PnP Modal SDK - v8 to v9

This migration guide provides steps for upgrading from version v8 to v9 of the Web3Auth PnP Modal SDK. The guide outlines significant changes and enhancements, including the support of additonal parameters in Wallet Services function, and build size reduction by 25%.

Breaking Changesâ

OpenloginAdapter is now AuthAdapterâ

In v9, we are deprecating theOpenloginAdapter and introducing theAuthAdapter . Additionally,OpenloginAdapterOptions has been renamed toAuthAdapterOptions . These changes have been made to enhance naming consistency. It's important to note that while the names have changed, the parameters used in the adapter settings remain the same.

import

{

OpenloginAdapter

}

from

"@web3auth/openlogin-adapter" ; import

{

AuthAdapter

}

from

"@web3auth/auth-adapter" ;

const loginAdapter =

new

OpenloginAdapter (OpenloginAdapterOptions) ; const loginAdapter =

new

AuthAdapter (AuthAdapterOptions) ;

Default EVM Adapter Enhancementsâ

In v9, the functionality ofgetDefaultExternalAdapters has changed. Previously, it displayed injected wallets available in the browser. Starting from v9, it will now show a list of WalletConnect-compatible wallets, enabling users to connect via WalletConnect if a wallet is not already installed. Please look at the image below.

Additionally, a newgetInjectedAdapters function has been introduced, which detects injected wallets and, if none are available, prompts users with an option to install one. Refer to the image below for details:

The key difference betweengetInjectedAdapters andgetDefaultExternalAdapters lies in their handling of missing wallets. WhilegetInjectedAdapters provides users with an installation option when no wallet is found,getDefaultExternalAdapters allows users to connect via WalletConnect in the absence of an installed wallet.

Show Injected Walletsâ

import

{ getInjectedAdapters }

from

"@web3auth/default-evm-adapter" ; const adapters =

await

getInjectedAdapters ({ options : web3AuthOptions }) ;

adapters . forEach ((adapter)

=>

{ web3auth . configureAdapter (adapter) ; }) ;

Show WalletConnect-Compatible Walletsâ

import

{ getDefaultExternalAdapters }

from

"@web3auth/default-evm-adapter" ;

const adapters =

await

getDefaultExternalAdapters ({ options : web3AuthOptions }) ;

adapters . forEach ((adapter)

=>

{ web3auth . configureAdapter (adapter) ; }) ;

Default Solana Adapter Enhancementsâ

In v9, a newgetInjectedAdapters function has been introduced, working similarly togetDefaultExternalAdapters . This function has been added to align with default EVM adapter. Unlike the default EVM adapter, there is no functionality change forgetDefaultExternalAdapters .

External Wallet Adapters Deprecationâ

Starting with v9, external wallet adapters for Metamask, Phantom, Solflare, and Slope are deprecated in favor of the Default EVM Adapter and Default Solana Adapter. These new adapters use MIPD to automatically detect injected wallets.

Metamask Adapter Migrationâ

import

{ MetamaskAdapter }

from

"@web3auth/metamask-adapter" ; import

{ getInjectedAdapters }

from

"@web3auth/default-evm-adapter" ;

const metamaskAdapter =

new

MetamaskAdapter ( ) ; const adapters =

await

getInjectedAdapters ({ options : web3AuthOptions }) ; const metamaskAdapter = adapters . find ((e)

=> e . name ==

"metamask") ;

web3auth . configureAdapter (metamaskAdapter) ;

Phantom Adapter Migrationâ

import

{ PhantomAdapter }

from

"@web3auth/phantom-adapter" ; import

{ getInjectedAdapters }

from

"@web3auth/default-solana-adapter" ;

const phantomAdapter =

new

PhantomAdapter ( ) ; const adapters =

await

getInjectedAdapters ({ options : web3AuthOptions }) ; const phantomAdapter = adapters . find ((e)

=> e . name ==

"phantom") ;

web3auth . configureAdapter (phantomAdapter) ;

