JavaScript code API Reference

JavaScript source code for a Functions request should comply with certain restrictions:

- Allowed Modules: VanillaDeno and moduleimports .
- Return Type: Must return a JavaScriptBufferobject representing the response bytes sent back to the invoking contract.
- Time Limit: Scripts must execute within a 10-second timeframe; otherwise, they will be terminated, and an error will be returned to the requesting contract.

HTTP requests

For making HTTP requests, use theFunctions.makeHttpRequestfunction.

Syntax

constresponse=awaitFunctions.makeHttpRequest({url:"http://example.com",method:"GET",// Optional// Other optional parameters})

Parameters

ParameterOptionalityDescriptionDefault ValueurlRequiredThe target URL.N/AmethodOptionalHTTP method to be used.'GET'headersOptionalHTTP headers for the request.N/AparamsOptionalURL query parameters.N/AdataOptionalBody content for the request.N/AtimeoutOptionalMaximum request duration in milliseconds.3000 msresponseTypeOptionalExpected response type.'json'

Return Object

Response TypeFieldsDescriptionSuccessdataResponse data sent by the server.statusNumeric HTTP status code.statusTextTextual representation of HTTP status.headersHTTP headers sent by the server in the response.ErrorerrorIndicates an error occurred (true).messageOptional error message.codeOptional error code.responseOptional server response.

Data encoding functions

The Functions library includes several encoding functions, which are useful for preparing data for blockchain contracts.

FunctionInput TypeOutput TypeDescriptionFunctions.encodeUint256Positive Integer32-byteBufferConverts a positive integer to a 32-byteBufferfor auint256in Solidity.Functions.encodeInt256Integer32-byteBufferConverts an integer to a 32-byteBufferfor anint256in Solidity.Functions.encodeStringStringBufferConverts a string to aBufferfor astringtype in Solidity. Note: Using these encoding functions is optional. The source code must return aUint8Array which represents thebytesthat are returned onchain.

constmyArr=newUint8Array(ARRAY_LENGTH)