Transferring Tokens (Example)

Transferring from one token to another, across one or more chains, is simple. The following section will explain step by step how to transfer tokens across one or more chains.

Here you can find our fullAPI reference

Step 1: Requesting a Quote

The request for a transfer from 1 USDC on Gnosis to USDC on Polygon looks the following:

```

Copy constgetQuote=async(fromChain,toChain,fromToken,toToken,fromAmount,fromAddress)=>{ constresult=awaitaxios.get('https://li.quest/v1/quote',{ params:{ fromChain, toChain, fromToken, toToken, fromAmount, fromAddress, } }); returnresult.data; }

constfromChain='DAI'; constfromToken='USDC'; consttoChain='POL'; consttoToken='USDC'; constfromAmount='1000000'; constfromAddress=YOUR_WALLET_ADDRESS;

constquote=awaitgetQuote(fromChain,toChain,fromToken,toToken,fromAmount,fromAddress);

```

Thequote response contains atransactionRequest object which can be directly passed on to your wallet/signer.

To learn more about quotes, head over to ourRequesting a Quote guide.

Step 2: Sending the Transaction

After receiving a quote, the transaction has to be sent to trigger the transfer.

Firstly, the wallet has to be configured. The following example connects your wallet to the Gnosis Chain.

```

Copy constprovider=newethers.providers.JsonRpcProvider('https://rpc.xdaichain.com/',100); constwallet=ethers.Wallet.fromMnemonic(YOUR_PERSONAL_MNEMONIC).connect( provider );

```

Afterward, the transaction can be sent using thetransactionRequest inside the previously retrievedquote :

```

Copy consttx=awaitwallet.sendTransaction(quote.transactionRequest); awaittx.wait();

```

For simple swaps on a single chain, this is enough. For cross-chain transfers, the processing takes a bit longer. To handle this the API provides an endpoint to check the transfer status.

Step 3: Waiting for the transfer to complete

Checking the status of the transfer is only necessary for cross-chain transfers To check if the token was successfully sent to the receiving chain, the/status endpoint can be called:

```

Copy constgetStatus=async(bridge,fromChain,toChain,txHash)=>{ constresult=awaitaxios.get('https://li.quest/v1/status',{ params:{ bridge, fromChain, toChain, txHash, } }); returnresult.data; }

result=awaitgetStatus(quote.tool,fromChain,toChain,tx.hash);

```

If you want to learn more about the status endpoint and how to deal with the potential results check outStatus Guide .

Links to the example transactions

Here you can find links to the sending and receiving transactions for the transfer from USDC on Gnosis to USDC on Polygon.

Sending:
https://blockscout.com/xdai/mainnet/tx/0x32375265d54d2897f776b10729fb7ba8f7ed4cb329364d8fa0e13c89ecbb9085

Receiving: https://polygonscan.com/tx/0x9c762972efd140124a7a3c2753f56dac35968b279dd678f043e716fe0d0dbc67

All things plugged together

The whole process then looks like this:

```

Copy constethers=require('ethers'); constaxios=require('axios');

constAPI_URL='https://li.quest/v1';

// Get a quote for your desired transfer
constgetQuote=async(fromChain,toChain,fromToken,toToken,fromAmount,fromAddress)=>{
constresult=awaitaxios.get({API_URL}/quote,{ params:{ fromChain, toChain, fromToken, toToken, fromAmount, fromAddress, }
}); returnresult.data; }

// Check the status of your transfer constgetStatus=async(bridge,fromChain,toChain,txHash)=>{
constresult=awaitaxios.get({API_URL}/status,{ params:{ bridge, fromChain, toChain, txHash, } }); returnresult.data; }

constfromChain='DAI'; constfromToken='USDC'; consttoChain='POL'; consttoToken='USDC'; constfromAmount='1000000';
constfromAddress=YOUR_WALLET_ADDRESS;

// Set up your wallet constprovider=newethers.providers.JsonRpcProvider('https://rpc.xdaichain.com/',100);
constwallet=ethers.Wallet.fromMnemonic(YOUR_PERSONAL_MNEMONIC).connect( provider );

construn=async()=>{ constquote=awaitgetQuote(fromChain,toChain,fromToken,toToken,fromAmount,fromAddress);
consttx=awaitwallet.sendTransaction(quote.transactionRequest);

awaittx.wait();

// Only needed for cross chain transfers if(fromChain!==toChain) { letresult; do{
result=awaitgetStatus(quote.tool,fromChain,toChain,tx.hash); }while(result.status!=='DONE'&&result.status!=='FAILED') } }

run().then(()=>{ console.log('DONE!') });

```

Checking and setting the Allowance

Before any transaction can be sent, it must be made sure that the user is allowed to send the requested amount from his wallet.

This can be achieved like this:

```

Copy const{Contract}=require('ethers');

constERC20_ABI=[ { "name":"approve", "inputs":[ { "internalType":"address", "name":"spender", "type":"address" }, {
"internalType":"uint256", "name":"amount", "type":"uint256" } ], "outputs":[ { "internalType":"bool", "name":"", "type":"bool" } ],
"stateMutability":"nonpayable", "type":"function" }, { "name":"allowance", "inputs":[ { "internalType":"address",
"name":"owner", "type":"address" }, { "internalType":"address", "name":"spender", "type":"address" } ], "outputs":[ {
"internalType":"uint256", "name":"", "type":"uint256" } ], "stateMutability":"view", "type":"function" } ];

// Get the current allowance and update it if needed
constcheckAndSetAllowance=async(wallet,tokenAddress,approvalAddress,amount)=>{ // Transactions with the native token
don't need approval if(tokenAddress===ethers.constants.AddressZero) { return }

consterc20=newContract(tokenAddress,ERC20_ABI,wallet);
constallowance=awaiterc20.allowance(awaitwallet.getAddress(),approvalAddress);

if(allowance.lt(amount)) { constapproveTx=awaiterc20.approve(approvalAddress,amount); awaitapproveTx.wait(); } }

awaitcheckAndSetAllowance(wallet,quote.action.fromToken.address,quote.estimate.approvalAddress,fromAmount);

``` Last updated2 months ago On this page *Step 1: Requesting a Quote * Step 2: Sending the Transaction * Step 3: Waiting for the transfer to complete * Links to the example transactions* All things plugged together * Checking and setting the Allowance

Was this helpful? Export as PDF