SdkBase

SDK class encapsulating bridge functions.

Hierarchy

- SdkShared
- ↳SdkBase
-

Methods

bumpTransfer

▸bumpTransfer (params):Promise <TransactionRequest

Increases the relayer fee for a specific transfer on origin; anyone is allowed to bump for any transfer.

Example

```

Copy // call SdkBase.create(), instantiate a signer

constparams={ domainId:"6648936", transferId:"0xdd252f58a45dc78fee1ac12a628782bda6a98315b286aadf76e4d7322bf135ca", asset:"0x0000000000000000000000000000000000000000",// can be either native asset or transacting asset relayerFee:"10000", };

consttxRequest=sdkBase.bumpTransfer(params); signer.sendTransaction(txRequest);

```

Parameters

Name Type Description params Object SdkBumpTransferParams object. params.asset string The asset address you want to pay in (use "0x0000000000000000000000000000000000000000" for native). params.domainId string The origin domain ID of the transfer. params.relayerFee string The additional relayer fee to increase the transfer by, in the specified asset. params.transferId string The transfer ID. Returns

Promise <TransactionRequest

providers.TransactionRequest object.

calculateAmountReceived

▸calculateAmountReceived (originDomain ,destinationDomain ,originTokenAddress ,amount ,receiveLocal? ,checkFastLiquidity?):Promise <{amountReceived :BigNumberish ;destinationSlippage :BigNumberish ;originSlippage :BigNumberish ;routerFee :BigNumberish ;isFastPath :boolean }>

Calculates the estimated amount received on the destination domain for a bridge transaction.

Parameters

Name Type Default value Description originDomain string undefined The domain ID of the origin chain. destinationDomain string undefined The domain ID of the destination chain. originTokenAddress string undefined The address of the token to be bridged from origin. amount BigNumberish undefined The amount of the origin token to bridge, in the origin token's native decimal precision. receiveLocal boolean false (optional) Whether the desired destination token is the local asset ("nextAsset"). checkFastLiquidity boolean false (optional) Whether to check current router liquidity for fast path availability. Returns

Promise <{amountReceived :BigNumberish ;destinationSlippage :BigNumberish ;originSlippage :BigNumberish ;routerFee :BigNumberish ;isFastPath :boolean }>

Estimated amount received for local/adopted assets, if applicable, in their native decimal precisions.

estimateRelayerFee

▸estimateRelayerFee (params):Promise <BigNumber

Calculates an estimated relayer fee in the native asset of the origin domain to be used in xcall.

Example

```

Copy // call SdkBase.create(), instantiate a signer

constparams={ originDomain:"6648936", destinationDomain:"1869640809", };

consttxRequest=sdkBase.estimateRelayerFee(params); signer.sendTransaction(txRequest);

```

Parameters

Name Type Default value Description params Object undefined SdkEstimateRelayerFeeParams object. params.originDomain string undefined The origin domain ID of the transfer. params.destinationDomain string undefined The destination domain ID of the transfer. params.callDataGasAmount string undefined The gas amount needed for calldata. params.originNativeToken string "0x0000000000000000000000000000000000000000" (optional) The native token of the origin domain. params.priceIn string "native" (optional) "native" for native asset denomination or "usd" to get the estimate in USD value. params.destinationNativeToken string "0x0000000000000000000000000000000000000000" (optional) The native token of the destination domain. params.originNativeTokenPrice number (uses external estimate - increases response time) (optional) The USD price of the origin native token. params.destinationNativeTokenPrice number (uses external estimate - increases response time) (optional) The USD price of the destination native token. params.destinationGasPrice string (uses external estimate - increases response time) (optional) The gas price of the destination chain, in gwei units. Returns

Promise <BigNumber

The relayer fee in native asset of the origin domain or USD equivalent.

xcall

▸xcall (params):Promise <TransactionRequest

Prepares xcall inputs and encodes the calldata. Returns an ethers TransactionRequest object, ready to be sent to an RPC provider.

