

Class: Multiplexer

Multiplexer for conditional orders - using ComposableCoW !

This class provides functionality to:

- Generate a merkle tree of conditional orders
- Generate proofs for all orders in the merkle tree
- Save proofs, with the ability to omit / skip specific conditional orders
- Support for passing an optional upload function to upload the proofs to a decentralized storage network

Constructors

constructor

•new Multiplexer (chain ,orders? ,root? ,location?) [Multiplexer](#)

Parameters

Name	Type	Default value	Description
chain	SupportedChainId	undefined	The chainId for where we're using ComposableCoW .
orders?	Orders	undefined	An optional array of conditional orders to initialize the merkle tree with.
root?	string	undefined	An optional root to verify against.
location	ProofLocation	ProofLocation.PRIVATE	The location of the proofs for the conditional orders.

Returns

[Multiplexer](#)

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:42

Properties

chain

•chain :[SupportedChainId](#)

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:29

ctx

•Private Optional ctx :string

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:34

location

•location :[ProofLocation](#)

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:30

orders

•Private orders [Orders](#) = {}

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:32

tree

•Private Optional tree :StandardMerkleTree

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:33

orderTypeRegistry

•Static orderTypeRegistry :Record [ConditionalOrder](#) <unknown ,unknown
= {}

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:27

Accessors

orderIds

•get orderIds ():string []

Get all the conditional order ids in the multiplexer.

Returns

string []

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:226

root

•get root ():string

Returns

string

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:230

Methods

add

►add <T ,P
(order):void

Add a conditional order to the merkle tree.

Type parameters

Name T P

Parameters

Name Type Description order [ConditionalOrder](#) <T ,P

The order to add to the merkle tree.

Returns

void

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:168

dumpProofs

►dumpProofs (filter?):string

The primary entry point for dumping the proofs and parameters for the conditional orders.

This is to be used by watchtowers / indexers to store the proofs and parameters for the conditional orders off-chain. The encoding returned by this method mayNOT contain all proofs and parameters, depending on the filter provided, and therefore should not be used to rehydrate the multiplexer from a user's perspective.

Parameters

Name	Type	Description
filter?	(v :string []) =>boolean	getProofs

Returns

string

A JSON-encoded string of the proofs and parameters for the conditional orders.

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:362

dumpProofsAndParams

►dumpProofsAndParams (filter?): [ProofWithParams](#) []

Parameters

Name	Type	Description
filter?	(v :string []) =>boolean	

Returns

[ProofWithParams](#) []

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:366

encodeToABI

►encodeToABI (filter?):string

ABI-encode the proofs and parameters for the conditional orders in the merkle tree.

Parameters

Name	Type	Description
filter?	(v :string []) =>boolean	getProofs

Returns

string

ABI-encoded data for the ProofStruct .

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:408

encodeToJSON

► encodeToJSON (filter?):string

JSON-encode the proofs and parameters for the conditional orders in the merkle tree.

Parameters

Name	Type	Description
filter?	(v :string []) =>boolean	getProofs

Returns

string

The JSON-encoded data for storage off-chain.

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:417

getById

► getById (id):[ConditionalOrder](#) <unknown ,unknown

Accessor for a given conditional order in the multiplexer.

Parameters

Name	Type	Description
id	string	The id of theConditionalOrder to retrieve.

Returns

[ConditionalOrder](#) <unknown ,unknown

AConditionalOrder with the givenid .

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:210

getByIndex

► getByIndex (i):[ConditionalOrder](#) <unknown ,unknown

Accessor for a given conditional order in the multiplexer.

Parameters

Name	Type	Description
i	number	The index of theConditionalOrder to retrieve.

Returns

[ConditionalOrder](#) <unknown ,unknown

AConditionalOrder at the given index.

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:219

getOrGenerateTree

► getOrGenerateTree ():StandardMerkleTree

Retrieve the merkle tree of orders, or generate it if it doesn't exist.

CAUTION : Developers of the SDK should prefer to use this method instead of generating the merkle tree themselves. This method makes use of caching to avoid generating the merkle tree needlessly.

Returns

StandardMerkleTree

The merkle tree for the current set of conditional orders.

Throws

If the merkle tree cannot be generated.

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:243

getProofs

►getProofs (filter?) :[ProofWithParams](#) []

Get the proofs with parameters for the conditional orders in the merkle tree.

Parameters

Name	Type	Description
filter?	(v :string []) =>boolean	A function that takes a conditional order and returns a boolean indicating whether the order should be included in the proof.

