Package 'gethr'

January 8, 2019

Type Package

Title Access to Ethereum-Based Blockchains Through Geth Nodes

Version 0.1.0

Description Full access to the Geth command line interface for running full Ethereum nodes. With gethr it is possible to carry out different tasks such as mine ether, transfer funds, create contacts, explore block history, etc. The package also provides access to all the available APIs. The officially exposed by Ethereum blockchains (eth, shh, web3, net) and some provided directly by Geth (admin, debug, miner, personal, txpool). For more details on Ethereum, access the project website <https://www.ethereum.org/>. For more details on the Geth client, access the project website <https://github.com/ethereum/go-ethereum/wiki/geth/>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

URL https://github.com/vicegd/gethr

BugReports https://github.com/vicegd/gethr/issues

Imports jsonlite, httr

Suggests testthat

RoxygenNote 6.1.1

NeedsCompilation no

Author Vicente Garcia Diaz [aut, cre]

Maintainer Vicente Garcia Diaz <garciavicente@uniovi.es>

Repository CRAN

Date/Publication 2019-01-08 18:00:20 UTC

R topics documented:

admin_addPeer *New node based on the enode*

Description

admin_addPeer request to add a new remote node to the list of tracked static nodes.

Usage

```
admin_addPeer(enode)
```

Arguments

enode	String - URL of the remote peer to start tracking.

Value

Boolean - true the peer was accepted for tracking or false if some error occurred.

See Also

Other admin functions: admin_datadir, admin_nodeInfo, admin_peers, admin_setSolc, admin_startRPC, admin_startWS, admin_stopRPC, admin_stopWS, gethr

Examples

```
admin_addPeer('enode://c1a07558238c0b31657450dd34a558752d63150ce334f3e99b9187
262b612f48a713a083cd1601bfe3bba761a908264320885633fa81d6d6ca0ef7a6e84a2bcd
  @[127.0.0.1]:30301')
```

admin_datadir *Path being used by the node.*

Description

admin_datadir returns the absolute path of the running Geth node.

Usage

```
admin_datadir()
```

Value

String - Absolute path being used by the node.

See Also

Other admin functions: admin_addPeer, admin_nodeInfo, admin_peers, admin_setSolc, admin_startRPC, admin_startWS, admin_stopRPC, admin_stopWS, gethr

Examples

```
admin_datadir()
```

---

admin_nodeInfo *Node basic information*

Description

admin_nodeInfo returns all the information known about the running Geth node at the networking granularity.

Usage

```
admin_nodeInfo()
```

Details

These include general information about the node itself as a participant of the P2P overlay protocol, as well as specialized information added by each of the running application protocols (e.g. eth, les, shh, bzz).

Value

Object - Properties with information about the node (enode, version, port, ip, id, etc.).

See Also

Other admin functions: admin_addPeer, admin_datadir, admin_peers, admin_setSolc, admin_startRPC, admin_startWS, admin_stopRPC, admin_stopWS, gethr

Examples

```
admin_nodeInfo()
```

admin_peers *Peers basic information*

Description

admin_peers returns all the information known about the connected remote nodes at the networking granularity.

Usage

```
admin_peers()
```

Details

These include general information about the nodes themselves as participants of the P2P overlay protocol, as well as specialized information added by each of the running application protocols (e.g. eth, les, shh, bzz).

Value

Object - Properties with information about the peers.

See Also

Other admin functions: admin_addPeer, admin_datadir, admin_nodeInfo, admin_setSolc, admin_startRPC, admin_startWS, admin_stopRPC, admin_stopWS, gethr

Examples

```
admin_peers()
```

admin_setSolc *Solidity compiler path*

Description

admin_setSolc sets the Solidity compiler path to be used by the node when invoking the eth_compileSolidity RPC method.

Usage

```
admin_setSolc(path)
```

Arguments

path	String - Solidity compiler path defaults to /usr/bin/solc if not set, so you only need to change it for using a non-standard compiler location.

See Also

Other admin functions: admin_addPeer, admin_datadir, admin_nodeInfo, admin_peers, admin_startRPC, admin_startWS, admin_stopRPC, admin_stopWS, gethr

Examples

```
admin_setSolc('/usr/bin/solidityc')
```

---

admin_startRPC *HTTP based JSON RPC API initialization.*

Description

admin_startRPC starts an HTTP based JSON RPC API webserver to handle client requests.

