Initializing MPC Core Kit Web SDK

After Installation, the next step to use Web3Auth MPC Core Kit is to Initialize the SDK.

However, the Initialization is a two step process, ie.

- Instantiation ofWeb3AuthMPCCoreKit
- Initialization ofWeb3AuthMPCCoreKit

Instantiating Web3Authâ

Import theWeb3AuthMPCCoreKit

class from@web3auth/mpc-core-kit packageâ

import

{

Web3AuthMPCCoreKit ,

WEB3AUTH_NETWORK

}

from

"@web3auth/mpc-core-kit" ;

Assign theWeb3AuthMPCCoreKit

class to a variableâ

const coreKitInstance =

new

Web3AuthMPCCoreKit (Web3AuthOptions) ; This Web3AuthMPCCoreKit constructor takes an object withWeb3AuthOptions as input.

Argumentsâ

Web3AuthOptions

â

- Table
- Interface

Parameter Type Description web3AuthClientId string The Web3Auth Client ID for your application. Find one athttps://dashboard.web3auth.io chainConfig CustomChainConfig Chain Config for the chain you want to connect to. Currently supports only EVM based chains. manualSync? boolean Enables you to manually sync with the Web3Auth Metadata Server, helping you manage the API calls to the server. Ideal for managing custom flows. baseUrl? string URL of the service worker handling the authentication in popup flow. For general usecases, you don't need to change this value. web3AuthNetwork WEB3AUTH_NETWORK_TYPE Web3Auth Network used for MPC Wallet Management. Use Devnet/ Testnet in testing environment and Mainnet in production environment. storageKey? SupportedStorageType Select the session key storage across local storage or session storage. Setting to "local" will persist social login session accross browser tabs. @defaultValue "local" asyncStorageKey IAsyncStorage Custom Async Storage Implementation. For general usecases, you don't need to change this value. sessionTime? number Determine the session length in seconds. By default the value is set at 86400 seconds, ie. 7 days. uxMode? CoreKitMode Four uxModes are supported:* 'popup' * : In this uxMode, a popup will be shown to user for login. * 'redirect' * : In this uxMode, user will be redirected to a new window tab for login. * 'nodejs' * : Use this for Node.js projects. * 'react-native' * : Use this for React Native apps.

Use of'redirect' mode is recommended in browsers where popups might get blocked. enableLogging? boolean Enables Logs for the SDK redirectPathName? string Specifies the url path where user will be redirected after login. Redirect Uri for OAuth is baseUrl/redirectPathName. disableHashedFactorKey? boolean Disables the Hashed Key, making the sure user logs in always in a non-custodial mode. Recommended only if you have proper MFA flow setup for the user while creating the account. tssLib? TssLib Custom TSS Library Implementation. For general usecases, you don't need to change this value.

hashedFactorNonce? string Nonce for the hashed factor. setupProviderOnInit? boolean Setup the provider on init. export

interface

Web3AuthOptions

{ /* * *The Web3Auth Client ID for your application. Find one at https://dashboard.web3auth.io*/ web3AuthClientId :

string ; /* * *Chain Config for the chain you want to connect to. Currently supports only EVM based chains*/ chainConfig ? :

CustomChainConfig ; /* * *@defaultValue false* / manualSync ? :

boolean ; /* * *@defaultValue {window.location.origin}/serviceworker* / baseUrl ? :

string ; /* * * *@defaultValue 'sapphire_mainnet'* / web3AuthNetwork ? :

WEB3AUTH_NETWORK_TYPE ; /* * * *@defaultValue 'local'* / storageKey ? :

SupportedStorageType ; /* * *asyncStorageKey take precedence over storageKey. * if asyncStorageKey is provided, storageKey will be ignored. * @defaultValue undefined* / asyncStorageKey ? :

IAsyncStorage ; /* * *@defaultValue 86400*/ sessionTime ? :

number ; /* * *@defaultValue 'POPUP'* / uxMode ? :

CoreKitMode ; /* * *@defaultValue false * enables logging of the internal packages.*/ enableLogging ? :

boolean ; /* * *This option is used to specify the url path where user will be * redirected after login. Redirect Uri for OAuth is baseUrl/redirectPathName. * * * @defaultValue "redirect" * * @remarks * At verifier's interface (where you obtain client id), please use baseUrl/redirectPathName * as the redirect_uri * * Torus Direct SDK installs a service worker relative to baseUrl to capture * the auth redirect at redirectPathName path. * * For ex: While using serviceworker ifbaseUrl is "http://localhost:3000/serviceworker" and * redirectPathName is 'redirect' (which is default) * then user will be redirected to http://localhost:3000/serviceworker/redirect page after login * where service worker will capture the results and send it back to original window where login * was initiated. * * For browsers where service workers are not supported or if you wish to not use * service workers,create and serve redirect page (i.e redirect.html file which is * available in serviceworker folder of this package) * * If you are using redirect uxMode, you can get the results directly on your redirectPathName * path using getRedirectResult function. * * For ex: if baseUrl is "http://localhost:3000" andredirectPathName is 'auth' * then user will be redirected to http://localhost:3000/auth page after login * where you can get login result by calling getRedirectResult on redirected page mount. * * Please refer to examples https://github.com/torusresearch/customauth/tree/master/examples * for more understanding. **/ redirectPathName ? :

