client",

  "data": {

              "client\_id": "test\_id\_1234",

              "is\_create\_if\_not\_found": true

  }

}

Sample response data:

{

  "@": "57bcfaee-bc78-49bc-b19e-9be9610c78ab",

  "result": {

              "@": "636f424a-4af6-48df-a0fe-1ea09a7b94e0",

              "res": "OK",

              "msg": ""

  },

  "data": {

              "@": "6ae53969-df92-47c5-a0b5-9ca4e85e03b3",

              "id": "test\_id\_1234",

              "create\_ts": 0,

              "name": "",

              "balance": {

                "@": "db877116-caef-42e2-a1f3-81123b72cf2e",

                "total": 0,

                "unlocked": 0,

                "locked": 0

              }

  }

}

Each request, regardless of the method being called, contains the following set of fields:

1. **method** - the name of the method to invoke (required)
2. **sid** - session identifier (only if it is explicitly specified that the method requires a valid session).
3. **data** - transmitted data. The set of specific fields depends on which method is called.

              In response, you may receive:

1. Error at the HTTP protocol level (anything other than code 200). There may be errors related only to the HTTP transport layer (for example, an error in the request format or server errors).
2. Successful response to the request at the HTTP protocol level (code 200). In this case, you need to check the result code at the application protocol level.

Each response, regardless of the method called, contains the following set of required

fields:

1. **result** - a structure describing the result. May contain fields:
   1. res - enum of the request result code (possible codes are listed below).
   2. msg - textual explanation of the query result (useful for debugging).
2. **data** - response data. The format depends on which method is called.

              The "@" fields can be ignored, it is part of the serialization / deserialization type identification system.

## Query Result Codes

1. OK - successful request
2. FAILED - error
3. UNKNOWN\_METHOD - unknown method
4. INVALID\_SID - invalid session identifier
5. DB\_ENTITY\_NOT\_FOUND - the requested item was not found

## Methods for cash register emulator

### Search for a client by id (address, public key)

**Name** : find\_client

**Requires Session** : No

**Description** : search for a client by id (address, public key) and if not found, then register a new one.

**Request parameters** :

1. client\_id - client identifier
2. is\_create\_if\_not\_found - flag, if true then a new client will be created automatically.

Request example:   
{

  "method": "find\_client",

  "sid": "",

  "ts": 0,

  "args": null,

  "data": {

              "client\_id": "test\_id\_1234",

              "is\_create\_if\_not\_found": true

  }

}

**Response parameters** :

1. id - client identifier
2. create\_ts - creation time in unix timestamp (seconds)
3. name - name
4. balance - balance data:
   1. total - total (unlocked + locked). In pennies
   2. unlocked - can be spent
   3. locked - cannot be spent
5. is\_on\_chain - whether an account has been created on the blockchain

Sample answer:

{

  "@": "57bcfaee-bc78-49bc-b19e-9be9610c78ab",

  "result": {

              "@": "636f424a-4af6-48df-a0fe-1ea09a7b94e0",

              "res": "OK",

              "msg": ""

  },

  "data": {

              "@": "6ae53969-df92-47c5-a0b5-9ca4e85e03b3",

              "id": " 9u1uhp9M6pttQv6t2YY8WJxyjDdKhM4z4X9UY636xdqu ",

              "create\_ts": 0,

              "name": "",

              "balance": {

                "@": "db877116-caef-42e2-a1f3-81123b72cf2e",

                "total": 0,

                "unlocked": 0,

                "locked": 0

              },

              " is\_on\_chain ": false

  }

}

### Making a purchase transaction

**Name** : make\_buy\_tx

**Requires Session** : No

**Description** : Creates a purchase transaction. E If the Unknown client\_id, then the transaction is simply written in the database. If client\_id is specified, then bonuses will be credited to him based on the 5% rule for each item. Also, if spend and spend\_sign are specified, then bonus payments are possible.

**Request parameters** :

1. client\_id - client identifier
2. items - commodity items (composition of the receipt)
   1. id - sequential number 0, 1, 2….
   2. article - article
   3. name - name
   4. price - price per item in kopecks
   5. qty - number of pieces
3. spend - the amount of bonuses to be written off
4. spend\_sign - signature to be charged
5. commit - the flag for committing the transaction. If false, then the transaction will be calculated, but no record will occur in the database.

