# **Robin8 BlockChain BRIDGE API**

This API provides access to information of Robin8 Smart Contract based on Qtum BlockChain.

The API uses the json rpc 2.0 protocol (see [Specification](http://www.jsonrpc.org/specification) ).

The following is a description of the API-methods:

|  |  |
| --- | --- |
| **Method** | **lastblockid**  No auth |
| **Params** |  |
| **Sample response** | jsonrpc: "2.0"  result: "46688951f843"  id: null |
| **Description:** | This method returns id of current block in qtum blockchain (last block).  **!!! This parameter (lastblockid) is required to create api-requests that require authorization.**  blockid is a last 7 chars of last block hash concatenated with last block number in hex-format.  Example: <http://176.31.125.26/bridge/?method=lastblockid> |

|  |  |
| --- | --- |
| **Method** | **balance**  No auth |
| **Params** |  |
| **Sample response** | jsonrpc: "2.0"  result:  QiQN8kFhoCEX9aGsG3jxeQbiwXMw4JGhvz: 2.77671271  QQNJ1yL2h1YucDdfXkcoWf41WrbiXZGJma : 0.9735195699999999  QhQVxpQ28FncbxZCkA9U3xwp6AZF9jWGZg : 0.00718098  total : 3.75741326  getbl : 3.75741326  id: null |
| **Description:** | This method returns data about the balance of the server wallet.  The "result" field contains a list of address as well as data on the balance of each address on the server.  The "result: total" contains the sum of balances for all addresses.  The "result: getbl" contains result of " **getbalance** " - qtum-daemon command.  Example: <http://176.31.125.26/bridge/?method=balance> |

|  |  |
| --- | --- |
| **Method** | **ipfscat**  No auth |
| **Params** | hash=[string] |
| **Sample response** | jsonrpc: "2.0"  result: "Description for cid 3 (using ipfs)"  id: null |
| **Description:** | This method returns data from ipfs by specified ipfs-hash.  Example: <http://176.31.125.26/bridge/?method=ipfscat&hash=Qmd6TT8mFTNHCJsVCj7m2JgvtXyY7LM5r5gCAt7syqVxSA> |

|  |  |
| --- | --- |
| **Method** | **readbycid**  No auth |
| **Params** | cid=[string] |
| **Sample response** | jsonrpc: "2.0"  result: "“The blockchain community and business world have been running alongside each other in parallel, and I think Qtum is where they meet.”"  id: null |
| **Description:** | This method reads data from the smart contract by specified CID (Content-ID)  The result can contain simple data received from BlockChain, or extended data obtained through an additional call to ipfs, or error.  Error examples:  1) Request undefined CID:  http://176.31.125.26/bridge/?method=readbycid&cid=100000000  Response:  jsonrpc: "2.0"  error :  code : 404  message : "Not found"  id : null  2) Request without CID:  http://176.31.125.26/bridge/?method=readbycid  Response:  jsonrpc: "2.0"  error :  code : 500  message : "Illegal \"cid\" parameter"  id : null  3) Responce when ipfs-daemon is not started:  jsonrpc: "2.0"  error :  code : 500  message : "ipfs daemon not ready"  id : null  Example of a successful query: <http://176.31.125.26/bridge/?method=readbycid&cid=7> |

|  |  |
| --- | --- |
| **Method** | **ownerbycid**  No auth |
| **Params** | cid=[string] |
| **Sample response** | jsonrpc: "2.0"  result : " QQNJ1yL2h1YucDdfXkcoWf41WrbiXZGJma "  id : null |
| **Description:** | This method returns a owner address of the specified CID if it is represented.  The "result" field contains owner address, associated with specified cid.  If specified cid not found, an error of the following type will be returned:  jsonrpc: "2.0"  error :  code : 404  message : "Not found"  id : null  Example of a successful query: <http://176.31.125.26/bridge/?method=ownerbycid&cid=8> |

|  |  |
| --- | --- |
| **Method** | **descrbycid**  No auth |
| **Params** | cid=[string] |
| **Sample response** | jsonrpc: "2.0"  result : "Test description1"  id : null |
| **Description:** | This method returns a description of the specified CID if it is represented.  The "result" field contains description, associated with specified cid.  If there is no description, an error of the following type will be returned:  jsonrpc: "2.0"  error :  code : 404  message : "Not found"  id : null  Example of a successful query: <http://176.31.125.26/bridge/?method=descrbycid&cid=1> |

