Pool

This documentation provides an overview of the IPool.sol. This contract defines the functions and events for interacting with a liquidity pool in Maverick AMM.

Table of Contents

- Contract Details
- Events
-
    - Swap
-
    - AddLiquidity
-
    - MigrateBinsUpStack
-
    - TransferLiquidity
-
    - RemoveLiquidity
-
    - BinMerged
-
    - BinMoved
-
    - ProtocolFeeCollected
-
    - SetProtocolFeeRatio
- *
- Structs
-
    - BinDelta
-
    - TwaState
-
    - BinState
-
    - AddLiquidityParams
-
    - RemoveLiquidityParams
-
    - State
- *
- Functions
-
    - fee()
-
    - tickSpacing()
-
    - tokenA()
-
    - tokenB()
-
    - factory()
-
    - binMap()
-
    - binPositions()
-
    - binBalanceA()
-
    - binBalanceB()
-
    - getTwa()
-
    - getCurrentTwa()
-
    - getState()
-
    - addLiquidity()

-
  - transferLiquidity()
-
  - removeLiquidity()
-
  - migrateBinUpStack()
-
  - swap()
-
  - getBin()
-
  - balanceOf()
-
  - tokenAScale()
-
  - tokenBScale()
- *
-

Contract Details

- Name :
- IPool
- Solidity Version :
- ^0.8.0
- SPDX License-Identifier : GPL-2.0-or-later
- Code:Github
-

Events

Swap

```

Copy
eventSwap(addresssender,addressrecipient,booltokenAIn,boolexactOutput,uint256amountIn,uint256amountOut,int32activeTick);

```

- sender
- : Theaddress
- that executed this swap
- recipient
- : Theaddress
- receiving this swap
- tokenAIn
- : Aboolean
- to determine if there is any input for Token A
- exactOutput
- : Aboolean
- to determine if there is any exact amount of tokens expected to receive
- amountIn
- : Theuint256
- amount of the input token
- amountOut
- : Theuint256
- amount of the output token
- activeTick
- : The active tickint32
- value for the pool
-

AddLiquidity

```

Copy eventAddLiquidity(addressindexedsender,uint256indexedtokenId, BinDelta[] binDeltas);

```

- sender

- : The indexed senderaddress
- that executedaddLiquidity()
- tokenId
- : Theuint256
- indexed ID of the receiving token
- binDeltas
- : An array ofBinDelta
- structures
-

MigrateBinsUpStack

```

Copy eventMigrateBinsUpStack(addressindexedsender,uint128binId,uint32maxRecursion);

```

- sender
- : The indexed senderaddress
- that executedMigrateBinsUpStack()
- binId
- : Theuint128
- bin ID that was migrated
- maxRecursion
- : Maximum recursion depth inuint32
-

TransferLiquidity

```

Copy eventTransferLiquidity(uint256fromTokenId,uint256toTokenId, RemoveLiquidityParams[] params);

```

- fromTokenId
- : Transfer liquidity from token IDuint256
- toTokenId
- : Transfer liquidity to token IDuint256
- params
- : Array ofRemoveLiquidityParams
- that specify the bins and amounts
-

RemoveLiquidity

```

Copy eventRemoveLiquidity(addressindexedsender,addressindexedrecipient,uint256indexedtokenId, BinDelta[] binDeltas);

```

- sender
- : Remove liquidity from senderaddress
- recipient
- : Remove liquidity to receiveraddress
- tokenId
- : Current indexeduint256
- tokenId to remove liquidity from
- binDeltas
- : Array ofBinDelta
- that specify the bins and amounts
-

BinMerged

```

Copy eventBinMerged(uint128indexedbinId,uint128reserveA,uint128reserveB,uint128mergeId);

```

- binId
- : The indexed bin IDuint128
- that was merged.
- reserveA
- : amount of A tokenuint128
- in bin.
- reserveB
- : amount of B tokenuint128
- in bin.
- mergeId
- : The current active binuint128
- .
-

BinMoved

```

Copy eventBinMoved(uint128indexedbinId,int128previousTick,int128newTick);

```

- binId
- : The indexed bin IDuint128
- that was moved.
- previousTick
- : Previous tick value inint128
- .
- newTick
- : New tick value inuint128
- .
-

ProtocolFeeCollected

```

Copy eventProtocolFeeCollected(uint256protocolFee,boolisTokenA);

```

- protocolFee
- : Amount of Protocol fee collected inuint256
- .
- isTokenA
- :boolean
- check if Token A was used for Protocol fee.
-

