create-config)

Configure SDK

Get started and set up LI.FI SDK in just a few lines of code.

Create config

To get started, you need to create an initial configuration for the LI.FI SDK. This configuration contains the shared settings and data required for the proper functioning of other SDK features that developers will use. Additionally, the configuration can be updated later as needed.

```

Copy import{ createConfig }from'@lifi/sdk'

createConfig({ integrator:'Your dApp/company name', })

```

Parameters

integrator (string, required)

LI.FI SDK requires anintegrator option to identify partners and allows them to monitor their activity on the partner dashboard, such as the transaction volume, enabling better management and support.

Usually, the integrator option is your dApp or company name.

This string must consist only of letters, numbers, hyphens, underscores, and dots and be a maximum of 23 characters long.

apiKey (string, optional)

Unique API key for accessing LI.FI API services. Necessary for higher rate limits. Read moreRate Limits & API Key

apiUrl (string, optional)

Default :https://li.quest/v1

The base URL for the LI.FI API. This is the endpoint through which all API requests are routed. It can be changed to the staging environment to test new features, for example.

If you are interested in a dedicated API endpoint for your service, reach out via ourDiscord or file an issue in our repositoryhere .

userId (string, optional)

A unique identifier for the user of your application. This can be used to track user-specific data and interactions within the LI.FI.

routeOptions (RouteOptions, optional)

Custom options for routing, applied when usinggetQuote ,getRoutes , andgetContractCallsQuote endpoints. These options can be configured once during SDK initialization or passed each time those functions are called. Read moreRequest Routes/Quotes

rpcUrls (RPCUrls, optional)

A mapping of chain IDs to arrays of RPC URLs. These URLs might be used for transaction execution and data retrieval.

```

Copy import{ createConfig,ChainId }from'@lifi/sdk'

createConfig({ integrator:'Your dApp/company name', rpcUrls:{

}, })

```

In a production app, it is recommended to pass through your authenticated RPC provider URL (Alchemy, Infura, Ankr, etc).

If no RPC URLs are provided, LI.FI SDK will default to public RPC providers.

chains (ExtendedChain[], optional)

Default : fetched from LI.FI API during initialization.

An array of chains that the SDK will support. Each chain must be configured with necessary details like chain ID, name, RPCs, etc. This information is used during the quote and route execution.

WhencreateConfig is called, the SDK preloads available chains from the LI.FI API. Developers can disable chain preloading to specify the chains themselves.

preloadChains (boolean, optional)

Default :true

A flag indicating whether to preload chain configurations. By default, the SDK will load chain details during initialization.

disableVersionCheck

(boolean, optional)

Default :false

A flag to disable version checking of the SDK. By default, the SDK checks its version on initialization and logs a message in the console if a new version is available, prompting the user to update the SDK.

providers (SDKProvider[], optional)

An array of provider configurations is used by the SDK. Providers are optional and only necessary if you plan to execute quotes or routes through the SDK. Read moreConfigure SDK Providers .

Setting up providers is not required if you are using the SDK solely to access the LI.FI API without quote/route SDK execution functionality and plan to handle the execution independently.

Read moreConfigure SDK Providers .

Update SDK configuration

LI.FI SDK provides various methods to manage and manipulate the SDK settings dynamically. To update the configuration, you need to import the global configuration object and use its methods.

```

Copy import{ config }from'@lifi/sdk';

```

After creating your configuration with thecreateConfig function,config acts as a global configuration object.

Configuration Methods

get()

Returns the current SDK configuration.

set(options: SDKConfig)

Sets the SDK configuration with the provided options.

setProviders(providers: SDKProvider[])

Sets the providers in the SDK configuration. If a provider already exists, it will be updated with the new information.

getChains()

Returns a promise that resolves to the list of configured chains. If the configuration is still loading, the promise will wait until the loading is complete.

setChains(chains: ExtendedChain[])

Sets the chains in the SDK configuration and updates chains RPC URLs. This method also clears the loading state.

getChainById(chainId: ChainId)

Returns a promise that resolves to the chain configuration for the specified chain ID. If the configuration is still loading, the promise will wait until the loading is complete.

getRPCUrls()

Returns a promise that resolves to the list of RPC URLs for the configured chains. If the configuration is still loading, the promise will wait until the loading is complete.

setRPCUrls(rpcUrls: RPCUrls)

Sets the RPC URLs for the chains in the SDK configuration. If some RPC URLs already exist for a chain, the new URLs will be appended to the existing ones.

Configure SDK Providers Last updated4 months ago