# Cross-Chain Contract Calls

Cross-Chain Contract Calls: This feature enables contract calls across chains, buy NFTs, stake or donate without needing gas on the destination chain! Destination calls are currently supported on Polygon, Binance, Optimism, Ethereum, Fantom, Avalanche and Arbitrum. To read more about what xChain Contract calls are and how they work, please clickhere .

With this feature, you can make arbitrary contract calls across blockchains. All you need to do is prepare thecallData for the contract call you want to make on the destination chain (e.g. stake KLIMA in the example below). Pass thatcallData along with information about where you want to pay for it to the endpoint. The response contains information on how many tokens must be submitted to pay for all transfer and swap fees.

Parameters

About the contract call:

Parameter Explaination Example Data toChain

(required) The chain the contract is deployed on. Can be the chain id or chain key. 137 orPOL toToken

(required) The token required to perform the contract interaction (can be something to stake, donate or to be used as payment). 0x4e78011ce80ee02d2c3e649fb657e45898257815 toAmount

(required) The amount of token required by the contract interaction. The LI.FI API will try and generate a quote that guarantees at least that amount on the destination chain. 300000000 toContractAddress (required) The address of the contract to interact with. 0x4D70a031Fc76DA6a9bC0C922101A05FA95c3A227 toContractCallData (required) ThecallData to be sent to the contract for the interaction on the destination chain. 0x7d7a... toContractGasLimit (required) The estimated gas used by the destination call. If this value is incorrect, the interaction may fail -- choose this carefully! 900000 toApprovalAddress (optional) If the approval address is different thant the contract to call, specify that address here. 0x4D70a031Fc76DA6a9bC0C922101A05FA95c3A227 contractOutputsToken (optional) Some contract interactions will output a token (e.g. staking), specify the token address to forward the token to the user. Omit this parameter if no token should be returned to the user. 0xb0c22d8d350c67420f06f48936654f567c73e8c8 toFallbackAddress (optional) If the call fails, use this address to send the bridged tokens to. If none is specified, the sending address will be used. 0x552008c0f6870c2f77e5cC1d2eb9bdff03e30Ea0 How does the user want to pay:

Parameter Explaination Example Data fromChain

(required) The sending chain. Can be the chain id or chain key. 56 orBSC fromToken (required) The token that the user pays with. Can be the address or the symbol. 0x1af3f329e8be154074d8769d1ffa4ee058b1dbc3 fromAddress (required) The wallet that will send the transaction and contains the starting token. 0x552008c0f6870c2f77e5cC1d2eb9bdff03e30Ea0 Additional options:

Parameter Explaination Example Data slippage (optional) The maximum allowed slippage for the transaction as a decimal value. 0.05 represents 5%. 0.02 integrator (optional) A string containing tracking information about the integrator of the API. my-dapp referrer (optional) A string containing tracking information about the referrer of the integrator. 0x552008c0f6870c2f77e5cC1d2eb9bdff03e30Ea0 fee

(optional) The percent of the integrator's fee that is taken from every transaction 0.01

Example: stake KLIMA on Polygon

The example below demonstrates how to stake KLIMA token (KlimaDAO ) into sKLIMA (staked KLIMA). The process involves

- sending someDAI
- token from Binance Smart Chain (BSC) to Polygon (POL)
- swappingDAI
- forKLIMA
- performing a contract call to the KLIMA staking contract.
-

ThecallData will be executed by our executer contract, not by the users wallet, so ensure that the contract does not expect the user to bemsg.sender . The staked tokens are sent to the executor which will then forward them to the user if thecontractOutputsToken parameter is set. In the end, the contract will produce the stakedKLIMA tokensKLIMA which will then be transferred back to the user.

```

Copy importaxiosfrom'axios'; import{ ethers }from'ethers';

constendpoint='https://li.quest/v1/quote/contractCall';
```

```
constDAI_ON_BSC='0x1af3f329e8be154074d8769d1ffa4ee058b1dbc3';
constKLIMA_ON_POL='0x4e78011ce80ee02d2c3e649fb657e45898257815';
constSKLIMA_ON_POL='0xb0c22d8d350c67420f06f48936654f567c73e8c8';

constKLIMA_STAKING_CONTRACT='0x4D70a031Fc76DA6a9bC0C922101A05FA95c3A227';

// Full ABI on // https://polygonscan.com/address/0x4D70a031Fc76DA6a9bC0C922101A05FA95c3A227#code
constKLIMA_STAKING_ABI=['function stake(uint _amount) external'];

constgenerateKLIMATransaction=async(receivedAmount:string)=>{ conststakeKlimaTx=awaitnewethers.Contract(
KLIMA_STAKING_CONTRACT, KLIMA_STAKING_ABI ).populateTransaction.stake(receivedAmount); returnstakeKlimaTx;
};

constgetQuote=async():Promise=>{ // We would like to stake this amount of KLIMA to get sKLIMA
conststakeAmount='300000000';

conststakeKlimaTx=awaitgenerateKLIMATransaction(stakeAmount);

constquoteRequest={ fromChain:'BSC', fromToken:DAI_ON_BSC,
fromAddress:'0x552008c0f6870c2f77e5cC1d2eb9bdff03e30Ea0', toChain:'POL', toToken:KLIMA_ON_POL,
toAmount:stakeAmount, toContractAddress:stakeKlimaTx.to, toContractCallData:stakeKlimaTx.data,
toContractGasLimit:'900000', contractOutputsToken:SKLIMA_ON_POL, };

constresponse=awaitaxios.post(endpoint,quoteRequest); returnresponse.data; };

getQuote().then(console.log);
```

Once the quote is generated, you can execute it like any other quote as explained here .

Links to the example transactions

Here you can find links to the sending and receiving transactions for the transfer from DAI on BNB Smart Chain to sKLIMA on Polygon via the custom contract call.

Sending: https://bscscan.com/tx/0x3cbc7e29ad89d0b04a8f59a4851fae17a65b7b5248567da74f5462e28642231f

Receiving: https://polygonscan.com/tx/0xeb399611d324467c9dbd7cf2006e3c73aa982ca19b67b987365e4098162a5a09

Last updated3 months ago On this page *Parameters * Example: stake KLIMA on Polygon * Links to the example transactions

Was this helpful? Export as PDF