LimitOrderProtocol

Quick links

Derives

- EIP712
- EthReceiver
- OnlyWethReceiver
- PredicateHelper
- SeriesEpochManager

Structs

Order

```
struct Order {
  uint256 salt;
  Address maker;
  Address receiver;
  Address makerAsset;
  Address takerAsset;
  uint256 makingAmount;
  uint256 takingAmount;
  MakerTraits makerTraits;
}
```

Functions

constructor

```
constructor(
  contract IWETH weth
)
```

Parameters:

Name	Type	Description
`weth`	contract IWETH	

invalidatorForOrderRFQ

```
function bitInvalidatorForOrder(
  address maker,
  uint256 slot
) external view returns(uint256 result)
```

Returns bitmask for double-spend invalidators based on lowest byte of order.info and filled quotes

Parameters:

Name	Type	Description
`maker`	address	Maker address
`slot`	uint256	Slot number to return bitmask for

Return Values:

Name	Type	Description
`result`	uint256	Each bit represents whether corresponding was already invalidated

remainingInvalidatorForOrder

```
function remainingInvalidatorForOrder(
  address maker,
  bytes32 orderHash
) external view returns(uint256)
```

Returns bitmask for double-spend invalidators based on lowest byte of order.info and filled quotes

Parameters:

Name	Type	Description
`maker`	address	Maker address
`orderHash`	bytes32	Hash of the order

Return Values:

Name	Type	Description
`remaining`	uint256	Remaining amount of the order

rawRemainingInvalidatorForOrder

```
function rawRemainingInvalidatorForOrder(
  address maker,
  bytes32 orderHash
) external view returns(uint256)
```

Returns bitmask for double-spend invalidators based on lowest byte of order.info and filled quotes

Parameters:

Name	Type	Description
`maker`	address	Maker address
`orderHash`	bytes32	Hash of the order

Return Values:

Name	Type	Description
`remainingRaw`	uint256	Remaining amount of the order plus 1 if order was partially filled, otherwise 0

simulate

```
function simulate(
  address target,
  bytes calldata data
) external
```

Delegates execution to custom implementation. Could be used to validate if `transferFrom` works properly The function always reverts and returns the simulation results in revert data.

Parameters:

Name	Type	Description
`target`	address	Addresses that will be delegated
`data`	bytes	Data that will be passed to delegatee

cancelOrder

```
function cancelOrder(
  MakerTraits makerTraits,
  bytes32 orderHash
) public
```

Cancels orders' quotes

Parameters:

Name	Type	Description
`makerTraits`	MakerTraits	Orders makerTraits
`orderHash`	bytes32	Hashes of the orders to cancel

cancelOrders

```
function cancelOrders(
  MakerTraits[] calldata makerTraits,
  bytes32[] calldata orderHashes
) external
```

Cancels orders' quotes

Parameters:

Name	Type	Description
`makerTraits`	MakerTraits	Orders makerTraits
`orderHashes`	bytes32	Hashes of the orders to cancel

bitsInvalidateForOrder

```
function bitsInvalidateForOrder(
  MakerTraits makerTraits,
  uint256 additionalMask
) external
```

Cancels all quotes of the maker (works for bit-invalidating orders only)

Parameters:

Name	Type	Description
`makerTraits`	MakerTraits	Order makerTraits
`additionalMask`	uint256	Additional bitmask to invalidate orders

hashOrder

```
function hashOrder(
  Order calldata order
) external view returns(bytes32)
```

Returns order hash, hashed with limit order protocol contract EIP712

Parameters:

Name	Type	Description
`order`	Order	Order

Return Values:

Name	Type	Description
`hash`	bytes32	Hash of the order

checkPredicate

```
function checkPredicate(
  bytes calldata predicate
) public view returns(bool)
```

See {IOrderMixin-checkPredicate}

Parameters:

Name	Type	Description
`predicate`	bytes	

Return Values:

Name	Type	Description
`success`	bool	

fillOrder

```
function fillOrder(
    Order calldata order,
    bytes32 r,
    bytes32 vs,
    uint256 amount,
    TakerTraits takerTraits
) external payable returns(uint256,uint256,bytes32)
```

Fills order's quote, fully or partially (whichever is possible)

Parameters:

Name	Type	Description
`order`	Order	Order quote to fill
`r`	bytes32	R component of signature
`vs`	bytes32	VS component of signature
`amount`	uint256	Taker amount to fill
`takerTraits`	TakerTraits	Specifies threshold as maximum allowed takingAmount when takingAmount is zero, otherwise specifies minimum allowed makingAmount. Top-most bit specifies whether taker wants to skip maker's permit.