SetProtocolFeeRatio

```

Copy eventSetProtocolFeeRatio(uint256protocolFee);

```

- protocolFee
- : The new amount of Protocol fee set inuint256
- .
-

Structs

BinDelta

Return parameters for Add/Remove liquidity.

```

Copy structBinDelta{ uint128deltaA; uint128deltaB; uint256deltaLpBalance; uint128binId; uint8kind; int32lowerTick; boolisActive; }
```

```

- deltaA
- : The amount of A tokenuint128
- that has been added or removed
- deltaB
- : The amount of B tokenuint128
- that has been added or removed
- deltaLpBalance
- : The amount of LP balanceuint256
- that has increase(add)
- or decreased(remove)
- binId
- : The bin IDuint128
- of the bin that changed
- kind
- : One of the 4 Kinds(0=static, 1=right, 2=left, 3=both)
- inuint8
- lowerTick
- : The lower price tickint32
- of the bin in its current state
- isActive
- : Aboolean
- to indicate whether the bin is still active
-

TwaState

Time weighted average state.

```

Copy structTwaState{ int96twa; int96value; uint64lastTimestamp; }

```

- twa
- : The twaint96
- at the last update instant
- value
- : The new valueint96
- that was passed in at the last update
- lastTimestamp
- : The timestampuint64
- of the last update in seconds
-

BinState

The bin state parameters.

```

Copy structBinState{ uint128reserveA; uint128reserveB; uint128mergeBinBalance; uint128mergeId; uint128totalSupply; uint8kind; int32lowerTick; }

```

- reserveA
- : The amount of A tokenuint128
- in bin
- reserveB
- : The amount of B tokenuint128
- in bin
- mergeBinBalance
- : The LP token balanceuint128
- that this bin possesses after merge
- mergeId
- : ThebinId
- that this binuint128
- has merged in to
- totalSupply
```

- : The total amount of LP tokensuint128
- in this bin
- kind
- : one of the 4 Kinds(0=static, 1=right, 2=left, 3=both)
- inuint8
- lowerTick
- : The lower price tickint32
- of the bin in its current state
-

AddLiquidityParams

Parameters for each bin that will get new liquidity.

```

Copy structAddLiquidityParams{ uint8kind; int32pos; boolisDelta; uint128deltaA; uint128deltaB; }

```

- kind
- : one of the 4 Kinds(0=static, 1=right, 2=left, 3=both)
- inuint8
- pos
- : The bin position inint32
- isDelta
- : Aboolean
- that indicates whether the bin position is relative to the current bin or an absolute position
- deltaA
- : The amount of A tokenuint128
- to add
- deltaB
- : The amount of B tokenuint128
- to add
-

RemoveLiquidityParams

Parameters for each bin that will have liquidity removed.

```

Copy structRemoveLiquidityParams{ uint128binId; uint128amount; }

```

- binId
- : The index of the binuint128
- losing liquidity
- amount
- : The LP balance amountuint128
- to remove
-

State

The state of the pool.

```

Copy structState{ int32activeTick; uint8status; uint128binCounter; uint64protocolFeeRatio; }

```

- activeTick
- : The current bin positionint32
- that contains the active bins
- status
- : The status valuesuint8
- defined inPool.sol
- e.g. locked or unlocked;
- binCounter
- : The indexuint128

- of the last bin created
- protocolFeeRatio
- : The ratio of the swap fee that is kept for the protocol inuint64
-

Functions

fee()

Retrieves the fee for the pool in 18 decimal format.

```

Copy functionfee()externalviewreturns(uint256);

```

- Returns :
-
    - The feeuint256
-
    - for the pool as a uint256 value
- *
-

tickSpacing()

Retrieves the tick spacing of the pool. The tick spacing is used to calculate the bin width.

```

Copy functiontickSpacing()externalviewreturns(uint256);

```

- Returns :
-
    - The tick spacing as auint256
-
    - value
- *
-

tokenA()

Retrieves the address of token A associated with the pool.

```

Copy functiontokenA()externalviewreturns(IERC20);

```

- Returns :
-
    - Theaddress
-
    - of token A as an IERC20 interface
- *
-

tokenB()

Retrieves the address of token B associated with the pool.

```

Copy functiontokenB()externalviewreturns(IERC20);

```

- Returns :
-
    - Theaddress

-
    - of token B as an IERC20 interface
- *
-

factory()

