

JavaScript code API Reference

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

- Allowed Modules: Vanilla [Deno](#) and module [imports](#) .
- 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 the `Functions.makeHttpRequest` function.

[Syntax](#)

```
const response = await Functions.makeHttpRequest({url: "http://example.com", method: "GET", // Optional // Other optional parameters})
```

[Parameters](#)

Parameter	Optionality	Description	Default Value	url	Required	The target URL.
method	Optional	HTTP method to be used.	'GET'	headers	Optional	HTTP headers for the request.
params	Optional	URL query parameters.	N/A	data	Optional	Body content for the request.
timeout	Optional	Maximum request duration in milliseconds.	3000	responseType	Optional	Expected response type. 'json'

[Return Object](#)

Response Type	Fields	Description
Success	data	Response data sent by the server.
status	Numeric	HTTP status code.
statusText	Textual	representation of HTTP status.
headers	HTTP	headers sent by the server in the response.
Error	error	Indicates an error occurred (true).
message	Optional	error message.
code	Optional	error code.
response	Optional	server response.

[Data encoding functions](#)

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

Function	Input Type	Output Type	Description
<code>Functions.encodeUint256</code>	Positive Integer	32-byte Buffer	Converts a positive integer to a 32-byte Buffer for a uint256 in Solidity.
<code>Functions.encodeInt256</code>	Integer	32-byte Buffer	Converts an integer to a 32-byte Buffer for an int256 in Solidity.
<code>Functions.encodeString</code>	String	String Buffer	Converts a string to a Buffer for a string type in Solidity.

Note: Using these encoding functions is optional. The source code must return a [Uint8Array](#) which represents the bytes that are returned onchain.

```
const myArr = new Uint8Array(ARRAY_LENGTH)
```