use-react-wallet @openmev-sdk

github.com/manifoldfinance/openmev-sdk/tree/master/packages/use-react-wallet

A React hook for connecting and interacting with Web3 Wallet Providers via OpenMEV RPC or any other RPC

Overview

Utilizes

:

- Web3Modal
- Zustand

Connect or Disconnect button

const ConnectWalletButton = () => { const { account, connect, disconnect } = useWallet(); return (<> {!account ? (connect()}>Connect Wallet) : (disconnect()}>Disconnect Wallet)}); };

Connecting

The connect

function passes along an optional config to a Web3Modal instance for additional customization.

You can use the account information from useWallet anywhere inside your React app, without any extra set up.

const UserAddress = () => { const { account } = useWallet(); if (!account) return null; return <>{account}; };

Transaction Signing

To run a transaction or sign a message, use the provider

object returned by the hook for connected wallets.

This is a standard Ethers.js Provider. Use the @openmev

/ethers-provider if you want OpenMEV support

const SignMessageButton = () => { const { account, provider } = useWallet(); if (!account) return null; const signMessage = async () => { const signature = await provider.getSigner().signMessage("Hello!"); console.log(signature); } return

Sign Message; }

Experimental

useSafeTimeout

is a utility Hook that allows you to safely call setTimeout

and clearTimeout

within a component, ensuring that all timeouts are cleared when the component unmounts.

Usage

{([]) => { const { safeSetTimeout, safeClearTimeout } = useSafeTimeout(); let timeoutId = null;

```
const handleOnClick = () => {
timeoutId = safeSetTimeout(() => window.alert('hello!'), 5000);
};

const cancelTimeout = () => {
 safeClearTimeout(timeoutId);
};

return (
<>
<Button onClick={handleOnClick}>Click me</Button>
<Button onClick={cancelTimeout}>Cancel timeout</Button>

);

}}
```