Retrieves the address of the factory contract associated with the pool.

```

Copy functionfactory()externalviewreturns(IFactory);

```

- Returns :
-
    - Theaddress
-
    - of the factory contract as anIFactory
-
    - interface
- *
-

binMap()

Retrieves the bitmap of active bins at the given tick.

```

Copy functionbinMap(int32tick)externalviewreturns(uint256);

```

- Parameters :
-
    - tick : The tickint32
-
    - for which to retrieve the active bin map
- *
- Returns :
-
    - The bitmap of active bins as auint256
-
    - value
- *
-

binPositions()

Retrieves the bin ID for the given tick and bin kind

```

Copy functionbinPositions(int32tick,uint256kind)externalviewreturns(uint128);

```

- Parameters:
-
    - tick
-
    - : The tickint32
-
    - for which to retrieve the bin ID
-
    - kind
-
    - : The kinduint256
-
    - of the bin(0=static, 1=right, 2=left, 3=both)
- *

- Returns:
-
  - The bin ID as a uint128 value
- *
-

binBalanceA()

Retrieves the internal accounting of the sum of token A balances across bins.

```
Copy functionbinBalanceA()externalviewreturns(uint128);
```

- Returns: The sum of token A balances as auint128
- value
-

binBalanceB()

Retrieves the internal accounting of the sum of token B balances across bins.

```
Copy functionbinBalanceB()externalviewreturns(uint128);
```

- Returns:
-
  - The sum of token B balances as auint128
-
  - value
- *
-

getTwa()

Retrieves the time-weighted average (TWA) state values.

```
Copy functiongetTwa()externalviewreturns(TwaStatememory);
```

- Returns:
-
  - ATwaState
-
  - structure containing the TWA, value, last timestamp, and look back
- *
-

getCurrentTwa()

Retrieves the log base binWidth of the time-weighted average price.

```
Copy functiongetCurrentTwa()externalviewreturns(int256);
```

- Returns:
-
  - The log base binWidth of the TWA as anint256
-
  - value.
- *
-

getState()

Retrieves the state of the pool.

```
Copy functiongetState()externalviewreturns(Statememory);
```

- Returns:
-
  - AState
-
  - structure containing the active tick, status, bin counter, and protocol fee ratio
- *
-

addLiquidity()

Adds liquidity to a pool.

```
Copy functionaddLiquidity(uint256tokenId,AddLiquidityParams[]calldataparams,bytescalldatadata) external
returns(uint256tokenAAmount,uint256tokenBAmount,BinDelta[]memorybinDeltas);
```

- Parameters:
-
  - tokenId
-
  - : The NFT token IDuint256
-
  - that will hold the position.
-
  - params
-
  - : An array ofAddLiquidityParams
-
  - structures that specify the mode, position, and liquidity details.
-
  - data
-
  - : A callback function thataddLiquidity
-
  - will call to transfer tokens.
- *
- Returns:
-
  - tokenAAmount
-
  - : The amount of token A added as auint256
-
  - value.
-
  - tokenBAmount
-
  - : The amount of token B added as auint256
-
  - value.
-
  - binDeltas
-
  - : An array ofBinDelta
-
  - structures representing the changes in bin states.
- *
-

transferLiquidity()

Transfers liquidity from one NFT token ID to another using an array of bins.

```
Copy functiontransferLiquidity(uint256fromTokenId,uint256toTokenId,RemoveLiquidityParams[]calldataparams)external;
```

- Parameters:
-
  - fromTokenId
-
  - : The NFT token IDuint256
-
  - that holds the position being transferred.
-
  - toTokenId
-
  - : The NFT token IDuint256
-
  - that is receiving the liquidity.
-
  - params
-
  - : An array ofRemoveLiquidityParams
-
  - structures specifying the bins and amounts to transfer.
- *
-

removeLiquidity()

Removes liquidity from a pool.

```
Copy functionremoveLiquidity(addressrecipient,uint256tokenId,RemoveLiquidityParams[]calldataparams) external
returns(uint256tokenAOut,uint256tokenBOut,BinDelta[]memorybinDeltas);
```

- Parameters:
-
  - recipient
-
  - : Theaddress
-
  - that will receive the removed tokens.
-
  - tokenId
-
  - : The NFT token IDuint256
-
  - that holds the position being removed.
-
  - params
-
  - : An array ofRemoveLiquidityParams
-
  - structures specifying the bins and amounts to remove.
- *
- Returns:
-
  - tokenAOut
-
  - : The amount of token Auin256
-
  - received as a result of removing liquidity.
-
  - tokenBOut
-
  - : The amount of token Buin256

