Handling Chain Switches

We help you handling network changes. Some routes require a network switch in the end-users wallet during execution. This need arises when the route first bridges a token to another chain and subsequently wants to swap it into the final output token using an exchange. Since that last swap requires sending a transaction to an external contract, a renewedSigner for the new chain is necessary. The LI.FI SDK can no longer use the initial Signer passed to the executeRoute or resumeRoute functions since they point to another chain.

You can enable or disable routes that require chain switches with theallowSwitchChain property in theRouteRequest . If you allow such routes, you need to handle those chain switches somehow. If you want to learn more about route requests, please read#build-a-routesrequest-object .

TheExecutionSettings you can pass to theexecuteRoute function allows you to pass aswitchChainHook . This hook is called every time such a chain switch is needed. It gives you the chain id of the required chain and expects aSigner for the new chain.

```

Copy interfaceExecutionSettings{ // ... switchChainHook?:SwitchChainHook }

typeSwitchChainHook=( requiredChainId:number )=>Promise

```

The execution fails if a chain switch is needed and you are not providing a new signer. In that case, you can resume the route withresumeRoute and the new Signer. The same applies if you haven't provided aswitchChainHook in the first place.

Example code snippet: MetaMask

The code snippet below shows how to handle a chain switch with the MetaMask browser extension.

```

Copy // ... prepare a transfer ...

// define the switchChainHook constswitchChainHook=(requiredChainId:number)=>{ // this is where MetaMask lives constethereum=(windowasany).ethereum

// check if MetaMask is available if(typeofethereum==='undefined')return

// use the MetaMask RPC API to switch chains automatically awaitethereum.request({ method:'wallet_switchEthereumChain', params:[{ chainId:requiredChainId }], })

// build a new provider for the new chain constnewProvider=newethers.providers.Web3Provider(window.ethereum)

// return the associated Signer returnnewProvider.getSigner() }

// execute the route constroute=awaitlifi.executeRoute(signer,chosenRoute,{ switchChainHook })

``` Last updated2 months ago On this page Was this helpful?Export as PDF