How to create and use a Biconomy Nexus account with permissionless.js

Biconomy Nexus Smart Account is a smart account building on the core concepts of ERC-7579. You can use Nexus with plugins such as session keys, and even write your own plugins.

Steps

Import the required packages

```

import{ createSmartAccountClient }from"permissionless" import{ createPublicClient, getContract, http, parseEther }from"viem" import{ createPimlicoClient }from"permissionless/clients/pimlico" import{ sepolia }from"viem/chains"

```

Create the clients

First we must create the public, (optionally) pimlico paymaster clients that will be used to interact with the Nexus account.

```

exportconstpublicClient=createPublicClient({ transport:http("https://rpc.ankr.com/eth_sepolia"), })

exportconstpimlicoClient=createPimlicoClient({ transport:http("https://api.pimlico.io/v2/sepolia/rpc?apikey=API_KEY"), entryPoint: { address: entryPoint06Address, version:"0.6", }, })

```

Create the signer

Nexus accounts can work with a variety of signing algorithms such as ECDSA, passkeys, and multisig. In permissionless.js, the default Nexus account validates ECDSA signatures.Any signer can be used as a signer for the Nexus account.

For example, to create a signer based on a private key:

```

import{ privateKeyToAccount }from"viem/accounts" import{ entryPoint06Address }from"viem/account-abstraction" import{ toBiconomySmartAccount, toNexusSmartAccount }from"permissionless/accounts"

constowner=privateKeyToAccount("0xPRIVATE_KEY")

```

Create the Nexus account

With a signer, you can create a Nexus account as such:

```

constnexusAccount=awaittoNexusSmartAccount({ client: publicClient, owners: [owner], version:"1.0.0", index:0n,// optional address:"0x...",// optional, only if you are using an already created account })

```

The Nexus account address is computed deterministically from the signer, but you can optionally pass anindex to create any number of different accounts using the same signer. You can also pass anaddress to use an already created Nexus account.

Create the smart account client

The smart account client is a permissionless.js client that is meant to serve as an almost drop-in replacement for viem'swalletClient .

```
```

```
constsmartAccountClient=createSmartAccountClient({ account: nexusAccount, chain: sepolia, bundlerTransport:http("https://api.pimlico.io/v2/sepolia/rpc?apikey=API_KEY"), paymaster: pimlicoClient, userOperation: { estimateFeesPerGas:async()=>(awaitpimlicoClient.getUserOperationGasPrice()).fast, }, })
```

Send a transaction

Transactions using permissionless.js simply wrap around user operations. This means you can switch to permissionless.js from your existing viem EOA codebase with minimal-to-no changes.

```
consttxHash=awaitsmartAccountClient.sendTransaction({ to:"0xd8da6bf26964af9d7eed9e03e53415d37aa96045", value:parseEther("0.1"), })
```

This also means you can also use viem Contract instances to transact without any modifications.

```
constnftContract=getContract({ address:"0xFBA3912Ca04dd458c843e2EE08967fC04f3579c2", abi: nftAbi, client: { public: publicClient, wallet: smartAccountClient, }, })

consttxHash=awaitnftContract.write.mint()
```

You can also send an array of transactions in a single batch.

```
constuserOpHash=awaitsmartAccountClient.sendUserOperation({ calls: [ { to:"0xd8da6bf26964af9d7eed9e03e53415d37aa96045", value:parseEther("0.1"), data:"0x", }, { to:"0x1440ec793aE50fA046B95bFeCa5aF475b6003f9e", value:parseEther("0.1"), data:"0x1234", }, ], })
```