Usage

```
admin_startRPC(host = "127.0.0.1", port = 8545, cors = "",
  apis = "eth,net,web3")
```

Arguments

host	String - Network interface to open the listener socket on.
port	Integer - Network port to open the listener socket on.
cors	String - Cross-origin resource sharing header to use.
apis	String - API modules to offer over this interface.

Value

Boolean - true the execution was successful.

See Also

Other admin functions: admin_addPeer, admin_datadir, admin_nodeInfo, admin_peers, admin_setSolc, admin_startWS, admin_stopRPC, admin_stopWS, gethr

Examples

```
admin_startRPC()
admin_startRPC('193.23.92.1', 8999, '*', 'eth,net,web3,personal,shh,miner,
txpool,admin,debug')
```

---

admin_startWS *WebSocket based JSON RPC API initialization.*

Description

admin_startWS starts a WebSocket based JSON RPC API webserver to handle client requests.

Usage

```
admin_startWS(host = "127.0.0.1", port = 8546, cors = "",
  apis = "eth,net,web3")
```

Arguments

host	String - Network interface to open the listener socket on.
port	Integer - Network port to open the listener socket on.
cors	String - Cross-origin resource sharing header to use.
apis	String - API modules to offer over this interface.

Value

Boolean - true the execution was successful.

See Also

Other admin functions: admin_addPeer, admin_datadir, admin_nodeInfo, admin_peers, admin_setSolc, admin_startRPC, admin_stopRPC, admin_stopWS, gethr

Examples

```
admin_startWS()
admin_startWS('193.23.92.1', 8999, '*', 'eth,net,web3,personal,shh,
miner,txpool,admin,debug')
```

admin_stopRPC *HTTP based JSON RPC API stop.*

Description

admin_stopRPC stops the HTTP based JSON RPC API webserver.

Usage

```
admin_stopRPC()
```

Value

Boolean - true the execution was successful.

See Also

Other admin functions: admin_addPeer, admin_datadir, admin_nodeInfo, admin_peers, admin_setSolc, admin_startRPC, admin_startWS, admin_stopWS, gethr

Examples

```
admin_stopRPC()
```

admin_stopWS *WebSocket based JSON RPC API stop.*

Description

admin_stopWS stops the WebSocket based JSON RPC API webserver.

Usage

```
admin_stopWS()
```

Value

Boolean - true the execution was successful.

See Also

Other admin functions: admin_addPeer, admin_datadir, admin_nodeInfo, admin_peers, admin_setSolc, admin_startRPC, admin_startWS, admin_stopRPC, gethr

Examples

```
admin_stopWS()
```

---

debug_backtraceAt *Logging backtrace location.*

Description

debug_backtraceAt sets the logging backtrace location. When a backtrace location is set and a log message is emitted at that location, the stack of the goroutine executing the log statement will be printed to stderr.

Usage

```
debug_backtraceAt(path)
```

Arguments

path	String - Backtrace location.

See Also

Other debug functions: debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRate, debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_backtraceAt('my_server.go:443')
```

debug_blockProfile *Block profiling activation.*

Description

debug_blockProfile turns on block profiling for the given duration and writes profile data to disk. It uses a profile rate of 1 for most accurate information.

Usage

```
debug_blockProfile(file, seconds)
```

Arguments

file	String - File to write the data.
seconds	Integer - Seconds to write the data.

See Also

Other debug functions: debug_backtraceAt, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRate, debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_blockProfile('file.log', 5)
```

debug_cpuProfile *CPU profiling activation.*

Description

debug_cpuProfile turns on CPU profiling for the given duration and writes profile data to disk.

Usage

```
debug_cpuProfile(file, seconds)
```

Arguments

file	String - File to write the data.
seconds	Integer - Seconds to write the data.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRate, debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_cpuProfile('file.log', 5)
```

---

debug_dumpBlock *State of the block.*

Description

debug_dumpBlock retrieves the state that corresponds to the block number and returns a list of accounts (including storage and code).

Usage

```
debug_dumpBlock(number)
```

Arguments

number	Integer - Number of the block.

Value

Object - Information of the state (balance, code, hash, nonce, root and storage).

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRate, debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_dumpBlock(15)
```

---

debug_gcStats *Garbage collector statistics.*

Description

debug_gcStats returns garbage collector statistics.

