# JCC\_EXCHANGE DOCUMENT

**[一、installtion 1](#_Toc26261258)**

**[二、usage 2](#_Toc26261259)**

**[1 Signature 2](#_Toc26261260)**

[1.1 Example 2](#_Toc26261261)

**[2 Init instance 4](#_Toc26261262)**

[2.1 Example 4](#_Toc26261263)

**[3 Create order 4](#_Toc26261264)**

[3.1 Example 4](#_Toc26261265)

**[4 Cancel order 5](#_Toc26261266)**

[4.1 Example 5](#_Toc26261267)

**[5 Transfer 6](#_Toc26261268)**

[5.1 Example 6](#_Toc26261269)

**[6 Set brokerage 7](#_Toc26261270)**

[6.1 Example 7](#_Toc26261271)

## Installtion

Github source: <https://github.com/JCCDex/jcc_exchange>

npm install jcc\_exchange

## 二、Usage

### 1 Signature

#### 1.1 Example

const jingtumSignTx = require("jcc\_jingtum\_lib/src/local\_sign");

const copyTx = Object.assign({}, tx);

const swtcSequence = require("jcc\_exchange/lib/util").swtcSequence;

const sequence = await swtcSequence.get(JCCExchange.getSequence, tx.Account);

copyTx.Sequence = sequence;

const sign = jingtumSignTx(copyTx, { seed: secret });

Parameters：

|  |  |  |  |
| --- | --- | --- | --- |
| Parameters | | Type | Description |
| secret | | String | Wallet secret |
| tx | | Object | Transaction Object |
|  | Account | String | Wallet address |
| Sequence | Number | Transaction sequence (from getSequence) |
| Fee | Number | Transaction fee(gas) |
| Flags | Number | Transaction flag |
| Platform | String | Platform address for service charge,default value is jDXCeSHSpZ9LiX6ihckWaYDeDt5hFrdTto(Only existent when create order) |
| TakerGets | Object or String | Paid token and value (Only existent when create order) |
| TakerPays | Object or String | Got token and value (Only existent when create order) |
| OfferSequence | Number | Order sequence (Only existent when cancel order) |
| Amount | Objector String | Token amount (Only existent when transfer or set brokerage) |
|  | currency | String | Token name(case sensitive) |
| value | String | Transfer amount |
| issuer | String | Issuer address（default value is jGa9J9TkqtBcUoHe2zqhVFFbgUVED6o9or） |
|  | FeeAccountID | String | Address for receiving service charge（jingtum address），need be active(Only existent when set brokerage) |
|  | OfferFeeRateNum | Number | Fee numerator(Only existent when set brokerage) |
|  | OfferFeeRateDen | Number | Fee denominator(Only existent when set brokerage) |
|  | Destination | String | Destination address(Only existent when transfer) |
|  | Memos | Array or String | Memos(Only existent when transfer) |
|  | TransactionType | String | Transaction Type: OfferCreate(create) OfferCancel(cancel)  Payment(transfer)  Brokerage(set brokerage) |

Example：(Create order)

const tx = {

Flags: 0,

Fee: 0.00001,

Sequence:0,

Account: 'jpgWGpfHz8GxqUjz5nb6ej8eZJQtiF6KhH',

TransactionType: 'OfferCreate',

TakerGets: {

value: '0.03',

currency: 'CNY',

issuer: 'jGa9J9TkqtBcUoHe2zqhVFFbgUVED6o9or'

},

TakerPays: '1',

Platform: "jDXCeSHSpZ9LiX6ihckWaYDeDt5hFrdTto"

}

Result：

12000722000000006440000000000F424065D40AA87BEE538000000000000000000000000000434E590000000000A582E432BFC48EEDEF852C814EC57F3CD2D4159668400000000000000A732102C13075B18C87A032226CE383AEFD748D7BB719E02CD7F5A8C1F2C7562DE7C12A74473045022100DDAAA2406910BCDF7355ABDBFC8863099BA15759565E771C0E4396E37843880D02203D296D5BAB6966545E9AE5F809D149B2A33EE29551A9F3AC7F61CBD40D94BD4D81141270C5BE503A3A22B506457C0FEC97633B44F7DD8D14896E3F7353697ECE52645D9C502F08BB2EDC5717

Result parameters：

|  |  |  |
| --- | --- | --- |
| Parameters | Type | Description |
|  | String | Signature data |

### 

### 2 Init instance

#### 2.1 Example

const hosts = ["localhost"];

const port = 80;

const https = false;

const retry = 3;

