

Wagmi Connector for Web3Auth PnP Web SDKs

[@web3auth/web3auth-wagmi-connector](#)

[^](#)

wagmi is a collection of React Hooks containing everything you need to start working with Ethereum. [@web3auth/web3auth-wagmi-connector](#) is a connector for the popular [wagmi](#) library to help you integrate web3auth plug and play packages. It works with both the [@web3auth/no-modal](#) as well as the [@web3auth/modal](#) packages.

This package can be used to initialize [a wagmi client](#) that will seamlessly manage the interaction (wallet connection state and configuration, such as: auto-connection, connectors, and ethers providers) of your dApp with Web3Auth.

This Documentation is based on the 6.0.0

SDK Version. [^](#)

DEMO Checkout the [Modal](#) Example Apps to see how wagmi works with Web3Auth. note This version of wagmi connector is made for [@wagmi/core](#) v2.6.5 .

Installation [^](#)

- npm
- Yarn
- pnpm

npm install --save [@web3auth/web3auth-wagmi-connector](#) yarn add [@web3auth/web3auth-wagmi-connector](#) pnpm add [@web3auth/web3auth-wagmi-connector](#)

Initialization [^](#)

Import the Web3AuthConnector

class from [@web3auth/web3auth-wagmi-connector](#) [^](#)

import

{

Web3AuthConnector

}

from

"[@web3auth/web3auth-wagmi-connector](#)";

Arguments [^](#)

Web3AuthConnectorParams

[^](#)

- Table
- Interface

Parameter Description web3AuthInstance Pass the Web3Auth instance here. It's the mandatory field and accepts IWeb3Auth or IWeb3AuthModal . loginParams? Login Parameters (only meant while using the [@web3auth/no-modal](#) package). It accepts OpenloginLoginParams . modalConfig? Initialisation Parameters (only meant while using the [@web3auth/modal](#) package). It accepts Record . export

interface

Web3AuthConnectorParams

{ web3AuthInstance :

IWeb3Auth

|

IWeb3AuthModal ; loginParams ? :

OpenloginLoginParams ; modalConfig ? :

Record < WALLET_ADAPTER_TYPE ,

ModalConfig

; }

modalConfig

[â](#)

- Table
- Interface

modalConfig: { ADAPTER : { params } }

Parameter Description label Label of the adapter you want to configure. It's a mandatory field which accepts string . showOnModal? Whether to show an adapter in modal or not. Default value is true . showOnDesktop? Whether to show an adapter in desktop or not. Default value is true . showOnMobile? Whether to show an adapter in mobile or not. Default value is true . Additionally, to configure the Openlogin Adapter's each login method, we have the loginMethods? parameter.

Parameter Description loginMethods? To configure visibility of each social login method for the openlogin adapter. It accepts LoginMethodConfig as a value. initModal (params ? :

{ modalConfig ? :

Record < WALLET_ADAPTER_TYPE ,

ModalConfig

; }) :

Promise < void

;

interface

ModalConfig

extends

BaseAdapterConfig

{ loginMethods ? :

LoginMethodConfig ; }

interface

BaseAdapterConfig

{ label :

string ; showOnModal ? :

boolean ; showOnMobile ? :

boolean ; showOnDesktop ? :

boolean ; } loginMethods: { label: { params } }

In loginMethods , you can configure the visibility of each social login method for the openlogin adapter. The social login is corresponded by the label parameter. Below is the table indicating the different params available for customization.

For labels you can choose between these options: google , facebook , twitter , reddit , discord , twitch , apple , line , github , kakao , linkedin , weibo , wechat , email_passwordless

- Table
- Type Declaration

params

Parameter Description name? Display Name. It accepts string as a value. description? Description for the button. If provided, it renders as a full length button. else, icon button. It accepts string as a value. logoHover? Logo to be shown on mouse hover. It accepts string as a value. logoLight? Light logo for dark background. It accepts string as a value. logoDark? Dark logo for light background. It accepts string as a value. mainOption? Show login button on the main list. It accepts boolean as a value. Default value is false. showOnModal? Whether to show the login button on modal or not. Default value is true. showOnDesktop? Whether to show the login button on desktop. Default value is true. showOnMobile? Whether to show the login button on mobile. Default value is true. declare

type

LoginMethodConfig

=

Record < string , { /* * Display Name. If not provided, we use the default for openlogin app name :

string ; /* * Description for button. If provided, it renders as a full length button. else, icon button description ? :

string ; /* * Logo to be shown on mouse hover. If not provided, we use the default for openlogin app logoHover ? :

string ; /* * Logo to be shown on dark background (dark theme). If not provided, we use the default for openlogin app logoLight ? :

string ; /* * Logo to be shown on light background (light theme). If not provided, we use the default for openlogin app logoDark ? :

string ; /* * Show login button on the main list/ mainOption ? :

boolean ; /* * Whether to show the login button on modal or not showOnModal ? :

boolean ; /* * Whether to show the login button on desktop showOnDesktop ? :

boolean ; /* * Whether to show the login button on mobile showOnMobile ? :

boolean ; }

;

