XRPL Private Key Provider for PnP Web SDKs

@web3auth/xrpl-provider

â

The@web3auth/xrpl-provider is a Web3Auth provider that simplifies interaction with the XRPL Blockchain by serving as a wrapper around theXRPL JSON RPC API . It is used to interact with the XRPL blockchain and perform various operations like getting user's account, balance, signing a transaction, sending a transaction etc.

In this section we'll explore more about how you can use this provider with our SDKs.

Installationâ

@web3auth/xrpl-provider

â

npm install --save @web3auth/xrpl-provider

Initialisationâ

Import theXrplPrivateKeyProvider class from@web3auth/xrpl-provider .

import

{

XrplPrivateKeyProvider

}

from

"@web3auth/xrpl-provider" ;

Assign theXrplPrivateKeyProvider

class to a variableâ

After creating your Web3Auth instance, you need to initialize the Torus Wallet UI Plugin and add it to a class for further usage.

const privateKeyProvider =

new

XrplPrivateKeyProvider ({

config :

XrplPrivKeyProviderConfig

}) ; This constructor takes an object with a config ofXrplPrivKeyProviderConfig as input.

Argumentsâ

XrplPrivKeyProviderConfig

export

interface

XrplPrivKeyProviderConfig

extends

BaseProviderConfig

```
{ chainConfig :
Omit < CustomChainConfig ,
"chainNamespace"
&
Pick < CustomChainConfig ,
"wsTarget"
    ; } export
type
CustomChainConfig
=
{ chainNamespace :
ChainNamespaceType ; /* * The chain id of the chain/ chainId :
string ; /* * RPC target Url for the chain/ rpcTarget :
string ; /* * web socket target Url for the chain/ wsTarget ? :
string ; /* * Display Name for the chain/ displayName :
string ; /* * Url of the block explorer/ blockExplorer :
string ; /* * Default currency ticker of the network (e.g: ETH)/ ticker :
string ; /* * Name for currency ticker (e.g:Ethereum) / tickerName :
string ; /* * Number of decimals for the currency ticker (e.g: 18)/ decimals ? :
number ; } ; export
interface
BaseProviderConfig
extends
BaseConfig
{ chainConfig :
Partial < CustomChainConfig
    ; networks ? :
Record < string ,
CustomChainConfig
    ; skipLookupNetwork ? :
boolean ; } export
interface
BaseConfig
{ /* * Determines if this controller is enabled / disabled ? :
boolean ; }
```

Getting thechainConfig

â

- Mainnet
- Testnet
- Devnet

const chainConfig =

{ chainNamespace :

CHAIN_NAMESPACES . OTHER , chainId :

"0x1" , // Avoid using public rpcTarget & wsTarget in production. // Use services like Infura, Quicknode etc rpcTarget :

"https://ripple-node.tor.us" , wsTarget :

"wss://s2.ripple.com" , ticker :

"XRP" , tickerName :

"XRPL" , displayName :

"xrpl mainnet" , blockExplorerUrl :

"https://livenet.xrpl.org" , } ; const chainConfig =

{ chainNamespace :

CHAIN_NAMESPACES . OTHER , chainId :

"0x2" , // Avoid using public rpcTarget & wsTarget in production. // Use services like Infura, Quicknode etc rpcTarget :

"https://testnet-ripple-node.tor.us" , wsTarget :

"wss://s.altnet.rippletest.net" , ticker :

"XRP" , tickerName :

"XRPL" , displayName :

"xrpl testnet" , blockExplorerUrl :

"https://testnet.xrpl.org" , } ; const chainConfig =

{ chainNamespace :

CHAIN_NAMESPACES . OTHER , chainId :

"0x3" , // Avoid using public rpcTarget & wsTarget in production. // Use services like Infura, Quicknode etc rpcTarget :

"https://devnet-ripple-node.tor.us" , wsTarget :

"wss://s.devnet.rippletest.net/" , ticker :

"XRP" , tickerName :

"XRPL" , displayName :

"xrpl devnet" , blockExplorerUrl :

"https://devnet.xrpl.org" , } ;

Initializing Providerâ

- PnP Modal SDK
- PnP NoModal SDK
- CoreKit SFA Web SDK

import

{

Web3Auth

```
}
from
"@web3auth/modal" ; import
{
AuthAdapter
}
from
"@web3auth/auth-adapter" ; import
{
XrplPrivateKeyProvider
}
from
"@web3auth/xrpl-provider" ;
const privateKeyProvider =
new
XrplPrivateKeyProvider ( { config : chainConfig , } ) ;
const web3auth =
new
Web3Auth ( { clientId ,
// get from https://dashboard.web3auth.io privateKeyProvider , web3AuthNetwork =
"sapphire_mainnet" ,
// testnet, mainnet, cyan, aqua } ) ; import
{
Web3AuthNoModal
}
from
"@web3auth/no-modal" ; import
{
AuthAdapter
}
from
"@web3auth/auth-adapter" ; import
{
XrplPrivateKeyProvider
}
from
"@web3auth/xrpl-provider" ;
```

```
const privateKeyProvider =
new
XrplPrivateKeyProvider ( { config : chainConfig , } ) ;
const web3auth =
new
Web3AuthNoModal ( { clientId ,
// get it from Web3Auth Dashboard web3AuthNetwork :
"sapphire_mainnet" , privateKeyProvider , } ) ;
const authAdapter =
new
AuthAdapter ( { privateKeyProvider } ) ; web3auth . configureAdapter ( authAdapter ) ; import
{
Web3Auth
}
from
"@web3auth/single-factor-auth" ; import
{
XrplPrivateKeyProvider
}
from
"@web3auth/xrpl-provider" ;
const privateKeyProvider =
new
XrplPrivateKeyProvider ( { config : chainConfig , } ) ;
const web3auth =
new
Web3Auth ( { clientId ,
// Get your Client ID from Web3Auth Dashboard web3AuthNetwork :
"sapphire_mainnet" , privateKeyProvider , } ) ;
```

Usageâ

After configuring the provider, you may utilize various functions from the@web3auth/xrpl-provider library for tasks such as obtaining the user's account, executing transactions, and signing messages.

info All the RPC methods which are available by default on XRPL Blockchain are also available on the XRPL Provider.

Please refer to ourXRPL Connect Blockchain Reference for more information.

Examplesâ

Integrate PnP Modal SDK with XRP Ledger SAMPLE APP Use XRP Ledger with Plug and Play Modal SDK Source Code Guide pnp web @web3auth/modal javascript xrpl ### Integrate PnP No Modal SDK with XRP Ledger SAMPLE APP Use XRP Ledger with Plug and Play No Modal SDK Source Code Guide pnp web @web3auth/no-modal javascript xrpl Edit