Usage

```
debug_gcStats()
```

Value

Object - Information about the gargabe collector operation.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRate, debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_gcStats()
```

debug_getBlockRlp *Recursive Length Prefix of the block.*

Description

debug_getBlockRlp retrieves and returns the Recursive Length Prefix by number of block.

Usage

```
debug_getBlockRlp(number)
```

Arguments

number	Integer - Number of the block.

Value

Data - Recursive Length Prefix of the block.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRate, debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_getBlockRlp(29)
```

debug_goTrace *Go runtime tracing activation.*

Description

debug_goTrace turns on Go runtime tracing for the given duration and writes trace data to disk.

Usage

```
debug_goTrace(file, seconds)
```

Arguments

file	String - File to write the data.
seconds	Integer - Seconds to write the data.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_memStats, debug_seedHash, debug_setBlockProfileRate, debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_goTrace('file.log', 5)
```

---

debug_memStats *Runtime memory statistics*

Description

debug_memStats returns detailed runtime memory statistics.

Usage

```
debug_memStats()
```

Value

Object - Detailed runtime memory statistics.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_seedHash, debug_setBlockProfileRate, debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_memStats()
```

---

debug_seedHash *Seed hash of the block*

Description

debug_seedHash fetches and retrieves the seed hash of the block by number.

Usage

```
debug_seedHash(number)
```

Arguments

number	Integer - Number of the block.

Value

Data - Seed hash of the block by number.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_setBlockProfileRate, debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_seedHash(29)
```

debug_setBlockProfileRate *Block profile data collection rate.*

Description

debug_setBlockProfileRate sets the rate (in samples/sec) of goroutine block profile data collection.

Usage

```
debug_setBlockProfileRate(rate)
```

Arguments

rate	Integer - Rate of the profiling. A non-zero rate enables block profiling, setting it to zero stops the profile.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_setBlockProfileRate(100)
```

debug_setHead *Current head of the chain configuration.*

Description

debug_setHead sets the current head of the local chain by block number. Note, this is a destructive action and may severely damage your chain. Use with extreme caution.

Usage

```
debug_setHead(number)
```

Arguments

number	Integer - Number of the block.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRat debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_setHead(29)
```

---

debug_stacks *Printer representation of the stacks.*

Description

debug_stacks returns a printed representation of the stacks of all goroutines.

Usage

```
debug_stacks()
```

Value

String - Representation of the stacks of all goroutines. Note that the web3 wrapper for this method takes care of the printing and does not return the string.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRat debug_setHead, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_stacks()
```

---

debug_startCPUProfile *CPU profiling permanent activation.*

Description

debug_startCPUProfile turns on CPU profiling indefinitely, writing to the given file.

Usage

```
debug_startCPUProfile(file)
```

Arguments

file	String - File to write the data.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRat debug_setHead, debug_stacks, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_startCPUProfile('file.log')
```

debug_startGoTrace *Go runtime trace permanent activation.*

Description

debug_startGoTrace starts writing a Go runtime trace to the given file.

Usage

```
debug_startGoTrace(file)
```

Arguments

file	String - File to write the data.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRat debug_setHead, debug_stacks, debug_startCPUProfile, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_startGoTrace('file.log')
```

debug_stopCPUProfile *CPU profiling stop.*

Description

debug_stopCPUProfile turns off CPU profiling.

Usage

```
debug_stopCPUProfile()
```

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRat debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_stopCPUProfile()
```

---

debug_stopGoTrace *Go trace stop.*

Description

debug_stopGoTrace turns off Go trace.

Usage

```
debug_stopGoTrace()
```

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRat debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_stopGoTrace()
```

debug_traceBlock *Full trace of a block.*

Description

debug_traceBlock returns a full stack trace of all invoked opcodes of all transaction that were included in this block.

Usage

```
debug_traceBlock(block)
```

Arguments

block	Data - Recursive Length Prefix of the block.