Usage

Since this package acts like a connector, it basically takes in your whole Web3Auth instance and makes it readable for the wagmi library. While connecting the web3auth web packages, you need to initialize the Web3Auth Instance as mentioned in the [modal docs](#) and [no-modal docs](#) . You can configure this instance with all the options available in Web3Auth Modal package and set it up as you wish.

Modal SDK

While all the parameters can be passed directly to the instance, the only parameters that remain during the initialisation remains (passed on to the initModal() function). You can pass these parameters to the modalConfig object in the Web3AuthConnector class.

modalConfig :

```
{ [ WALLET_ADAPTERS . OPENLOGIN ] :
```

```
{ loginMethods :
```

```
{ google :
```

```
{ name :
```

```
"google login" , logoDark :
```

```
"url to your custom logo which will shown in dark mode" , } , facebook :
```

```
{ ... }, }, }, }
```

Example[â](#)

No Modal SDK[â](#)

While all the parameters can be passed directly to the instance, the only parameters that remain during the login remains (passed on to the `connectTo()` function). You can pass these parameters to the `loginParams` object in the `Web3AuthConnector` class.

warning It is mandatory to pass `loginParams` object while using the connector with Web3Auth Core package. This is because the `connectTo()` function requires params to connect to the adapter/ social login desired by the user. * Google * Facebook * JWT * Auth0 * Reddit * Discord * Twitch * Apple * GitHub * LinkedIn * Twitter * Weibo * Line * Email Passwordless * SMS Passwordless

`loginParams :`

```
{ loginProvider :
```

```
'google' , } loginParams :
```

```
{ loginProvider :
```

```
'facebook' , } loginParams :
```

```
{ loginProvider :
```

```
"jwt" , extraLoginOptions :
```

```
{ id_token :
```

```
"idToken" ,
```

```
// in JWT Format verifierIdField :
```

```
"sub" ,
```

```
// same as your JWT Verifier ID } } loginParams :
```

```
{ loginProvider :
```

```
"jwt" , extraLoginOptions :
```

```
{ verifierIdField :
```

```
"sub" ,
```

```
// same as your JWT Verifier ID domain :
```

```
"https://YOUR-APPLICATION-DOMAIN" ,
```

```
// your Auth0 domain } , } loginParams :
```

```
{ loginProvider :
```

```
'email_passwordless' , extraLoginOptions :
```

```
{ login_hint :
```

```
"hello@web3auth.io" ,
```

```
// email to send the OTP to } , } loginParams :
```

```
{ loginProvider :
```

```
'sms_passwordless' , extraLoginOptions :
```

```
{ login_hint :
```

```
"+65-XXXXXXX" , } } loginParams :
```

```
{ loginProvider :
```

```

'reddit' , } loginParams :
{ loginProvider :
'discord' , } loginParams :
{ loginProvider :
'twitch' , } loginParams :
{ loginProvider :
'apple' , } loginParams :
{ loginProvider :
'github' , } loginParams :
{ loginProvider :
'linkedin' , } loginParams :
{ loginProvider :
'twitter' , } loginParams :
{ loginProvider :
'weibo' , } loginParams :
{ loginProvider :
'line' , }

```

Examples

examples Checkout the various [examples](#) made for the wagmi connector using various UI Kits for better integration and user experience Here are a few examples of a wagmi client using both the `Web3AuthConnector` and the `defaultInjectedConnector` respectively.