Example request:

{

  "method": "make\_buy\_tx",

  "sid": "",

  "ts": 0,

  "args": null,

  "data": {

              "client\_id": "9u1uhp9M6pttQv6t2YY8WJxyjDdKhM4z4X9UY636xdqu",

              "items": [

                {

                  "id": 0,

                  "article": "xxBZWDI4",

                  "name": "Test",

                  "price": 4137,

                  "qty": 5

                },

                {

                  "id": 1,

                  "article": "IgEQ83hH",

                  "name": "Test",

                  "price": 6028,

                  "qty": 4

                },

                {

                  "id": 2,

                  "article": "CkRpAamD",

                  "name": "Test",

                  "price": 1956,

                  "qty": 5

                },

                {

                  "id": 3,

                  "article": "KDF5Lexy",

                  "name": "Test",

                  "price": 229,

                  "qty": 3

                }

              ],

              "spend": 0,

              "spend\_sign": "",

              " new\_account\_sign ": "",

              "commit": "false"

  }

}

**Response parameters** :

1. id - transaction identifier (uuid)
2. items - commodity items (composition of the receipt)
   1. id - sequential number 0, 1, 2….
   2. article - article
   3. name - name
   4. price - price per item in kopecks
   5. qty - number of pieces
3. create\_ts - creation time in unix timestamp (seconds)
4. client\_id - client identifier
5. total - total amount of the check (in kopecks)
6. earned - bonuses accrued (in kopecks)
7. spend - bonuses written off (in kopecks)
8. spend\_sign - signature to be charged
9. new\_account\_sign - signature for creating a new account (required if is\_on\_chain == false).
10. spend\_raw\_tx is a raw transaction that needs to be signed (for withdrawal). The field is filled if spend> 0 and commit = false
11. new\_account\_raw\_tx - "raw" transaction to be signed (to create a new account on the blockchain). The field is filled if is\_on\_chain = false 0 and commit = false

Sample answer:

{

  "@": "0c0ef560-3650-42d6-9ae3-c97af0bd419d",

  "result": {

              "@": "636f424a-4af6-48df-a0fe-1ea09a7b94e0",

              "res": "OK",

              "msg": ""

  },

  "data": {

              "@": "8572b0d0-a63e-459c-b308-c879298a454e",

              "id": "d8d30ec3-e31b-5ff0-06c5-3d624be6fc9a",

              "items": [

                {

                  "@": "b616772b-113f-4909-a2f6-7552a36504c1",

                  "id": 0,

                  "article": "xxBZWDI4",

                  "name": "Test",

                  "price": 4137,

                  "qty": 5

                },

                {

                  "@": "b616772b-113f-4909-a2f6-7552a36504c1",

                  "id": 1,

                  "article": "IgEQ83hH",

                  "name": "Test",

                  "price": 6028,

                  "qty": 4

                },

                {

                  "@": "b616772b-113f-4909-a2f6-7552a36504c1",

                  "id": 2,

                  "article": "CkRpAamD",

                  "name": "Test",

                  "price": 1956,

                  "qty": 5

                },

                {

                  "@": "b616772b-113f-4909-a2f6-7552a36504c1",

                  "id": 3,

                  "article": "KDF5Lexy",

                  "name": "Test",

                  "price": 229,

                  "qty": 3

                }

              ],

              "create\_ts": 1614091193,

              "client\_id": "test\_id\_1234",

              "total": 55264,

              "earned": 2762,

              "spend": 0,

              "spend\_sign": "",

"Spend\_raw\_tx": "VjNwM1ozcEJRbmxvY0VGaFp6bG9PVFF5ZEhWTVVWb3lVM2xPTXpCd2FVTkpSbEpVTXpGaVZqRmlUbkpGUmtWeWRsWnBaMkpTUWt4R1ZURnlZVTFGVkdJME1qSm9iVWRtU2xSSVozVjRVRE0yY1VWcFdFWXpOVGd5T0c0NFdIaEJZelpyYzNsb09HRkVjek5ETWxoUVRHcFdSbTFPUjBNNU1W",

" new\_account\_raw\_tx ": ""

  }

}