# Calling

How to Implement Calling (Executing Cross Chain Transactions)

Summary

1 Fetch the Price

This optional step is quicker than fetching an executable quote but invaluable when crafting a snappy user experience. In this step, we'll fetch the price to display a quote preview to the user. This can be skipped if the call is on the server without user interaction. 2 Fetch the Quote

In this step, we'll fetch a full quote that expires after 30 seconds. It will be a complete quote with all the details required to execute it. 3 Execute the Quote

Once we have the quote all that's left to do is execute the quote and display a progress state to the user.

Contract Compatibility

Before we begin it's important to ensure that a contract is compatible with Relay. Please review ourContract Compatibility overview to make any necessary changes.

Fetch the Price

The price is a lightweight quote, excluding steps and calldata; but it includes a preview of what the full quote might contain. Prices are subject to change once you get the quote so you should handle price changes accordingly. You'll need to determine where the user wants to call the contract from, where the contract has been deployed to, and their preferred payment currency. Afterwards, you can pass everything into thegetPrice SDK action or theusePrice UI hook. Note that for calling a contract across chains you'll need to provide three key pieces of information in addition to the usual data required to call:

- tradeType
- must beEXACT_OUTPUT
- .
- txs
- can either be an array ofto
- ,data
- andvalue
- or it can be the response from viem'ssimulateContract
- method. You can see an example below of how this is done.
- amount
- must be the sum of all the txs values.

sdk sdk simulateContract hooks hooks simulateContract Copy import { getClient} from '@reservoir0x/relay-sdk' import { useAccount} from 'wagmi' const { address} = useAccount ( ) const CONTRACT_ADDRESS = "0xabc" //Replace with your contract address const CALL_DATA = "0x" // Replace with your calldata const price= await getClient ( ) ?. actions. getPrice ( { user: address originChainId: 1 , // The chain id to call from destinationChainId: 7777777 , // The chain id to call to amount: '100000000000000000' , // Total value of txs originCurrency: '0x0000000000000000000000000000000' , // ERC20 Address destinationCurrency: '0x0000000000000000000000000000000' , // ERC20 Address tradeType: 'EXACT_OUTPUT' , txs: [ { to: CONTRACT_ADDRESS , value: "100000000000000000" , //Must match the amount property data: CALL_DATA } ] } )

Fetch the Quote

If you followed the previous step, you can pass that same data into thegetQuote SDK action oruseQuote UI hook. You'll still need to meet the requirements fortradeType ,amount andtxs , refer to the previous step for more details.

sdk sdk simulateContract hooks hooks simulateContract Copy import { getClient} from '@reservoir0x/relay-sdk' import { useWalletClient} from 'wagmi' const { data: wallet} = useWalletClient ( ) const CONTRACT_ADDRESS = "0xabc" //Replace with your contract address const CALL_DATA = "0x" // Replace with your calldata const quote= await getClient ( ) ?. actions. getQuote ( { wallet, chainId: 1 , // The chain id to call from toChainId: 7777777 , // The chain id to call to amount:

'100000000000000000' , // Total value of txs currency: '0x0000000000000000000000000000' , // ERC20 Address toCurrency: '0x0000000000000000000000000000' , // ERC20 Address tradeType: 'EXACT_OUTPUT' , txs: [ { to: CONTRACT_ADDRESS , value: "100000000000000000" , //Must match the amount property data: CALL_DATA } ] } ) In the above example, we fetch the quote to call a Zora contract from Mainnet Ethereum. In the quote object, we'll get data on the fees required to do this call. You can learn more about feeshere . You'll see that we provided an amount that matches the sum of the txs, that the tradeType isEXACT_OUTPUT and that we passed in a valid txs array.

Quotes can go stale after 30 seconds. Clients should regularly fetch fresh quotes so that users are always submitting valid transactions.Learn more about Relay quotes.

Execute the Quote

After getting the quote, all that's left is to execute the quote and render the progress to the user. Follow the code snippets below to execute the quote:

sdk hooks Copy import { getClient} from '@reservoir0x/relay-sdk' import { useWalletClient} from 'wagmi' const { data: wallet} = useWalletClient ( ) const quote= await getClient ( ) ?. actions. getQuote ( { ... options } ) await getClient ( ) ?. actions. execute ( { quote, wallet, onProgress : ( steps, currentStep, currentStepItem, fees, details, txHashes) => { console . log ( steps, currentStep, currentStepItem, fees, details, txHashes) } } ) While the quote is being executed if you're rendering the quote in an interface you may want to display progress to the user. There's a provided progress callback to get the updated state of execution. Learn more about the progress statehere .

Preflight Checklist

☐ Verify that the user has enough balance on the origin chain to cover the full txs amount + fees. Check out our fees doc for more details on calculating the fees. ☐ Have the Relay SDK set up and the Relay hooks package installed if applicable ☐ Ensure the user is on the correct chain (it should match the chainId you get the quote from) ☐ Handle any errors that may come about ☐ Render the progress of the execution to provide a best in class user experience Swapping Solana Support twitter Powered by Mintlify

On this page * Summary * Contract Compatibility * Fetch the Price * Fetch the Quote * Execute the Quote * Preflight Checklist