Modal SDK

```

// Web3Auth Libraries import
{
Web3AuthConnector
}
from
"@web3auth/web3auth-wagmi-connector" ; import
{
Web3Auth
}
from
"@web3auth/modal" ; import
{
EthereumPrivateKeyProvider
}
from
"@web3auth/ethereum-provider" ; import

```

```

{
CHAIN_NAMESPACES ,
WEB3AUTH_NETWORK
}

from

"@web3auth/base" ; import

{
Chain
}

from

"wagmi/chains" ;

export
default
function
Web3AuthConnectorInstance ( chains :
Chain [ ] )
{ // Create Web3Auth Instance const name =
"My App Name" ; const chainConfig =
{ chainNamespace :
CHAIN_NAMESPACES . EIP155 , chainId :
"0x"
+ chains [ 0 ] . id . toString ( 16 ) , rpcTarget : chains [ 0 ] . rpcUrls . default . http [ 0 ] ,
// This is the public RPC we have added, please pass on your own endpoint while creating an app displayName : chains [ 0 ]
. name , tickerName : chains [ 0 ] . nativeCurrency ?. name , ticker : chains [ 0 ] . nativeCurrency ?. symbol ,
blockExplorerUrl : chains [ 0 ] . blockExplorers ?. default . url [ 0 ]
as
string , } ;

const privateKeyProvider =
new
EthereumPrivateKeyProvider ( { config :
{ chainConfig }
} ) ;

const web3AuthInstance =
new
Web3Auth ( { clientId :
"BPi5PB_UiIZ-cPz1GtV5i1I2iOSOHuimiXBI0e-Oe_u6X3oVAbCiAZOTEBtTXw4tsluTITPqA8zMsfxIKMjiqNQ" , chainConfig ,
privateKeyProvider , uiConfig :
{ appName : name , loginMethodsOrder :
[ "github" ,

```

```

"google" ], defaultLanguage :
"en" , modalZIndex :
"2147483647" , logoLight :
"https://web3auth.io/images/w3a-L-Favicon-1.svg" , logoDark :
"https://web3auth.io/images/w3a-D-Favicon-1.svg" , uxMode :
"redirect" , mode :
"light" , } , web3AuthNetwork :
WEB3AUTH_NETWORK . SAPPHIRE_MAINNET , enableLogging :
true , } ) ;
return
Web3AuthConnector ( { web3AuthInstance , } ) ; }

```

No Modal SDK[â](#)

```

// Web3Auth Libraries import
{
Web3AuthConnector
}
from
"@web3auth/web3auth-wagmi-connector" ; import
{
Web3AuthNoModal
}
from
"@web3auth/no-modal" ; import
{
EthereumPrivateKeyProvider
}
from
"@web3auth/ethereum-provider" ; import
{
OpenloginAdapter
}
from
"@web3auth/openlogin-adapter" ; import
{
CHAIN_NAMESPACES ,
UX_MODE ,
WEB3AUTH_NETWORK

```

```

}

from
"@web3auth/base" ; import
{
Chain
}

from
"wagmi/chains" ;

const name =
"My App Name" ; const iconUrl =
"https://web3auth.io/docs/contents/logo-ethereum.png" ;

export
default
function
Web3AuthConnectorInstance ( chains :
Chain [ ] )
{ // Create Web3Auth Instance
const chainConfig =
{ chainNamespace :
CHAIN_NAMESPACES . EIP155 , chainId :
"0x"
+ chains [ 0 ] . id . toString ( 16 ) , rpcTarget : chains [ 0 ] . rpcUrls . default . http [ 0 ] ,
// This is the public RPC we have added, please pass on your own endpoint while creating an app displayName : chains [ 0 ]
. name , tickerName : chains [ 0 ] . nativeCurrency ?. name , ticker : chains [ 0 ] . nativeCurrency ?. symbol ,
blockExplorerUrl : chains [ 0 ] . blockExplorers ?. default . url [ 0 ]
as
string , } ;

const privateKeyProvider =
new
EthereumPrivateKeyProvider ( { config :
{ chainConfig }
} ) ;

const web3AuthInstance =
new
Web3AuthNoModal ( { clientId :
"BPi5PB_UilZ-cPz1GtV5i1I2iOSOHuimiXBI0e-Oe_u6X3oVAbCiAZOTEBtTXw4tsluTITPqA8zMsfxIKMjiqNQ" ,
privateKeyProvider , web3AuthNetwork :
WEB3AUTH_NETWORK . SAPPHIRE_MAINNET , } ) ;

// Add openlogin adapter for customisations const openloginAdapterInstance =

```


new

OpenloginAdapter ({ adapterSettings :

{ uxMode :

UX_MODE . REDIRECT , whiteLabel :

{ appName : name , logoLight : iconUrl , logoDark : iconUrl , defaultLanguage :

"en" , mode :

"light" ,

// whether to enable dark mode. defaultValue: false } , } , }) ; web3AuthInstance . configureAdapter (openloginAdapterInstance) ;

return

Web3AuthConnector ({ web3AuthInstance , loginParams :

{ loginProvider :

"google" , } , }) ; }[Edit this page](#) [Previous Solflare Wallet](#) [Next Web3Auth PnP Android SDK](#)