Getting all possible Connections

A connection is a pair of two tokens (on the same chain or on different chains) that can be exchanged via our platform. The complete API documentation for our/connections endpoint can be foundhere . After gathering information about available chains and tokens, the next step could be learning which transfers from one token to another are possible. We call those transfer possibilities "connections".

To get those connections we provide a/connections endpoint. It accepts filtering byfromChain ,toChain ,fromToken, toToken orchainTypes . Please note that the alternatives for chainTypes are SVM (Solana virtual machine) and EVM. By default, only EVM connections will be returned.

Which combination of filters you want to use, is up to you. There is only one limitation:

Since we support many different chains with many different tokens, the connections endpoint has to be filtered to ensure the response size doesn't get too big.

For this reason, eitherfromChain ortoChain has to be passed in the query. ```

Copy constgetConnections=async(fromChain,toChain,fromToken,toToken,chainTypes)=>{ constresult=awaitaxios.get('https://li.quest/v1/connections',{ params:{ fromChain, toChain, fromToken, toToken, chainTypes } }); returnresult.data; }

```

For example, if you want to know which tokens you can receive on Gnosis by sending out USDT on Polygon (https://li.quest/v1/connections?fromChain=POL&toChain=DAI&fromToken=USDT ) you will get a response that looks the following:

```

Copy { "connections":[{ "fromChainId":137, "toChainId":100, "fromTokens":[{ "address":"0xc2132d05d31c914a87c6611c10748aeb04b58e8f", "decimals":6, "symbol":"USDT", "chainId":137, "coinKey":"USDT", "name":"USDT", "logoURI":"https://static.debank.com/image/matic_token/logo_url/0xc2132d05d31c914a87c6611c10748aeb04b58e8f/66eadee7b7bb16b75e02b570ab8d5c01.png", "priceUSD":"1" }], "toTokens":[{ "address":"0x4ecaba5870353805a9f068101a40e0f32ed605c6", "decimals":6, "symbol":"USDT", "chainId":100, "coinKey":"USDT", "name":"USDT", "logoURI":"https://static.debank.com/image/xdai_token/logo_url/0x4ecaba5870353805a9f068101a40e0f32ed605c6/66eadee7b7bb16b75e02b570ab8d5c01.png", "priceUSD":"1" }, { "address":"0x6a023ccd1ff6f2045c3309768ead9e68f978f6e1", "decimals":18, "symbol":"ETH", "chainId":100, "coinKey":"ETH", "name":"ETH", "logoURI":"https://static.debank.com/image/xdai_token/logo_url/0x6a023ccd1ff6f2045c3309768ead9e68f978f6e1/61844453e63cf81301f845d7864236f6.png", "priceUSD":"3085.55" }, ...// more toTokens will be in this list] }, ...// more connections will be in this list] }

```

It is essential to know that there can be multiple connections for the samefromChainId andtoChainId pair. This is because we try to group the available tokens as much as possible to reduce redundancy, but not every token can be exchanged for every other token.

Scenario: Showing all possible connections based on a predefined destination token

Imagine the following scenario: You are a game that allows the players to buy custom skins using USDT on Polygon. To allow your players to swap their own assets into USDT, you integrate LI.FI.

In your user interface you now want to show your players a list of all tokens grouped by their chain that they can swap to USDT (on Polygon) in order to buy a skin:

```

Copy constgetAllPossibleConnections=async()=>{ constresult=awaitaxios.get('https://li.quest/v1/connections',{ params:{ toChain:'POL', toToken:'USDT', } }); returnresult.data.connections; }

constpossibleConnections=awaitgetAllPossibleConnections();

constfromTokensByChain={}; possibleConnections.forEach(connection=>{ if(!fromTokensByChain[connection.fromChainId]) { fromTokensByChain[connection.fromChainId]=[]; }

fromTokensByChain[connection.fromChainId].push(...connection.fromTokens); });

```

Now you should be able to simply render this list and provide an excellent overview of the available exchange options.

If you don't feel like implementing all of that yourself and you happen to work in a JavaScript/TypeScript environment, you should look at our SDK .

However, if you want to include our beautiful widget in your application, head to our Widget Documentation .
```