Limit order structure

Field Type Inner Solidity type Description salt string uint256 some unique value. It is necessary to be able to create limit orders with the same parameters (so that they have a different hash) makerAsset string address the address of the asset you want to sell (address of a token contract) takerAsset string address the address of the asset you want to buy (address of a token contract) maker string address the address of the limit order creator receiver string address by default contains a zero address, which means that taker asset will be sent to the address of the creator of the limit order. If you set a value, then taker asset will be sent to the specified address allowedSender string address by default contains a zero address, which means that a limit order is available for everyone to fill. If you set a value, then the limit order will be available for execution only for the specified address (private limit order) makingAmount string uint256 amount of maker asset takingAmount string uint256 amount of taker asset interactions string bytes hex bytes string with interactions meta fields concat(makerAssetData, takerAssetData, getMakingAmount, getTakingAmount, predicate, permit, preIntercation, postInteraction) offsets string uint256 every 32's bytes represents offset of the n'ths interaction

interactions

meta fields

Index Field Type Inner Solidity type Description 0 makerAssetData string bytes the technical info about a maker asset and its amount 1 takerAssetData string bytes the technical info about a taker asset and its amount 2 getMakingAmount string bytes technical information to get the amount of the maker asset 3 getTakingAmount string bytes technical information to get the amount of the taker asset 4 predicate string bytes a predicate call data. See morePredicate docs 5 permit string bytes a permit (EIP-2612) call data. Could be built usingutility library 6 preInteraction string bytes What to do before the transfer. A call data for InteractiveNotificationReceiver. See moreInteraction receiver docs 7 postInteraction string bytes What tot do after the transfer has been made. a call data for InteractiveNotificationReceiver. See moreInteraction receiver docs

Example:

await

new

LimitOrderBuilder (...) . buildLimitOrder ({ makerAssetAddress :

'0x000000000000000000000000000000000000000A' , takerAssetAddress :

'0x000000000000000000000000000000000000000B' , makerAddress :

'0x0000000000000000000000000000000000000001' , receiver :

'0x0000000000000000000000000000000000000002' , allowedSender :

'0x0000000000000000000000000000000000000003' , makingAmount :

'100' , takingAmount :

'200' , getMakingAmount :

'0x11' , getTakingAmount :

'0x2222' , predicate :

'0x333333' , permit :

'0x44444444' , preInteraction :

'0x5555555555' , postInteraction :

'0x666666666666' , }) ;

// { // salt: '390590399942', // makerAsset: '0x000000000000000000000000000000000000000A', // takerAsset: '0x000000000000000000000000000000000000000B', // maker: '0x0000000000000000000000000000000000000001', // receiver: '0x0000000000000000000000000000000000000002', // allowedSender: '0x0000000000000000000000000000000000000003', // makingAmount: '100', // takingAmount: '200', // offsets: '566158801043199617334225446602998088782951927103916713681406090280960', // interactions: '0x112222333333444444445555555555666666666666' // } Where the each offset is iteration bytes offset:

// interactions: '0x112222333333444444445555555555666666666666' // ^ ^ ^ ^ ^ ^ ^ // 1 2 4 7 11 16 21

// Note: offset is bytes offset. Not a hex offset. // // > Buffer.from('112222333333444444445555555555666666666666', 'hex') /// ^ ^ ^ ^ ^ ^ ^ // 1 2 4 7 11 16 21

(0n

<<

(0n))

// makerAssetData, length: 0, start: 0, end: 0 +

(0n

<<

(32n

*

1n))

// takerAssetData, length: 0, start: 0, end: 0 +

(1n

<<

(32n

*

2n))

// getMakingAmount, length: 1, start: 0, end: 1 +

(3n

<<

(32n

*

3n))

// getTakingAmount, length: 2, start: 2, end: 3 +

(6n

<<

(32n

*

4n))

// predicate, length: 3, start: 4, end: 6 +

(10n

<<

(32n

*

5n))

// permit, length: 4, start: 7, end: 10 +

(15n

<<

(32n

*

6n))

// preInteraction, length: 5, start: 11, end: 15 +

(21n

<<

(32n

*

7n))

// postInteraction, length: 6, start: 16, end: 21

566158880104319961733422544660299808878295192710391671368140609028096

0x150000000f0000000a000000060000000300000001000000000000000 // ^ ^ ^ ^ ^ ^ ^ // 21 15 10 6 3 1 0 Edit this page Previous Create a limit order Next Fill a limit order