Return Values:

Name	Type	Description
`makingAmount`	uint256	Actual amount transferred from maker to taker
`takingAmount`	uint256	Actual amount transferred from taker to maker
`orderHash`	bytes32	Hash of the filled order

fillOrderExt

```
function fillOrderExt(
    Order calldata order,
    bytes32 r,
    bytes32 vs,
    uint256 amount,
    TakerTraits takerTraits,
    bytes calldata extension
) external payable returns(uint256,uint256,bytes32)
```

See {IOrderMixin-fillOrderExt}

Parameters:

Name	Type	Description
`order`	Order	
`r`	bytes32	
`vs`	bytes32	
`amount`	uint256	
`takerTraits`	TakerTraits	
`extension`	bytes	

Return Values:

Name	Type	Description
`makingAmount`	uint256	
`takingAmount`	uint256	
`orderHash`	bytes32	

fillOrderTo

```
function fillOrderTo(
  Order calldata order,
  bytes32 r,
  bytes32 vs,
  uint256 amount,
  TakerTraits takerTraits,
  address target,
  bytes calldata interaction
) external payable returns(uint256,uint256,bytes32)
```

Same as `fillOrder` but allows to specify funds destination instead of `msg.sender`

Parameters:

Name	Type	Description
`order`	Order	Order quote to fill
`r`	bytes32	R component of signature
`vs`	bytes32	VS component of signature
`amount`	uint256	Taker amount to fill

Name	Type	Description
`takerTraits`	TakerTraits	Specifies threshold as maximum allowed takingAmount when takingAmount is zero, otherwise specifies minimum allowed makingAmount. Top-most bit specifies whether taker wants to skip maker's permit.
`target`	address	Address that will receive swap funds
`interaction`	bytes	A call data for Interactive. Taker may execute interaction after getting maker assets and before sending taker assets.

Return Values:

Name	Type	Description
`makingAmount`	uint256	Actual amount transferred from maker to taker
`takingAmount`	uint256	Actual amount transferred from taker to maker
`orderHash`	bytes32	Hash of the filled order

fillOrderToExt

```
function fillOrderToExt(
    Order calldata order,
    bytes32 r,
    bytes32 vs,
    uint256 amount,
    TakerTraits takerTraits,
    address target,
    bytes calldata interaction,
    bytes calldata extension
) public payable returns(uint256,uint256,bytes32)
```

See {IOrderMixin-fillOrderToExt}

Parameters:

Name	Type	Description
`order`	Order	
`r`	bytes32	
`vs`	bytes32	
`amount`	uint256	
`takerTraits`	TakerTraits	
`target`	address	

Name	Type	Description
`interaction`	bytes	
`extension`	bytes	

Return Values:

Name	Type	Description
`makingAmount`	uint256	
`takingAmount`	uint256	
`orderHash`	bytes32	

fillOrderToWithPermit

```
function fillOrderToWithPermit(
    Order calldata order,
    bytes32 r,
    bytes32 vs,
    uint256 amount,
    TakerTraits takerTraits,
    address target,
    bytes calldata interaction,
    bytes calldata permit
) external returns(uint256,uint256,bytes32)
```

Same as `fillOrderTo` but calls permit first. It allows to approve token spending and make a swap in one transaction. Also allows to specify funds destination instead of `msg.sender`

Parameters:

Name	Type	Description
`order`	Order	Order quote to fill
`r`	bytes32	R component of signature
`vs`	bytes32	VS component of signature
`amount`	uint256	Taker amount to fill
`takerTraits`	TakerTraits	Specifies threshold as maximum allowed takingAmount when takingAmount is zero, otherwise specifies minimum allowed makingAmount. Top-most bit specifies whether taker wants to skip maker's permit.
`target`	address	Address that will receive swap funds
`interaction`	bytes	A call data for Interactive. Taker may execute interaction after getting maker assets and before sending taker assets.