JCCExchange.init(hosts, port, https, retry);

Parameters：

hosts：list of host

port：http port

https: if true means is https, otherwise http

retry: retry number, default 3

### 3 Create order

#### 3.1 Example

const address = "";

const secret = "";

const amount = "1";

const base = "jjcc";

const counter = "swt";

const sum = "1";

const type = "buy";

const platform = "";

const issuer;

const hash = await JCCExchange.createOrder(address, secret, amount, base, counter, sum, type, platform, issuer);

Parameters：

|  |  |  |
| --- | --- | --- |
| Paramters | Type | Description |
| address | String | Wallet address |
| secret | String | Wallet secret |
| amount | String | Order amount |
| base | String | Token name(case insensitive) |
| counter | String | Token name(case insentive)(if buy,counter is paid token;if buy,counter is got token) |
| sum | String | Order total price |
| type | String | Order type(“buy”or “sell”) |
| platform | String | Platform address; default value is jDXCeSHSpZ9LiX6ihckWaYDeDt5hFrdTto |
| issuer | String | Issuer address，default value is jGa9J9TkqtBcUoHe2zqhVFFbgUVED6o9or |

Result：

hash:"9F1D72707EFBA863B8DAD487857E4D7B4E54E90CF6348BBEA5F32509C4390DE4"

Paramters：

|  |  |  |
| --- | --- | --- |
| Parameters | Type | Description |
| hash | String | Transaction hash |

### 4 Cancel order

#### 4.1 Example

const address = "";

const secret = "";

const orderSequence = 0;

const hash = await JCCExchange.cancelOrder(address, secret, orderSequence);

Parameters：

|  |  |  |
| --- | --- | --- |
| Parameters | Type | Description |
| address | String | Wallet address |
| secret | String | Wallet secret |
| orderSequence | Number | Order sequence |

result：

hash:"6E3D45E77EB20D73439D10958AF28A648DC49302FEE1212BB4D5E010624F554D"

Result parameters：

|  |  |  |
| --- | --- | --- |
| Parameters | Type | Description |
| hash | String | Transaction hash |

### 5 Transfer

#### 5.1 Example

const address = "jpgWGpfHz8GxqUjz5nb6ej8eZJQtiF6KhH";

const secret = "";

const amount = "1";

const memo = "test";

const to = "jKTtq57iqHoHg3cP7Rryzug9Q2puLX1kHh";

const token = "jjcc";

const issuer;

const hash = await JCCExchange.transfer(address, secret, amount, memo, to, token, issuer);

Paramters：

|  |  |  |
| --- | --- | --- |
| Paramters | Type | Description |
| address | String | Wallet address |
| secret | String | Wallet secret |
| amount | String | amount |
| memo | String | Memo |
| to | String | Destination address |
| token | String | Token name(case insensitive) |
| issuer | String | Issuer address; default value is jGa9J9TkqtBcUoHe2zqhVFFbgUVED6o9or |

Result：

hash:"445821D60F9294E4ACF911D95166320D2F8EFFC6B4C6C133701B53272684D166"

Paramters：

|  |  |  |
| --- | --- | --- |
| Parameters | Type | Description |
| hash | String | Transaction hash |

### 6 Set brokerage

#### 6.1 Example

const platformAccount = "jJSEWTMsB3WFsZyoGGi977wQdTqTmkBFwV";

const platformSecret = "sxxxxxxx";

const feeAccount = "j98XXLZUCqvP4x7rgFeeYekNgZCotffBH5";

const rateNum = "2";

const rateDen = "1000";

const token = "jjcc";

const issuer;

const hash = await JCCExchange.setBrokerage (platformAccount, platformSecret, feeAccount, rateNum, rateDen, token, issuer);

Parameters：

|  |  |  |
| --- | --- | --- |
| Parameters | Type | Description |
| platformAccount | String | Wallet address |
| platformSecret | String | Wallet secret |
| feeAccount | String | Address of receiving |
| rateNum | Number | Fee numerator |
| rateDen | Number | Fee denominator |
| token | String | Token name(case insensitive) |
| issuer | String | Issuer address; default value is jGa9J9TkqtBcUoHe2zqhVFFbgUVED6o9or |

result：

hash:"95A3F55A7DD81695D511CF7F257A9B86FA9DF391B000517A38A8AE303A0CB200"

Paramters：

|  |  |  |
| --- | --- | --- |
| Parameters | Type | Description |
| hash | String | Transaction hash |