How to use an ERC-7579 compatible smart account with permissionless.js

ERC-7579 defines a standard for modular smart account interfaces. It also defines behavior for interoperability with minimal restrictions for accounts and modules.

Currently Safe and Kernel are the only smart accounts that implements ERC-7579.

For this guide, we will use the Safe smart accounts as an example. If you would like to find out more about the Safe smart account, you can check out theSafe-specific guide .

This guide will show you how to create and use a ERC-7579 compatible smart account with permissionless.js.

Steps

Import the required packages

```
import{ owner }from"../owner" import{ createSmartAccountClient }from"permissionless" import{ sepolia }from"viem/chains" import{ encodePacked, http }from"viem" import{ erc7579Actions }from"permissionless/actions/erc7579" import{ createPublicClient }from"viem" import{ createPimlicoClient }from"permissionless/clients/pimlico" import{ entryPoint07Address }from"viem/account-abstraction" import{ toSafeSmartAccount }from"permissionless/accounts"
```

Create the clients

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

```
constapiKey="YOUR_PIMLICO_API_KEY" constbundlerUrl=https://api.pimlico.io/v2/sepolia/rpc?apikey={apiKey}

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

constpimlicoClient=createPimlicoClient({ transport:http(bundlerUrl), entryPoint: { address: entryPoint07Address, version:"0.7", }, })
```

Create the SafeAccount

For a full list of options for creating a SafeAccount, take a look at the reference documentation page fortoSafeSmartAccount . You can also pass anaddress to use an already created SafeAccount.

```
constsafeAccount=awaittoSafeSmartAccount({ client: publicClient, owners: [owner], version:"1.4.1", entryPoint: { address: entryPoint07Address, version:"0.7", }, safe4337ModuleAddress:"0x7579EE8307284F293B1927136486880611F20002", erc7579LaunchpadAddress:"0x7579011aB74c46090561ea277Ba79D510c6C00ff", attesters: ["0x000000333034E9f539ce08819E12c1b8Cb29084d"],// This address belongs to Rhinestone. By designating them as attesters, you authorize that only modules explicitly approved by Rhinestone can be installed on your safe. attestersThreshold:1, })
```

The Safe account requires a newsafe4337ModuleAddress &erc7579LaunchpadAddress for it to be 7579 compatible. The address0x000000333034E9f539ce08819E12c1b8Cb29084d belongs to Rhinestone. By designating them as attesters, you authorize that only modules explicitly approved by Rhinestone can be installed on your safe.

Create the smart account client and extend it with the ERC7579 actions

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: safeAccount, chain: sepolia,
bundlerTransport:http(bundlerUrl), paymaster: pimlicoClient, userOperation: { estimateFeesPerGas:async()=>{
return(awaitpimlicoClient.getUserOperationGasPrice()).fast }, }, }).extend(erc7579Actions())
```

Interact with the 7579 actions

You can install a module on the Safe account using theinstallModule action.

```
constownableExecutorModule="0xc98B026383885F41d9a995f85FC480E9bb8bB891"
constmoduleData=encodePacked(["address"], ["0xd8dA6BF26964aF9D7eEd9e03E53415D37aA96045"])
constuserOpHash=awaitsmartAccountClient.installModule({ type:"executor", address: ownableExecutorModule, context:
moduleData, })

constreceipt=awaitpimlicoClient.waitForUserOperationReceipt({ hash: userOpHash })
```

InstallModule returns user operation hash, not transaction hash, you must usewaitForUserOperationReceipt to wait for the user operation to be included on-chain. You can also call all other ERC7579 actions, examplesupportsExecutionMode .

```
constisExecutionModeSupported=awaitsmartAccountClient.supportsExecutionMode({ type:"delegatecall",
revertOnError:true, selector:"0x", context:"0x", })
```