Name	Type	Description
`permit`	bytes	Should contain abi-encoded calldata for `IERC20Permit.permit` call

Return Values:

Name	Type	Description
`makingAmount`	uint256	Actual amount transferred from maker to taker
`takingAmount`	uint256	Actual amount transferred from taker to maker
`orderHash`	bytes32	Hash of the filled order

fillContractOrder

```
function fillContractOrder(
    Order calldata order,
    bytes calldata signature,
    uint256 amount,
    TakerTraits takerTraits,
    address target,
    bytes calldata interaction
) external returns(uint256,uint256,bytes32)
```

Same as `fillOrderTo` but calls permit first. It allows to approve token spending and make a swap in one transaction. Also allows to specify funds destination instead of `msg.sender

Parameters:

Name	Type	Description
`order`	Order	Order quote to fill
`signature`	bytes	Signature to confirm quote ownership
`amount`	uint256	Taker amount to fill
`takerTraits`	TakerTraits	Specifies threshold as maximum allowed takingAmount when takingAmount is zero, otherwise specifies minimum allowed makingAmount. Top-most bit specifies whether taker wants to skip maker's permit.
`target`	address	Address that will receive swap funds
`interaction`	bytes	A call data for Interactive. Taker may execute interaction after getting maker assets and before sending taker assets.

Return Values:

Name	Type	Description
`makingAmount`	uint256	Actual amount transferred from maker to taker
`takingAmount`	uint256	Actual amount transferred from taker to maker
`orderHash`	bytes32	Hash of the filled order

fillContractOrderWithPermit

```
function fillContractOrderWithPermit(
    Order calldata order,
    bytes calldata signature,
    uint256 amount,
    TakerTraits takerTraits,
    address target,
    bytes calldata interaction,
    bytes calldata permit
) external returns(uint256,uint256,bytes32)
```

Same as `fillOrderTo` but calls permit first. It allows to approve token spending and make a swap in one transaction. Also allows to specify funds destination instead of `msg.sender`

Parameters:

Name	Type	Description
`order`	Order	Order quote to fill
`signature`	bytes	Signature to confirm quote ownership
`amount`	uint256	Taker amount to fill
`takerTraits`	TakerTraits	Specifies threshold as maximum allowed takingAmount when takingAmount is zero, otherwise specifies minimum allowed makingAmount. Top-most bit specifies whether taker wants to skip maker's permit.
`target`	address	Address that will receive swap funds
`interaction`	bytes	A call data for Interactive. Taker may execute interaction after getting maker assets and before sending taker assets.
`permit`	bytes	Should contain abi-encoded calldata for `IERC20Permit.permit` call

Return Values:

Name	Type	Description
`makingAmount`	uint256	Actual amount transferred from maker to taker
`takingAmount`	uint256	Actual amount transferred from taker to maker

Name	Type	Description
`orderHash`	bytes32	Hash of the filled order

fillContractOrderExt

```
function fillContractOrderExt(
    Order calldata order,
    bytes calldata signature,
    uint256 amount,
    TakerTraits takerTraits,
    address target,
    bytes calldata interaction,
    bytes calldata permit,
    bytes calldata extension
) public returns(uint256,uint256,bytes32)
```

See {IOrderMixin-fillContractOrderExt}

Parameters:

Name	Type	Description
`order`	Order	
`signature`	bytes	
`amount`	uint256	
`takerTraits`	TakerTraits	
`target`	address	
`interaction`	bytes	
`permit`	bytes	
`extension`	bytes	

Return Values:

Name	Type	Description
`makingAmount`	uint256	
`takingAmount`	uint256	
`orderHash`	bytes32	

Events

OrderFilled

```
event OrderFilled(
    bytes32 orderHash,
    uint256 makingAmount
);
```

Emitted when order gets filled

Parameters:

Name	Type	Description
`orderHash`	bytes32	Hash of the order
`makingAmount`	uint256	Amount of the maker asset that was transferred from maker to taker

Previous
‹ GenericRouter

Next
UnoswapRouter ›