string ; /* * *@defaultValue false * Disables the cloud factor key, enabling the one key semi custodial flow. * Recommended for Non Custodial Flow.* / disableHashedFactorKey ? :

boolean ; /* * *@defaultValue null * Overwrite tss-lib for nodejs. * Required for nodejs mode. * Do not use this option for non nodejs mode.* / tssLib ? :

unknown ; /* * *@defaultValue Web3AuthOptions.web3AuthClientId * Overwrites the default value (clientId) used as nonce for hashing the hash factor key. * * If you want to aggregate the mfa status of client id 1 and client id 2 apps * set hashedFactorNonce to some common clientID, which can be either client id 1 or client id 2 or any other unique string * #PR 72 * Do not use this unless you know what you are doing.* / hashedFactorNonce ? :

string ; /* * *@defaultValue true * Setup Provider after login success reconstruct.* / setupProviderOnInit ? :

boolean ; }

export

type

WEB3AUTH_NETWORK_TYPE

=

(typeof

WEB3AUTH_NETWORK) [keyof

typeof

WEB3AUTH_NETWORK] ; export

declare

const

WEB3AUTH_NETWORK :

{ readonly

MAINNET :

"sapphire_mainnet" ; readonly

DEVNET :

"sapphire_devnet" ; } ;

export

type

SupportedStorageType

=

"local"

|

"session"

|

"memory"

|

IStorage ;

export

type

CoreKitMode

=

UX_MODE_TYPE

|

"nodejs"

|

"react-native" ; export

type

UX_MODE_TYPE

=

(typeof

UX_MODE) [keyof

typeof

UX_MODE] ; export

declare

const

UX_MODE :

{ readonly

POPUP :

"popup" ; readonly

REDIRECT :

"redirect" ; } ;

Adding a Custom Chain Configurationâ

warning Currently Web3Auth MPC Core Kit supportsonly EVM Compatible Chains . We're constantly working on adding support for more chains.

chainConfig

â

- Table
- Type Declarations

Parameter Description chainNamespace The namespace of your preferred chain. CheckoutProviders SDK Reference for understanding RPC Calls. It acceptsChainNamespaceType as a value. chainId The chain id of the selected blockchain in hexstring format. rpcTarget * RPC Target URL for the selectedchainNamespace * &chainId * . * We provide a default RPC Target for certain blockchains, but due to congestion it might be slow hence it is recommended to provide your own RPC Target URL. wsTarget Web socket target URL for the chain instring . displayName Display Name for the chain instring . blockExplorerUrl Blockchain's explorer URL instring . (eg:https://etherscan.io) ticker Default currency ticker instring for the network (e.g:ETH) tickerName Name for currency ticker instring (e.g:Ethereum) decimals? Number of decimals innumber for the currency ticker (e.g:18) logo Logo for the chain. isTestnet? Defines whether the network is testnet or not. declare

const

CHAIN_NAMESPACES :

{ readonly

EIP155 :

"eip155" ; readonly

SOLANA :

"solana" ; readonly

OTHER :

"other" ; } ;

declare

type

ChainNamespaceType

=

(typeof

CHAIN_NAMESPACES) [keyof

typeof

CHAIN_NAMESPACES] ; declare

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*/ blockExplorerUrl :

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 ; /* * *Logo for the chain*/ logo :

string ; /* * *Whether the network is testnet or not*/ isTestnet ? :

boolean ; } ;

Usageâ

chainConfig :

{ chainNamespace :

"eip155" , chainId :

"0x89" ,

// hex of 137, polygon mainnet rpcTarget :

"https://rpc.ankr.com/polygon" , // Avoid using public rpcTarget in production. // Use services like Infura, Quicknode etc displayName :

"Polygon Mainnet" , blockExplorer :

"https://polygonscan.com" , ticker :

"MATIC" , tickerName :

"Matic" , }

Exampleâ

const coreKitInstance =

new

Web3AuthMPCCoreKit ({ web3AuthClientId :

"BILuyqFCuDXAqVmAuMbD3c4oWEFd7PUENVPyVC-zmsF9euHAvUjqbTCpKw6gO05DBif1YImIVtyaxmEbcLLlb6w" , web3AuthNetwork :

WEB3AUTH_NETWORK . MAINNET , uxMode :

"popup" , }) ;

Initializing Web3Authâ

init(params?: InitParams)

[â]

- Table
- Interface

Parameter Type Description handleRedirectResult boolean true to handle the redirect result during init() rehydrate boolean true to rehydrate the session during init() export

interface

InitParams

{ /* * *@defaultValue* *true* * *handle the redirect result during init()*/ handleRedirectResult :

boolean ; /* * *@defaultValue* *true* * *rehydrate the session during init()*/ rehydrate ? :

boolean ; } The final step in the initialization process is to initialize theWeb3AuthMPCCoreKit instance, ie.coreKitInstance in our case.

This is done by calling theinit() function ofcoreKitInstance .

await coreKitInstance . init ( ) ;Edit this page Previous Install Next Authentication