Solflare Adapter Migrationâ

import

{ SolflareAdapter }

from

"@web3auth/solflare-adapter" ; import

{ getInjectedAdapters }

from

"@web3auth/default-solana-adapter" ;

const solflareAdapter =

new

SolflareAdapter ( ) ; const adapters =

await

getInjectedAdapters ({ options : web3AuthOptions }) ; const solflareAdapter = adapters . find ((e)

=> e . name ==

"solflare") ;

web3auth . configureAdapter (solflareAdapter) ;

Slope Adapter Migrationâ

import

{ SlopeAdapter }

from

"@web3auth/slope-adapter" ; import

{ getInjectedAdapters }

from

"@web3auth/default-solana-adapter" ;

const slopeAdapter =

new

SlopeAdapter ( ) ; const adapters =

await

getInjectedAdapters ({ options : web3AuthOptions }) ; const slopeAdapter = adapters . find ((e)

=> e . name ==

"slope") ;

web3auth . configureAdapter (slopeAdapter) ;

Hooks Auto Initâ

Starting with v9, theinitModal method for initializing the Modal SDK has been removed. Initialization is now handled internally by the hooks.

App.tsx const

{ initModal }

=

useWeb3Auth ( ) ; await

initModal (params :

{ modalConfig :

{ [WALLET_ADAPTERS . AUTH] :

{ label :

"auth" , loginMethods :

{ apple :

{ name :

"apple" , showOnModal :

false , } , } , } , }) ; The modal config instead ofinitModal now needs to be passed inWeb3AuthContextConfig .

web3AuthProviderProps.ts const web3AuthContextConfig : Web3AuthContextConfig =

{ web3AuthOptions , modalConfig :

{ [WALLET_ADAPTERS . AUTH] :

{ label :

"auth" , loginMethods :

{ apple :

{ name :

"apple" , showOnModal :

false , } , } , } , adapters :

[openloginAdapter] , plugins :

[walletServicesPlugin] , } ;

Minor Changesâ

In v9, there are also minor changes that improves the developer experience, and overall performance.

JSON-RPC Method Deprecationsâ

Starting v9, theeth_signTypedData_v1 andeth_signTypedData_v3 are now deprecated. These method can no longer be used. If you were using one of them, you should now useeth_signTypedData_v4 .

Status and Events Improvementâ

After v9, all the status and events in Web3Auth are now typed. This addition significantly improves the developer experience, and reduces chances for runtime error.

Additional Changesâ

Apart from breaking changes, there are lot of enhancements, and quality of life improvement.

Wallet Services Enhancementsâ

In v9, there has been significant enhancements to the Wallet Services plugin. Now, you can programmatically control the modals's visibility. Along with that the build size is reduced by 25%. Please checkout the parameters, and their usage wrt their functions.

TopUp Parametersâ

Name Description receiveWalletAddress? Specifies the recipient's address. By default, it is set to the currently connected address. tokenList? Specifies the tokens to display in the list. By default, all tokens are shown. Use the ticker name to specify which tokens to display, such as[USDC, USDT, ETH] . Please checkout the coverage details forfull list of supported networks and tokens . fiatList? Specifies the available fiat currencies enabled for purchase. Use the currency acronym to define which fiat currencies to display, such as[USD, SGD, INR, JPY] . Please checkout the coverage details forfull list of supported currencies . show Determines whether the checkout UI is displayed. This can be used to programmatically control its visibility.

WalletConnect Scanner Parametersâ

Name Description show Determines whether the Wallet Connect UI is displayed. This can be used to programmatically control its visibility.

Wallet UI Parametersâ

Name Description show Determines whether the wallet UI is displayed. This can be used to programmatically control its visibility. path? Specifies the path for the wallet services.

Functionality To Enable Key Exportâ

Starting v9, the private key export functionality for Web3Auth PnP Web SDK can be disabled from the dashboard. If the key export is disabled, theeth_private_key andsolanaPrivateKey RPC methods would throw error. By default, the key export will be enabled. Edit this page Previous Wagmi Connector Next v7 to v8