-
  - received as a result of removing liquidity.
-
  - binDeltas
-
  - : An array ofBinDelta
-
  - structures representing the changes in bin states.
- *
-

migrateBinUpStack()

Migrates bins up the linked list of merged bins so that its mergeId is the current active bin.

```

Copy functionmigrateBinUpStack(uint128binId,uint32maxRecursion)external;

```

- Parameters:
-
  - binId
-
  - : An array of the bin IDsuint128
-
  - to be migrated.
-
  - maxRecursion
-
  - : The maximum recursion depthuint32
-
  - of the migration. Set tozero
-
  - to recurse until the active bin is found.
- *
-

swap()

Swaps tokens.

```

Copy functionswap( addressrecipient, uint256amount, booltokenAIn, boolexactOutput, uint256sqrtPriceLimit, bytescalldatadata )externalreturns(uint256amountIn,uint256amountOut);

```

- Parameters:
-
  - recipient
-
  - : Theaddress
-
  - that will receive the output tokens.
-
  - amount
-
  - : The amount of tokensuint256
-
  - to swap.
-
  - tokenAIn
-
  - : Aboolean
-
  - indicating whether token A is the input.
-
  - exactOutput
-

  - : Aboolean
-
  - indicating whether the amount specified is the exact output amount(true)
-
  - .
-
  - sqrtPriceLimit
-
  - : The limiting square root priceuint256
-
  - of the swap. A value of0
-
  - indicates no limit. The limit is only engaged forexactOutput=false
-
  - . If the limit is reached, only part of the input amount will be swapped, and the callback will only require that amount of the swap to be paid.
-
  - data
-
  - : A callback functionbytes
-
  - that swap will call to transfer tokens.
- *
- Returns:
-
  - amountIn
-
  - : The amount of tokensuint256
-
  - swapped as input.
-
  - amountOut
-
  - : The amount of tokensuint256
-
  - received as output.
- *
-

getBin()

Retrieves the bin information for a given bin ID.

```
Copy functiongetBin(uint128binId)externalviewreturns(BinStatememorybin);
```

- Parameters:
-
  - binId
-
  - : The index of the binuint128
-
  - .
- *
- Returns:
-
  - ABinState
-
  - structure containing the details of the bin.
- *
-

balanceOf()

Retrieves the LP token balance for a given tokenId at a specific binId.

```
```

Copy functionbalanceOf(uint256tokenId,uint128binId)externalviewreturns(uint256lpToken);

```

- Parameters:
-
  - tokenId
-
  - : The NFT token IDuint256
-
  - .
-
  - binId
-
  - : The index of the binuint128
-
  - .
- *
- Returns:
-
  - The LP token balance as auint256
-
  - value.
- *
-

tokenAScale()

Retrieves the tokenA scale value.

```

Copy functiontokenAScale()externalviewreturns(uint256);

```

msb is a flag to indicate whether tokenA has more or less than 18 decimals. Scale is used in conjuction withMath.toScale/Math.fromScale functions to convert from token amounts to D18 scale internal pool accounting.

- Returns:
-
  - The tokenA scale value as auint256
-
  - .
- *
-

tokenBScale()

Retrieves the tokenB scale value.

```

Copy functiontokenBScale()externalviewreturns(uint256);

```

msb is a flag to indicate whether tokenA has more or less than 18 decimals. Scale is used in conjuction withMath.toScale/Math.fromScale functions to convert from token amounts to D18 scale internal pool accounting.

- Returns:
-
  - The tokenB scale value as auint256
-
  - .
- *
-

Previous Router Next Factory Last updated9 months ago On this page *Table of Contents * Contract Details * Events * Swap * AddLiquidity * MigrateBinsUpStack * TransferLiquidity * RemoveLiquidity * BinMerged * BinMoved * ProtocolFeeCollected * SetProtocolFeeRatio * Structs * BinDelta * TwaState * BinState * AddLiquidityParams * RemoveLiquidityParams * State * Functions * fee() * tickSpacing() * tokenA() * tokenB() * factory() * binMap() * binPositions() * binBalanceA() * binBalanceB() * getTwa() * getCurrentTwa() * getState() * addLiquidity() * transferLiquidity() * removeLiquidity() * migrateBinUpStack() * swap() *
```

getBin() * balanceOf() * tokenAScale() * tokenBScale()