Returns

[ProofWithParams](#) []

An array of proofs and their order's parameters for the conditional orders in the merkle tree.

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:377

prepareProofStruct

►prepareProofStruct (location? ,filter? ,uploader?):Promise <ProofStruct

The primary entry point for dapps integrating withComposableCoW to generate the proofs and parameters for the conditional orders.

After populating the multiplexer with conditional orders, this method can be used to generate the proofs and parameters for the conditional orders. The returnedProofStruct can then be used withsetRoot orsetRootWithContext on aComposableCoW - enabled Safe.

Parameters

Name	Type	Description
location	ProofLocation	-
filter?	(v :string []) =>boolean	getProofs
uploader?	(offChainEncoded :string) =>Promise <string	

-

Returns

Promise <ProofStruct

The ABI-encodedProofStruct forsetRoot andsetRootWithContext .

Parma

locFn A function that takes the off-chain encoded input, and returns thelocation for theProofStruct , and thedata for theProofStruct .

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:280

remove

►remove (id):void

Remove a conditional order from the merkle tree.

Parameters

Name	Type	Description
id	string	The id of theConditionalOrder to remove from the merkle tree.

Returns

void

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:179

reset

►reset ():void

A helper to reset the merkle tree.

Returns

void

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:424

toJSON

►toJSON ():string

Serialize the multiplexer to JSON.

This will include all state necessary to reconstruct the multiplexer, including the root.

Returns

string

The JSON representation of the multiplexer, including the root but excluding the merkle tree.

Remarks

This willNOT include the merkle tree.

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:143

update

►update (id ,updater):void

Update a given conditional order in the merkle tree.

Parameters

Name	Type	Description
id	string	The id of theConditionalOrder to update.
updater	(order ConditionalOrder <unknown> ,ctx? :string) => ConditionalOrder <unknown> ,unknown	

A function that takes the existingConditionalOrder and context, returning an updatedConditionalOrder .

Returns

void

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:189

decodeFromJSON

►decodeFromJSON (s): [ProofWithParams](#) []

The primary method for watch towers to use when deserializing the proofs and parameters for the conditional orders.

Parameters

Name	Type	Description
s	string	The serialized proofs with parameters for consumption by watchtowers / indexers.

Returns

[ProofWithParams](#) []

TheProofWithParams array.

Throws

If theProofWithParams array cannot be deserialized.

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:262

fromJSON

►fromJSON (s): [Multiplexer](#)

Given a serialized multiplexer, create the multiplexer and rehydrate all conditional orders. Integrity of the multiplexer will be verified by generating the merkle tree and verifying the root.

NOTE : Before using this method, you must register all conditional order types usingMultiplexer.registerOrderType .

Parameters

Name	Type	Description
s	string	The serialized multiplexer.

Returns

[Multiplexer](#)

The multiplexer with all conditional orders rehydrated.

Throws

If the multiplexer cannot be deserialized.

Throws

If the merkle tree cannot be generated.

Throws

If the merkle tree cannot be verified against the root.

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:98

poll

►poll (owner ,p ,chain ,provider ,offChainInputFn?):Promise <[DataStruct ,string]>

Poll a conditional order to see if it is tradeable.

Parameters

Name	Type	Description
owner	string	The owner of the conditional order.
p	ProofWithParams	The proof and parameters.
chain	SupportedChainId	Which chain to use for the ComposableCoW contract.
provider	Provider	An RPC provider for the chain.
offChainInputFn?	(owner :string ,params : ConditionalOrderParams) =>Promise <string	

A function, if provided, that will return the off-chain input for the conditional order.

Returns

Promise <[DataStruct ,string]>

The tradeableGPv2Order.Data struct and thesignature for the conditional order.

Throws

If the conditional order is not tradeable.

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:339

registerOrderType

►registerOrderType (orderType ,conditionalOrderClass):void

Register a conditional order type with the multiplexer.

CAUTION : This is required for usingMultiplexer.fromJSON andMultiplexer.toJSON .

Parameters

Name	Type	Description
orderType	string	The order type to register.
conditionalOrderClass	(...args :any []) => ConditionalOrder <unknown ,unknown	

The class to use for the given order type.

Returns

void

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:435

resetOrderTypeRegistry

►resetOrderTypeRegistry ():void

Reset the order type registry.

Returns

void

Defined in

external/cow-sdk/src/composable/Multiplexer.ts:445 [Previous](#) [CowError](#) [Next](#) [OrderBookApi](#)