

[npm](#) :

npm

install @lavanet/lava-sdk import

```
{
```

```
LavaSDK
```

```
}
```

from

```
"@lavanet/lava-sdk" ;
```

```
// create and initialize an SDK instance that handles several chains async
```

```
function
```

```
createLavaHandler ( )
```

```
{
```

```
const lavaHandler =
```

```
await
```

```
LavaSDK . create ( { badge :
```

```
{ badgeServerAddress :
```

```
"https://badges.lavanet.xyz" , projectId :
```

```
" // "
```

```
// get your projectId from gateway.lavanet.xyz } , chainIds :
```

```
[ "ETH1" ,
```

```
"NEAR" ,
```

```
"COS5" ,
```

```
"STRK" ,
```

```
"AXELAR" ] , geolocation :
```

```
"1" ,
```

```
//Optional, 1 for US, 2 for Europe } ) ;
```

```
return lavaHandler } ;
```

```
// send eth_blockNumber over JSONRPC to Ethereum Mainnet async
```

```
function
```

```
getEthereumBlockNum ( lavaHandler )
```

```
{
```

```
const ethRelay =
```

```
await lavaHandler . sendRelay ( { chainId :
```

```
"ETH1" , method :
```

```
"eth_blockNumber" , params :
```

```
[ ] , rpcInterface :
```

```
"jsonrpc" , } ) ;
```

```

return

parseInt ( ethRelay . result ,

16 ) } } ;

//send block call to NEAR providers for finalized blocks async

function

getNearBlockHeight ( lavaHandler )

{

const nearRelay =

await lavaHandler . sendRelay ( { chainId :

"NEAR" , method :

"block" , params :

{

"finality" :

"final"

} , rpcInterface :

"jsonrpc" , } ) ;

return nearRelay . result . header . height ; } } ;

//send abci_info method call over TendermintRPC to CosmosHub Mainnet async

function

getCosmosBlockHeight ( lavaHandler )

{

const cosmosHubRelay =

await lavaHandler . sendRelay ( { chainId :

"COS5" , method :

"abci_info" , params :

[ ] , rpcInterface :

"tendermintrpc" , } ) ;

return cosmosHubRelay . result . response . last_block_height ; } } ;

//send starknet_blockNumber over Mainnet async

function

getStarknetBlockNumber ( lavaHandler )

{

const starknetRelay =

await lavaHandler . sendRelay ( { chainId :

"STRK" , method :

"starknet_blockNumber" , params :

[ ] , rpcInterface :

```

```

"jsonrpc" , } ) ;

return starknetRelay . result ; } ;

//send GET "/blocks/latest" to Axelar RPC REST API async

function

getAxelarBlockHeight ( lavaHandler )

{

const axelarRelay =

await lavaHandler . sendRelay ( { chainId :

"AXELAR" , connectionType :

"GET" , url :

"/blocks/latest" } )

return axelarRelay . block . header . height } ;

// Create our SDK instance and pass it to each Function // Console.log() the results! async

function

useMultiChainWithBadges ( )

{

try

{ const lavaRelayHandler =

await

createLavaHandler ( )

console . log ( "RESULTS" ) ; console . log ( "=====") ; console . log ( "Axelar Block Number:" ,

await

getAxelarBlockHeight ( lavaRelayHandler ) ) ; console . log ( "Ethereum Block Number:" ,

await

getEthereumBlockNum ( lavaRelayHandler ) ) ; console . log ( "NEAR Block Height:" ,

await

getNearBlockHeight ( lavaRelayHandler ) ) ; console . log ( "CosmosHub Block Height:" ,

await

getCosmosBlockHeight ( lavaRelayHandler ) ) ; console . log ( "Starknet Block Number:" ,

await

getStarknetBlockNumber ( lavaRelayHandler ) ) ; console . log ( "\n" ) ; }

catch

( error )

{ console . error ( 'An error occurred:' , error ) ; } ;

} ;

// Make your calls ( async

( )

```

=>

```
{ await
```

```
useMultiChainWithBadges ( ) ; } ) ( ) Edit this page Previous Examples Next Lava SDK Beta Gallery
```