Example

```

Copy // call SdkBase.create(), instantiate a signer

constparams={ origin:"6648936" destination:"1869640809" to:"0x3cEe6c5c0fB713925BdA590829EA574b7b4f96b6" asset:"0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48" delegate:"0x3cEe6c5c0fB713925BdA590829EA574b7b4f96b6" amount:"1000000" slippage:"300" callData:"0x", relayerFee:"10000000000000" };

consttxRequest=sdkBase.xcall(params); signer.sendTransaction(txRequest);

```

Parameters

Name Type Description params Object SdkXCallParams object. params.amount undefined |string (optional) The amount of tokens (in specified asset) to send with the xcall. IfwrapNativeOnOrigin is true, this will be used as the amount of native token to deposit into the wrapper contract and withdraw as wrapped native token for sending (e.g. deposit ETH to the WETH contract in exchange for the WETH ERC20). params.asset undefined |string (optional) The target asset to send with the xcall. Can be set toaddress(0) if this is a 0-value transfer. IfwrapNativeOnOrigin is true, this should be the target wrapper contract (e.g. WETH) address. params.callData undefined |string (optional) Calldata to execute (can be empty: "0x"). params.delegate undefined |string (optional) Address allowed to cancel an xcall on destination. params.destination string The destination domain ID. params.origin string The origin domain ID. params.receiveLocal undefined |boolean (optional) Whether to receive the local asset ("nextAsset"). params.relayerFee undefined |string (optional) Fee paid to relayers, in native asset on origin. UsecalculateRelayerFee to estimate. params.relayerFeeInTransactingAsset undefined |string (optional) Fee paid to relayers, in transacting asset on origin. UsecalculateRelayerFee to estimate. params.slippage undefined |string (optional) Maximum acceptable slippage in BPS. For example, a value of 30 means 0.3% slippage. params.to string Address receiving funds or the target contract. params.wrapNativeOnOrigin undefined |boolean (optional) Whether we should wrap the native token before sending the xcall. This will use the Multisend utility contract to deposit ETH, approve Connext as a spender, and call xcall. If set true,asset should be the target wrapper contract (e.g. WETH) address. params.unwrapNativeOnDestination undefined |boolean (optional) Whether we should unwrap the wrapped native token when the transfer reaches its destination. By default, if sending a wrapped native token, the wrapped token is what gets delivered at the destination. Setting this totrue means we should overwritecallData to target the Unwrapper utility contract, which will unwrap the wrapped native token and deliver it to the target recipient (theto address). Returns

Promise <TransactionRequest

providers.TransactionRequest object.

create

▸Static create (_config):Promise <SdkBase

Create a singleton instance of the SdkBase class.

Parameters

Name Type Default value Description _config Object undefined SdkConfig object. _config.chains Recordundefined Chain config, at minimum with providers for each chain. _config.signerAddress string undefined Signer address for transactions. _config.logLevel "fatal" |"error" |"warn" |"info" |"debug" |"trace" |"silent" "info" (optional) Logging severity level. _config.network "testnet" |"mainnet" "mainnet" (optional) Blockchain environment to interact with. Returns

Promise <SdkBase

providers.TransactionRequest object.

Example

```

Copy import{ SdkBase }from"@connext/sdk";

constconfig={ signerAddress:"", network:"mainnet", chains:{ 6648936:{// the domain ID for Ethereum Mainnet providers:["https://rpc.ankr.com/eth"], }, 1869640809:{// the domain ID for Optimism providers:["https://mainnet.optimism.io"] }, 1886350457:{// the domain ID for Polygon providers:["https://polygon-rpc.com"] }, }, }

constsdkBase=awaitSdkBase.create(config);

```

See theDeployments page for all domain IDs and asset addresses.

updateSlippage

▸updateSlippage (params):Promise <TransactionRequest

Updates the slippage tolerance for a specific transfer on origin; only the origin sender is allowed to do so.

Example

```

Copy // call SdkBase.create(), instantiate a signer

constparams={ domainId:"6648936", transferId:"0xdd252f58a45dc78fee1ac12a628782bda6a98315b286aadf76e4d7322bf135ca", relayerFee:"1000", };

consttxRequest=sdkBase.updateSlippage(params); signer.sendTransaction(txRequest);

```

Parameters

Name Type Description params Object SdkUpdateSlippageParams object. params.domainId string The origin domain ID of the transfer. params.slippage string The new relayer fee to use for this transfer, in BPS. params.transferId string The transfer ID. Returns

Promise <TransactionRequest

providers.TransactionRequest object.