Continue

Compound.js is a JavaScript SDK for Ethereum and the Compound Protocol. It wraps around Ethers.js, which is its only dependency. It is designed for both the web browser and Node.js. The SDK is currently in open beta. Use at your own risk. For bugs reports and feature requests, either create an issue in the GitHub repository or send a message in the Development channel of the Compound Discord. Constructor Creates an instance of the Compound.js SDK. [provider] (Provider | string) Optional Ethereum network provider. Defaults to Ethers.js fallback mainnet provider. [options] (object) Optional provider options. RETURN (object) Returns an instance of the Compound.js SDK. var compound = new Compound(window.ethereum); // web browservar compound = new Compound(' '); // HTTP providervar compound = new Compound('goerli'); // Uses Ethers.js fallback (for testing only)// Init with private key (server side)var compound = new Compound(' , { privateKey: '0x your_private_key ', // preferably with environment variable});// Init with HD mnemonic (server side)var compound = new Compound('mainnet' { mnemonic: 'clutch captain shoe...', // preferably with environment variable});Compound III (Comet) Object Initialization. This accepts the same parameters as the Compound constructor. An error will be thrown initially and whenever a method is called if the provider does not match the network of the specific Comet deployment. The SDK constants as well as a method in the Comet documentation note the Comet deployments that Compound.js supports.var compound = new Compound(window.ethereum);var comet = compound.comet.MAINNET USDC(); // provider from `compound` will be used unless on is explicitly passedEthereum Methods These methods facilitate interactions with the Ethereum blockchain. Ethereum Read This is a generic method for invoking JSON RPCs eth_call with Ethers.js. Use this method to execute a smart contracts constant or non-constant member without using gas. This is a read-only method intended to read a value or test a transaction for valid parameters. It does not create a transaction on the block chain. address (string) The Ethereum address the transaction is directed to. method (string) The smart contract member in which to invoke. [parameters] (any[]) Parameters of the method to invoke. [options] (CallOptions) Options to set for eth_call, optional ABI (as JSON object), and Ethers.js method overrides. The ABI can be a string of the single intended method, an array of many methods, or a JSON object of the ABI generated by a Solidity compiler. RETURN (Promise) Return value of the invoked smart contract member or an error object if the call failed. const cEthAddress = Compound.util.getAddress(Compound.cETH);(async function() { const srpb = await Compound.eth.read(cEthAddress, 'function supplyRatePerBlock() returns (uint256)', // [], // [optional] parameters // { } // [optional] call options, provider, network, plus Ethers.js "overrides" }; console.log('cETH market supply rate per block:', srpb.toString()); })().catch(console.error);Ethereum Trx This is a generic method for invoking JSON RPCs eth_sendTransaction with Ethers.js. Use this method to create a transaction that invokes a smart contract method. Returns an Ethers.js TransactionResponse object. address (string) The Ethereum address the transaction is directed to. method (string) The smart contract member in which to invoke. [parameters] (any[]) Parameters of the method to invoke. [options] (CallOptions) Options to set for eth_sendTransaction, (as JSON object), and Ethers.js method overrides. The ABI can be a string optional ABI of the single intended method, an array of many methods, or a JSON object of the ABI generated by a Solidity compiler. RETURN (Promise) Returns an Ethers.js TransactionResponse object or an error object if the transaction failed. const oneEthInWei = '1000000000000000000';const cEthAddress = '0x4ddc2d193948926d02f9b1fe9e1daa0718270ed5';const provider = window.ethereum;(async function() { console.log('Supplying ETH to the Compound Protocol...'); // Mint some cETH by supplying ETH to the Compound Protocol...'); // Mint some cETH by supplying ETH to the Compound Protocol...'); // Mint some cETH by