LI.Fuel

To onboard users to a new chain they need both tokens they can bridge via LI.FI and some gas to be able to submit their first transactions. LI.Fuel helps to do both in one transaction. Note : currently enabled only for ETH, POL, Gnosis and SOL.

LI.Fuel allows to receive a part of the bridged tokens as gas on the destination chain. This is possible by adding thefromAmountForGas to a quote or routes request:

- AddfromAmountForGas
- to the quote
- Sending out 0.01 ETH on Ethereum and use 0.001 of it for LI.Fuelhttps://etherscan.io/tx/0x4f7dd60440d43800ed0f81410720d3f206ff11bcc80c375478f2c54b8f51d136
- Status of gas delivery:https://li.quest/v1/gas/status?txHash=0x78fa5481857c85bee83c1c0c69231e72adbde86bdbf01914f8e4f19562a80287
- Delivered gas (0.287 MATIC on Polygon):https://polygonscan.com/tx/0x66011514291d69f4c05be15de207f7ba887f7beaceb0dec504cda28a28f6c96b
- Status of token delivery:https://li.quest/v1/status?fromChain=1&toChain=137&txHash=0x4f7dd60440d43800ed0f81410720d3f206ff11bcc80c375478f2c54b8f51d136
- Delivered token (0.00899 WETH on Polygon):https://polygonscan.com/tx/0x0da187e326e3914ac3390270faaf78352eed557c8b336f8a4d1b98d2373966ea
-

Example Flow

A user is on Ethereum and wants to transfer 5 USDC to Polygon.

1. Check if a user has gas on the destination chain (Polygon):
2.

```

Copy constbalance=awaitgetProvider(destinationChain).getBalance(userAddress)

```

1. If a user has no gas on the destination chain we need to find out how much gas would be good to have on Polygon. The/gas/suggestion
2. endpoint provides that information. Optionally, the token a user is currently sending can be added to the request to calculate how much of that token should be swapped into gas:
3.

```

Copy https://li.quest/v1/gas/suggestion/137?fromChain=1&fromToken=0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48

{ // The service is available, has enough liquidity and is enabled for this chain "available":true,

// The recommended amount is high enough to execut 4-10 approvals + swaps on the destionation chain. // The amount is calculated based on the current market conditions (gasPrice) on the chain, to ensure that // users don't get stuck, but also don't send too much gas. "recommended": { "token":{...}, "amount":"275731340000000000", "amountUsd":"0.24" },

// LI.Fuel allows a maximum of 5 to be swapped for MATIC "limit": { "token":{...}, "amount":"9803921568627450980", "amountUsd":"5.00" },

// LI.Fuel will take 0.01 from the transferred amount as fee for the service (mainly gas costs on Polygon) "serviceFee": { "token":{...}, "amount":"2447142884471250", "amountUsd":"0.01" },

// USDC token we passed, 0.24 for gas in USDC is 240000 "fromToken":{...}, "fromAmount":"240000" }

```

1. Knowing these recommendations, any value between the USD amount of the service fee and the limit can be chosen. We would suggest using the recommended amount as default. As we passed the token (USDC) the user wants to pay with, we can use the receivedfromAmount
2. parameter which is the amount of USDC the user needs to pay on Ethereum to get the recommended amount of gas on the destination chain. In this example, it is240000
3. . To include this in a route add the value asfromAmountForGas: 240000
4. .
5.

In a quote request:

```

Copy https://li.quest/v1/quote?
fromChain=1&fromAmount=5000000&fromToken=0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48&toChain=137&toToken=0x2791Bca1f2de4661ED88A30C99A7a9449Aa84174&fromAddress=0xdemo&fromAmountForGas=240000

```

In a routes request:

```

Copy curl--location'https://li.quest/v1/advanced/routes'\ --header'Content-Type: application/json'\ --data'{ "fromChainId": 1, "fromTokenAddress": "0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48", "toChainId": 137, "toTokenAddress": "0x2791Bca1f2de4661ED88A30C99A7a9449Aa84174", "fromAddress": "0x552008c0f6870c2f77e5cC1d2eb9bdff03e30Ea0", "toAddress": "0x552008c0f6870c2f77e5cC1d2eb9bdff03e30Ea0",

"fromAmount": "10000000", "fromAmountForGas": "240000",

"options": { "integrator": "lifuel-demo" } }'

```

1. Submit the quote/route as any other cross-chain route. In the transaction the funds are split automatically to send the requested amount to LI.Fuel, and the rest to the selected bridge. Monitor the transfer using the status endpoint.
2. (optional) LI.Fuel is designed to pay out gas to the user quickly, so he already has the gas when the other bridged assets arrive. It is not required to check the status of the gas delivery but it can be done by passing the transaction hash from the source chain to this endpoint:
3.

```

Copy https://li.quest/v1/gas/status?txHash=0x78fa5481857c85bee83c1c0c69231e72adbde86bdbf01914f8e4f19562a80287

```

Last updated1 month ago On this page Was this helpful?Export as PDF