D4D4D4;--ch-t-background: #1E1E1E;--ch-t-lighter-inlineBackground: #1e1e1ee6;--ch-t-editor-background: #1E1E1E;--ch-t-editor-foreground: #D4D4D4;--ch-t-editor-rangeHighlightBackground: #ffffff0b;--ch-t-editor-infoForeground: #3794FF;--ch-t-editor-selectionBackground: #264F78;--ch-t-focusBorder: #007FD4;--ch-t-tab-activeBackground: #1E1E1E;--ch-t-tab-activeForeground: #ffffff;--ch-t-tab-inactiveBackground: #2D2D2D;--ch-t-tab-inactiveForeground: #ffffff80;--ch-t-tab-border: #252526;--ch-t-tab-activeBorder: #1E1E1E;--ch-t-editorGroup-border: #444444;--ch-t-editorGroupHeader-tabsBackground: #252526;--ch-t-editorLineNumber-foreground: #858585;--ch-t-input-background: #3C3C3C;--ch-t-input-foreground: #D4D4D4;--ch-t-icon-foreground: #C5C5C5;--ch-t-sideBar-background: #252526;--ch-t-sideBar-foreground: #D4D4D4;--ch-t-sideBar-border: #252526;--ch-t-list-activeSelectionBackground: #094771;--ch-t-list-activeSelectionForeground: #fffffe;--ch-t-list-hoverBackground: #2A2D2E; }

AI agent swaps on CoW Swap

CoW swap ensures best prices and fastest execution and minimizes MEV.

You can find a working code example to run locally in ourAI agent with Safe Smart Account CoW Swap example repository(opens in a new tab) .

Here is a quick guide to get you up and running:

Requirements

- A deployed Safe Smart Account
- The AI agent is a signer on the Safe
- This example assumes, that the threshold of the Safe Smart Account is one, so the AI agent can sign autonomously.
- If you require more signatures, you have to collect those signatures programmatically of with theSafe Wallet(opens in a new tab)
- .

Let your AI agent send an intent

Setup the Safe Smart Account

Your Safe Smart Account should be deployed. Now, initialize an instance with theProtocol Kit :

_10 import Safe from "@safe-global/protocol-kit"; _10 _10 const preExistingSafe = await Safe.init({ _10 provider: RPC_URL,

```
_10 signer: AGENT_PRIVATE_KEY, _10 safeAddress: SAFE_ADDRESS, _10 });
```

Send swap intent

Now, you can use the CoW Swap SDK to assemble a transaction that you can sign and execute with your Safe Smart Account. The swap will then be executed.

Please be aware that the CoW Swap's SDK uses Ethers, while Safe's SDK use viem. You will see some warnings in the logs, but the code works nonetheless.

In this example, we buy COW and pay with WETH.

```
_78 import { _78 SwapAdvancedSettings, _78 TradeParameters, _78 TradingSdk, _78 SupportedChainId, _78 OrderKind,
_78 SigningScheme, _78 } from "@cowprotocol/cow-sdk"; _78 import { VoidSigner } from "@ethersproject/abstract-signer";
_78 import { JsonRpcProvider } from "@ethersproject/providers"; _78 _78 const traderParams = { _78 chainId:
SupportedChainId.SEPOLIA, _78 signer: new VoidSigner( _78 smartContractWalletAddress: SAFE_ADDRESS, _78 new
JsonRpcProvider("https://sepolia.gateway.tenderly.co") _78 ), _78 appCode: "awesome-app", _78 }; _78 _78 const cowSdk
= new TradingSdk(traderParams, { logs: false }); _78 _78 const parameters: TradeParameters = { _78 kind:
OrderKind.SELL, _78 sellToken: WETH_ADDRESS, _78 sellTokenDecimals: 18, _78 buyToken: COW_ADDRESS, _78
buyTokenDecimals: 18, _78 amount: INPUT_AMOUNT, _78 }; _78 _78 const advancedParameters: SwapAdvancedSettings
= { _78 quoteRequest: { _78 // Specify the signing scheme _78 signingScheme: SigningScheme.PRESIGN, _78 }, _78 }; _78
_78 const orderId = await cowSdk.postSwapOrder(parameters, advancedParameters); _78 _78 console.log(Order ID:
[{orderId}]); _78 _78 const preSignTransaction = await cowSdk.getPreSignTransaction({ _78 orderId, _78 account:
smartContractWalletAddress, _78 }); _78 _78 const customChain = defineChain({ _78 ...sepolia, _78 name: "custom chain",
_78 transport: http(RPC_URL), _78 }); _78 _78 const publicClient = createPublicClient({ _78 chain: customChain, _78
transport: http(RPC_URL), _78 }); _78 _78 const safePreSignTx: MetaTransactionData = { _78 to: preSignTransaction.to,
_78 value: preSignTransaction.value, _78 data: preSignTransaction.data, _78 operation: OperationType.Call, _78 }; _78 _78
const safeTx = await preExistingSafe.createTransaction({ _78 transactions: [safePreSignTx], _78 onlyCalls: true, _78 }); _78
_78 // You might need to collect more signatures here _78 _78 const txResponse = await
preExistingSafe.executeTransaction(safeTx); _78 console.log(Sent tx hash: [{txResponse.hash}]); _78 console.log("Waiting for the
tx to be mined"); _78 await publicClient.waitForTransactionReceipt({ _78 hash: txResponse.hash as 0x{string}, _78 });
```

Next steps

Now, where your AI agent can execute trades autonomously, you are free to use this power as you like. You can find more specific information in theCoW Swap Trading SDK docs(opens in a new tab).

If you have a technical question about Safe Smart Accounts, feel free to reach out onStack Exchange(opens in a new tab) with the safe-core tag.

Was this page helpful?

Report issue