JSON-RPC methods

Here you can find the JSON-RPC API endpoints. Youcan call these APIs using a variety of tools.

Error codes

You may encounter various errors when interacting with a network:

- JSON-RPC errors
- operations on the blockchain network. It means the server has successfully received the JSON-RPC
- request but encountered an issue processing it. Causes might include invalid parameters, a method not
- found, or execution errors related to the requested operation.
- HTTP errors
- : These happen at the transport layer during data transmission to the blockchain
- network. They could stem from Infura-related issues like rate limits, API key problems, or
- service availability issues.
- Smart contract errors
- : These arise during attempts to execute transactions in the EVM
- involving smart contracts.

JSON-RPC errors

Error codes returned by Infura's RPC service network APIs can vary slightly between implementations, but they generally follow theJSON-RPC 2.0 specification and Ethereum-specific conventions.

Below is a table listing the error codes, their messages, and meanings. The "Standard" category includes common JSON-RPC errors, while the "Non-standard" category encompasses server errors defined by the implementation.

Code Message Meaning Category -32700 Parse error The JSON request is invalid, this can be due to syntax errors. Standard -32600 Invalid request The JSON request is possibly malformed. Standard -32601 Method not found The method does not exist, often due to a typo in the method name or the method not being supported. Standard -32602 Invalid argument Invalid method parameters. For example,"error":{"code":-32602,"message":"invalid argument 0: json: cannot unmarshal hex string without 0x prefix into Go value of type common.Hash"} indicates the0x prefix is missing from the hexadecimal address. Standard -32603 Internal error An internal JSON-RPC error, often caused by a bad or invalid payload. Standard -32000 Invalid input Missing or invalid parameters, possibly due to server issues or a block not being processed yet. Non-standard -32001 Resource not found The requested resource cannot be found, possibly when calling an unsupported method. Non-standard -32002 Resource unavailable The requested resource is not available. Non-standard -32003 Transaction rejected The transaction could not be created. Non-standard -32004 Method not supported The requested method is not implemented. Non-standard -32005 Limit exceeded The request exceeds your request limit. Non-standard -32006 JSON-RPC version not supported The version of the JSON-RPC protocol is not supported. Non-standard Example error response:

{ "id": 1337 "jsonrpc": "2.0", "error": { "code": -32003, "message": "Transaction rejected" } }

HTTP errors

Infura-specific error codes or messages could include errors for rate limits, API key issues, or service availability problems.

Code Message Meaning 400 Bad request Incorrect HTTP Request type or invalid characters, ensure that your request body and format is correct. 401 Unauthorized This can happen when one or multiple security requirements are not met. Example responses:project id required in the url ,invalid project id ,invalid project id or project secret ,invalid JWT . 403 Forbidden The request was intentionally refused due to specific settings mismatch, check your key settings. Example response:"error": {"code":-32002,"message":"rejected due to project ID settings"} . 429 Too Many Requests The daily request total or request per second are higher than your plan allows. Refer to theAvoid rate limiting topic for more information. Example responses:"error": {"code": -32005, "message": "daily request count exceeded, request rate limited"} ,"error": {"code": -32005, "message": "project ID request rate exceeded"} . 500 Internal Server Error Error while processing the request on the server side. 502 Bad Gateway Indicates a communication error which can have various causes, from networking issues to invalid response received from the server. 503 Service Unavailable Indicates that the server is not ready to handle the request. 504 Gateway Timeout The request ended with a timeout, it can indicate a networking issue or a delayed or missing response from the server.

Smart contract errors

When interacting with smart contracts, you might also encounter errors related to the execution of transactions in the EVM:

- Out of gas
- : The transaction does not have enough gas to complete.

- Revert
- : The transaction was reverted by the EVM, often due to a condition in the smart contract code.
- Bad instruction
- : The transaction tried to execute an invalid operation.
- Bad jump destination
- : A jump was made to an invalid location in the smart contract code.

Value encoding

Specific types of values passed to and returned from Ethereum RPC methods require special encoding:

Quantity

AQuantity (integer, number) must:

- Be hex-encoded.
- Be "0x"-prefixed.
- Be expressed using the fewest possible hex digits per byte.
- Express zero as "0x0".

ExamplesQuantity values:

Value Validity Reason 0x invalid empty not a valid quantity 0x0 valid interpreted as a quantity of zero 0x00 invalid leading zeroes not allowed 0x41 valid interpreted as a quantity of 65 0x400 valid interpreted as a quantity of 1024 0x0400 invalid leading zeroes not allowed ff invalid values must be prefixed

Block identifier

The RPC methods below take a default block identifier as a parameter.

- eth_getBalance
- eth_getStorageAt
- eth_getTransactionCount
- eth_getCode
- eth_call
- eth_getProof

Since there is no way to clearly distinguish between aData parameter and aQuantity parameter,EIP-1898 provides a format to specify a block either using the block hash or block number. The block identifier is a JSONobject with the following fields:

Property Type Description [blockNumber] {Quantity } The block in the canonical chain with this number OR[blockHash] {Data } The block uniquely identified by this hash. TheblockNumber andblockHash properties are mutually exclusive; exactly one of them must be set. requireCanonical {boolean }\ (optional) Whether or not to throw an error if the block is not in the canonical chain as described below. Only allowed in conjunction with theblockHash tag. Defaults tofalse .

Data

AData value (for example, byte arrays, account addresses, hashes, and bytecode arrays) must:

- Be hex-encoded.
- Be "0x"-prefixed.
- Be expressed using two hex digits per byte.

ExamplesData values:

Value Valid Reason 0x valid interpreted as empty data 0x0 invalid each byte must be represented using two hex digits 0x00 valid interpreted as a single zero byte 0x41 true interpreted as a data value of 65 0x004200 true interpreted as a data value of 16896 0xf0f0f false bytes require two hex digits 004200 false values must be prefixed

Last updatedonNov 5, 2024 Previous Supported networks Next Debug methods