|  |  |
| --- | --- |
| **Method** | **setdescrforcid**  Authorization required |
| **Params** | cid=[string]  descr=[string]  addr=[string]  blockid=[integer]  secret=[string] |
| **Sample response** | jsonrpc: "2.0"  result :  result :  txid : "07dc68204eebcad993dc332aefb0899fa340b6320b18227d9c212b1b062656a3"  sender : "QQNJ1yL2h1YucDdfXkcoWf41WrbiXZGJma"  hash160 : "2949a0eefcc7b6aa648c0a4845fa68ed1d985dc5"  cid :"2"  descr : "Test description for cid 2"  addr : "QQNJ1yL2h1YucDdfXkcoWf41WrbiXZGJma"  secret : "66774530e770b99241f9880f137202bba88476cfc83fde521379f0fa33f052e8"  blockid : "2f577f41f864"  id : null |
| **Description:** | This method allows set descriptions for the specified CID.  Since this is a blockchain-writing method, it requires authorization and the presence of coins in the account from which the recording will be made.  This function is "reverse" for descrbycid, and this means that data written with this method can be read using descrbycid .  Parameters cid and descr denote the Content-ID and description, respectively.  Parameters addr, blockid, secret requred for all writing api-methods.  The parameter addr means the qtum-address on the server, on behalf of which the Smart-contract will be accessed.  The parameter blockid means result of previous call lastblockid api-method.  The parameter secret means sha256-hash from the concatenation of the following all input parameters in this sequence: api\_pass + blockid + addr + cid + descr  api\_pass is a secure api-key which the client must have to sign the api-request.  Example of a successful query: <[Complexly link](http://176.31.125.26/bridge/?method=setdescrforcid&cid=2&descr=Test description for cid 2&addr=QQNJ1yL2h1YucDdfXkcoWf41WrbiXZGJma&secret=66774530e770b99241f9880f137202bba88476cfc83fde521379f0fa33f052e8&hint=1&blockid=2f577f41f864)> |

|  |  |
| --- | --- |
| **Method** | **makecid**  Authorization required |
| **Params** | cus=[string]  owneraddr=[string]  addr=[string]  blockid=[integer]  secret=[string] |
| **Sample response** | jsonrpc: "2.0"  result :  result :  txid : "175541a6de7276ce5e1c1bf3fe7c12ebd8f55a0892263e13bed2e8f7b79f76ed"  sender : "QQNJ1yL2h1YucDdfXkcoWf41WrbiXZGJma"  hash160 : "2949a0eefcc7b6aa648c0a4845fa68ed1d985dc5"  hash : "QmWSAu67FRCdKFkQDudcMdz9r6k8xocPkyitWqMEmYVB4z"  cus : "Test string"  addr : "QQNJ1yL2h1YucDdfXkcoWf41WrbiXZGJma"  owneraddr : "QQNJ1yL2h1YucDdfXkcoWf41WrbiXZGJma"  owner\_hex\_addr : "2949a0eefcc7b6aa648c0a4845fa68ed1d985dc5"  secret : "feec34cbfc0124bb534c368fa5ed3274fc1e66e49d84f09a06a24ba575b374f6"  blockid : "e5a4f701f886"  id : null |
| **Description:** | This method allows create new CID from specified CUS-data.  Since this is a blockchain-writing method, it requires authorization and the presence of coins in the account from which the recording will be made.  This function is "reverse" for readbycid, and this means that data written with this method can be read using readbycid.  Parameters cus and owneraddr denote the Content-Unical-String and Owner-Address, respectively.  Parameters addr, blockid, secret requred for all writing api-methods.  The parameter addr means the qtum-address on the server, on behalf of which the Smart-contract will be accessed.  The parameter blockid means result of previous call lastblockid api-method.  The parameter secret means sha256-hash from the concatenation of the following all input parameters in this sequence: api\_pass + blockid + addr + owneraddr + cus  api\_pass is a secure api-key which the client must have to sign the api-request.  Example of a successful query: <[Complexly link](http://176.31.125.26/bridge/?method=makecid&cus=Test string&addr=QQNJ1yL2h1YucDdfXkcoWf41WrbiXZGJma&secret=feec34cbfc0124bb534c368fa5ed3274fc1e66e49d84f09a06a24ba575b374f6&blockid=e5a4f701f886)> |

|  |  |
| --- | --- |
| **Method** | **getcid**  No auth |
| **Params** | hash=[string] |
| **Sample response** | jsonrpc: "2.0"  result: "46688951f843"  id: null |
| **Description:** | This method returns CID (Content-ID). |