Wallet integrations with Celestia

This page covers how developers can use Keplr and React to add Celestia network parameters to wallets, and how to add custom networks to Leap and Cosmostation.

Add Celestia network parameters to Keplr with React

Before we demonstrate how to export the specific parameters for Celestia's testnets, we need to create a ReactJS component that allows us to connect directly to Keplr and pass it the network parameters.

In the following code, we show how you can export a component that detects whether Keplr is installed and sets the network params for it:

Keplr jsx // @site/src/components/AddNetworkKeplr.js import React from

"react" ; import styles from

"./Keplr.module.css" ;

export

default

function

AddNetworkKeplr ({ params }) { async

function

add () { if (! window.keplr) { alert ("Please install keplr extension"); } else { if (window.keplr.experimentalSuggestChain) { try { await window.keplr. experimentalSuggestChain ({ chainId: params.chainId, chainName: params.chainName, rpc: params.rpc, rest: params.rest, bip44: { coinType: 118 , }, bech32Config: { bech32PrefixAccAddr: "celestia" , bech32PrefixAccPub: "celestia"

+

"pub" , bech32PrefixValAddr: "celestia"

+

"valoper" , bech32PrefixValPub: "celestia"

+

"valoperpub" , bech32PrefixConsAddr: "celestia"

+

"valcons" , bech32PrefixConsPub: "celestia"

+

"valconspub" , }, currencies: [{ coinDenom: "TIA" , coinMinimalDenom: "utia" , coinDecimals: 6 , coinGeckoId: "celestia" , },], feeCurrencies: [{ coinDenom: "TIA" , coinMinimalDenom: "utia" , coinDecimals: 6 , coinGeckoId: "celestia" , gasPriceStep: { low: 0.01 , average: 0.02 , high: 0.1 , }, },], stakeCurrency: { coinDenom: "TIA" , coinMinimalDenom: "utia" , coinDecimals: 6 , coinGeckoId: "celestia" , }, }); } catch { alert ("Failed to suggest the chain"); } } const

chainId

= params.chainId; // Enabling before using the Keplr is recommended. // This method will ask the user whether to allow access if they haven't visited this website. // Also, it will request that the user unlock the wallet if the wallet is locked. await window.keplr. enable (chainId); } }

return (< div

className

{styles.center}> < button

className

{styles.keplrButton} onClick = {add}> Add/switch To {params.chainName} </ button

</ div

); } // @site/src/components/AddNetworkKeplr.js import React from

"react" ; import styles from

"./Keplr.module.css" ;

export

default

function

AddNetworkKeplr ({ params }) { async

function

add () { if (! window.keplr) { alert ("Please install keplr extension"); } else { if (window.keplr.experimentalSuggestChain) { try { await window.keplr. experimentalSuggestChain ({ chainId: params.chainId, chainName: params.chainName, rpc: params.rpc, rest: params.rest, bip44: { coinType: 118 , }, bech32Config: { bech32PrefixAccAddr: "celestia" , bech32PrefixAccPub: "celestia"

+

"pub" , bech32PrefixValAddr: "celestia"

+

"valoper" , bech32PrefixValPub: "celestia"

+

"valoperpub" , bech32PrefixConsAddr: "celestia"

+

"valcons" , bech32PrefixConsPub: "celestia"

+

"valconspub" , }, currencies: [{ coinDenom: "TIA" , coinMinimalDenom: "utia" , coinDecimals: 6 , coinGeckoId: "celestia" , },], feeCurrencies: [{ coinDenom: "TIA" , coinMinimalDenom: "utia" , coinDecimals: 6 , coinGeckoId: "celestia" , gasPriceStep: { low: 0.01 , average: 0.02 , high: 0.1 , }, },], stakeCurrency: { coinDenom: "TIA" , coinMinimalDenom: "utia" , coinDecimals: 6 , coinGeckoId: "celestia" , }, }); } catch { alert ("Failed to suggest the chain"); } } const

chainId

= params.chainId; // Enabling before using the Keplr is recommended. // This method will ask the user whether to allow access if they haven't visited this website. // Also, it will request that the user unlock the wallet if the wallet is locked. await window.keplr. enable (chainId); } }

return (< div

className

{styles.center}> < button

className

{styles.keplrButton} onClick = {add}> Add/switch To {params.chainName} </ button

</ div

); } We still need to pass the Celestia network parameters to theAddNetworkKeplr function:

Mainnet Beta

Mocha

Arabica js import

'@site/src/components/AddNetworkKeplr'

export

const

MAINNET_PARAMS

= { { chainId: 'celestia', chainName: 'Celestia', rpc: 'https://rpc.lunaroasis.net/', rest: 'https://api.lunaroasis.net/' } }

{< AddNetworkKeplr

params

{ MAINNET_PARAMS }/>} import

'@site/src/components/AddNetworkKeplr'

export

const

MAINNET_PARAMS

= { { chainId: 'celestia', chainName: 'Celestia', rpc: 'https://rpc.lunaroasis.net/', rest: 'https://api.lunaroasis.net/' } }

{< AddNetworkKeplr

params

{ MAINNET_PARAMS }/>} js import

'@site/src/components/AddNetworkKeplr'

export

const

MOCHA_PARAMS

= { { chainId: 'mocha-4', chainName: 'Mocha testnet', rpc: 'https://rpc-mocha.pops.one/', rest: 'https://api-mocha.pops.one/' } }

{< AddNetworkKeplr

params

{ MOCHA_PARAMS }/>} import

'@site/src/components/AddNetworkKeplr'

export

const

MOCHA_PARAMS

= { { chainId: 'mocha-4', chainName: 'Mocha testnet', rpc: 'https://rpc-mocha.pops.one/', rest: 'https://api-mocha.pops.one/' } }

{< AddNetworkKeplr

params

{ MOCHA_PARAMS }/>} js import

'@site/src/components/AddNetworkKeplr'

export

const

ARABICA_PARAMS

= { { chainId: 'arabica-11', chainName: 'Arabica devnet', rpc: 'https://rpc.celestia-arabica-11.com/', rest: 'https://api.celestia-arabica-11.com' } }

{< AddNetworkKeplr

params

{ ARABICA_PARAMS }/>} import

'@site/src/components/AddNetworkKeplr'

export

const

ARABICA_PARAMS

= { { chainId: 'arabica-11', chainName: 'Arabica devnet', rpc: 'https://rpc.celestia-arabica-11.com/', rest: 'https://api.celestia-arabica-11.com' } }

{< AddNetworkKeplr

params

{ ARABICA_PARAMS }/>} Now, we can connect to the network that you would like to use in Keplr wallet.

Adding a custom chain to Leap

If you want to add a custom chain to Leap, you can do so by:

1. Clicking the Cosmos logo in the top corner of Leap wallet
2. Scrolling down and clicking "Add new chain"

You can then add the following parameters:

- Chain Id:arabica-11
- Chain Name:Arabica devnet
- New RPC URL:https://rpc.celestia-arabica-11.com/
- New REST URL:https://api.celestia-arabica-11.com
- Address Prefix:celestia
- Native Denom:utia
- Coin Type:118
- Decimals:6
- Block explorer URL (optional):https://explorer.celestia-arabica-10.com

Now, clickAdd chain and you will be able to view your Arabica account balance and transactions in Leap wallet.

You'll see that you're connected to Arabica Devnet.

Addding a custom chain to Cosmostation

Click the hamburger menu icon in the top corner of Cosmostation wallet. Scroll down and click "Add Custom Chain"

You can then add the following parameters:

- Custom Chain name:Mocha testnet
- Rest URL:https://api-mocha.pops.one
- New RPC URL:https://rpc-mocha.pops.one
- Currency symbol:TIA
- Address prefix:celestia

- Denom:utia
- Symbol image URL (optional):https://raw.githubusercontent.com/cosmos/chain-registry/master/testnets/celestiatestnet/images/celestia.svg
- Explorer URL (optional):https://testnet.mintscan.io/celestia-testnet
- Coin Type:118
- Decimals:6
- Gas rate Tiny:0.1
- Gas rate Low:0.25
- Gas rate Average:0.5

Now, clickAdd a custom chain and you will be able to view your Celestia account balance and transactions in Cosmostation wallet.

Switch chains to "Mocha testnet" and you'll see that you're connected to Celestia's Mocha testnet! [] Edit this page on GitHub] Last updated: Previous page Celestia-node Next page Integrate Celestia for service providers []