Getting Started with Lava SDK

Video Demonstration (~11m)

Written Guide (~5m)

0. Sign up for an Account on the Gateway

While not a strict prerequisite for using the SDK - using the Lava Gateway gives an easy and free way to get aprivateKey and/orbadge , which LavaSDK requires to initialize. We recommend deciding whether you're going to use Lava on thebackend orfrontend and starting there. LavaSDK is peer-to-peer (p2p) and executes relays in a decentralized manner.

1. Set up a new Node.JS project using Node Package Manager

To get started, we'll opt for a simple node application.

mkdir sdk-project/ cd sdk-project npm init -y

2. Install the SDK using yarn or NPM

npm i @lavanet/lava-sdk or

yarn

add @lavanet/lava-sdk

3. Create a newindex.js

file and import the Lava SDK

import

{

LavaSDK

}

from

"@lavanet/lava-sdk" ; OR

const

{

LavaSDK

}

=

require ("@lavanet/lava-sdk")

4. Initialize an instance of the SDK

info When developing on thebackend , it is currently best practice to hide the privatekey in an environmental variable instead of putting it in plain text in your code. For now, we'll useprivKey as a stand-in!

When developing on thefrontend , you don't need to use privatekeys at all. Simply input a badge! * Badges * Private Key

//Our Main Program Function async

function

useSDK ( )

{ // For CosmosHub Testnet, Juno Testnet, & Polygon Testnet Access const lavaNetwork =

await

LavaSDK . create ({ badge :

{ badgeServerAddress :

"https://badges.lavanet.xyz"

//or your own URL projectId :

"" } , chainIds :

['COS5T' , 'JUNT1' , 'POLYGON1'] }) ; //Our Main Program Function async

function

useSDK ( )

{ // For CosmosHub Testnet, Juno Testnet, & Polygon Testnet Access const lavaNetwork =

await

LavaSDK . create ({ privateKey : privKey , chainIds :

['COS5T' , 'JUNT1' , 'POLYGON1'] }) ; There is no limit to the amount of chains you can handle simultaneously! In addition to those shown in the example above, there are a number ofoptional parameters that you can view.

A full list of supported chains, their respective IDs, and supported interfaces can be found usinglavad

lavad q spec show-all-chains For a short list of just chainIDs run it as follows:

lavad q spec show-all-chains |

grep chainID

5. Make your queries or requests

We'll do so by sending relays within ouruseSDK() function!

//Example Juno Query - Grab an arbitrary block from Juno! const junoBlockResponse =

await lavaNetwork . sendRelay ({ method :

"block" , params :

["82500"] , chainId :

"JUNT1" , apiInterface :

"tendermintrpc" }) ;

console . log ("Juno Block: " , junoBlockResponse) ;

//Example Cosmos Query - Get the latest block from CosmosHub! const cosmosBlockResponse =

await lavaNetwork . sendRelay ({ method :

"GET" , url :

"/cosmos/base/tendermint/v1beta1/blocks/latest" , chainId :

"COS5T" , apiInterface :

"rest" }) ;

console . log ("Cosmos Block: " , cosmosBlockResponse)

//Example Polygon Query - Get the most recent block from Polygon! const polygonBlockResponse =

await lavaNetwork . sendRelay ({ method :

"eth_blockNumber" , params :

[ ] , chainId :

"POLYGON1" , apiInterface :

"jsonrpc" }) ;

console . log ("Polygon Block: " , polygonBlockResponse)

6. Now let’s implement the program logic

We want to calluseSDK() to run asynchronously.

(async

( )

=>

{ await

useSDK ( ) ; }) ( ) ;

7. Let’s run it

node index . js You should get 3 responses like so:

That’s it! You’ve successfully used LavaSDK.

For more information look around the rest of our documentation!

Having trouble? Head to ourDiscord! Edit this page Previous SDK Next Backend Use