Client Library API Reference

Integrate Chainlink CCIP into your project

npmyarnfoundryIf you use[NPM](#) , install the[@chainlink/contracts-ccip NPM package](#) and set it to the v1.4.0 release:

npminstall@chainlink/[](#)If you use[Yarn](#) , install the[@chainlink/contracts-ccip NPM package](#) and set it to the v1.4.0 release:

yarnadd@chainlink/[](#)If you use[Foundry](#) , install the v1.4.0 release:

forgeinstallsmartcontractkit/ccip@b06a3c2eecb9892ec6f76a015624413fffa1a122

CCIP senders and receivers use theCCIP Client Libraryto build CCIP messages.

import{Client}from"@chainlink/contracts-ccip/src/v0.8/ccip/libraries/Client.sol";

Types and Constants

EVMTokenAmount

Use this solidity struct to specify the token address and amount.

structEVMTokenAmount{addresstoken;uint256amount;} NameTypeDescriptiontokenaddresstoken address on the local chain.amountuint256Amount of tokens.

Any2EVMMessage

CCIP receivers use this solidity struct to parse the received CCIP message.

structAny2EVMMessage{bytes32messageId;uint64sourceChainSelector;bytessender;bytesdata;structClient.EVMTokenAmount[]destTokenAmounts;} NameTypeDescriptionmessageIdbytes32CCIP messageId, generated on the source chain.sourceChainSelectoruint64Source chain selector.senderbytesSender address.abi.decode(sender, (address))if the source chain is an EVM chain.databytesPayload sent within the CCIP message.destTokenAmountsClient.EVMTokenAmount[]Tokens and their amounts in their destination chain representation.

EVM2AnyMessage

CCIP senders use this solidity struct to build the CCIP message.

structEVM2AnyMessage{bytesreceiver;bytesdata;structClient.EVMTokenAmount[]tokenAmounts;addressfeeToken;bytesextraArgs;} NameTypeDescriptionreceiverbytesReceiver address. Useabi.encode(sender)to encode the address tobytes.databytesPayload sent within the CCIP message.tokenAmountsClient.EVMTokenAmount[]Tokens and their amounts in the source chain representation.feeTokenaddressAddress of feeToken. Setaddress(0)to pay in native gas tokens such as ETH on Ethereum or MATIC on Polygon.extraArgsbytesUsers fill in the[EVMExtraArgsV1 struct](#) then encode it to bytes using the[argsToBytes function](#)

EVM_EXTRA_ARGS_V1_TAG

bytes4EVM_EXTRA_ARGS_V1_TAG

EVMExtraArgsV1

structEVMExtraArgsV1{uint256gasLimit;} NameTypeDescriptiongasLimituint256specifies the maximum amount of gas CCIP can consume to executeccipReceive()on the contract located on the destination blockchain. Read[Setting gasLimit](#) for more details.

Functions

_argsToBytes

function_argsToBytes(structClient.EVMExtraArgsV1 extraArgs)internalpurereturns(bytesbts) It is used to convert the arguments to bytes.

Parameters

NameTypeDescriptionextraArgsClient.EVMExtraArgsV1Extra arguments.

Return Values

NameTypeDescriptionbtsbytesEncoded extra arguments inbytes.