Value

Object - Full stack trace of the block.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRat debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_traceBlock('0xf90217f90212a0a8ebc73b3bf1b5f81f62512d3ebfe349c95d91a3a21
71f8f6881b31e2f523000a01dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142
fd40d4934794b117a8bc3ecf2c3f006b89da6826e49b4193977aa024b20bd98ef519ccde0135d
29161a2ffec97cd7a3596679ccb7ff7ab5deaf73ea056e81f171bcc55a6ff8345e692c0f86e5b
48e01b996cadc001622fb5e363b421a056e81f171bcc55a6ff8345e692c0f86e5b48e01b996ca
dc001622fb5e363b421b901000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000083
0207001d8349f81980845bfc1f189bda8301081184676574688867 6f312e31312e318777696e6
46f7773a0979e83b4cbcc9b0bcec5e5e468fa64dc214724da7dc8a306af11227d70ba9a678815
5c0c88c5d3d7d7c0c0')
```

debug_traceBlockByHash *Full trace of a block by hash.*

Description

debug_traceBlockByHash returns a full stack trace of all invoked opcodes of all transaction that were included in this block by hash.

Usage

```
debug_traceBlockByHash(hash)
```

Arguments

hash	Data - Block hash.

Value

Object - Full stack trace of the block.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRat debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_traceBlockByHash('0xc41e977760873fbab207d0921bc124edce6ce4044d2718b2bba1
ebec45656ffa')
```

debug_traceBlockByNumber

Full trace of a block by number.

Description

debug_traceBlockByNumber returns a full stack trace of all invoked opcodes of all transaction that were included in this block by number.

Usage

```
debug_traceBlockByNumber(number)
```

Arguments

number	Integer - Block number.

Value

Object - Full stack trace of the block.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRat debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_traceBlockByNumber(43337)
```

debug_traceBlockFromFile

Full trace of a block using a file as input.

Description

debug_traceBlockFromFile returns a full stack trace of all invoked opcodes of all transaction that were included in this block using a file as input.

Usage

```
debug_traceBlockFromFile(file)
```

Arguments

file	String - File containing the Recursive Length Prefix of the block.

Value

Object - Full stack trace of the block.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRat debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_traceBlockFromFile('rlp.log')
```

---

debug_traceTransaction

Transaction tracing.

Description

debug_traceTransaction attempts to run the transaction in the exact same manner as it was executed on the network. It will replay any transaction that may have been executed prior to this one before it will finally attempt to execute the transaction that corresponds to the given hash.

Usage

```
debug_traceTransaction(hash, disableStorage = NULL,
  disableMemory = NULL, disableStack = NULL, tracer = NULL,
  timeout = NULL)
```

Arguments

hash	Data - Hash of the transaction.
disableStorage	Boolean - Setting this to true will disable storage capture (default = false).
disableMemory	Boolean - Setting this to true will disable memory capture (default = false)
disableStack	Boolean - Setting this to true will disable stack capture (default = false)
tracer	String - Setting this will enable JavaScript-based transaction tracing. If set, the previous arguments will be ignored.
timeout	String - Overrides the default timeout of 5 seconds for JavaScript-based tracing calls.

Value

Object - Full trace of the transaction.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRat debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_traceTransaction('0xe45f738135240ce9cedc58b21148ef424704e5c86798990a5
a36cb1ca4c5c3f4')
debug_traceTransaction('0xe45f738135240ce9cedc58b21148ef424704e5c86798990a5
a36cb1ca4c5c3f4', disableStorage = TRUE, disableStack = TRUE)
```

---

debug_verbosity *Logging verbosity.*

Description

debug_verbosity sets the logging verbosity ceiling. Log messages with level up to and including the given level will be printed.

Usage

```
debug_verbosity(level)
```

Arguments

level	Integer - File containing the Recursive Length Prefix of the block.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRat debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_verbosity(5)
```

---

debug_vmodule *Logging verbosity by pattern.*

Description

debug_vmodule sets the logging verbosity pattern.

Usage

```
debug_vmodule(pattern)
```

Arguments

pattern	String - Pattern used to indicate the verbosity.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRat debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_writeBlockProfile, debug_writeMemProfile, gethr

Examples

```
debug_vmodule('eth/*=4')
debug_vmodule('p2p=5')
```

---

debug_writeBlockProfile

Goroutine blocking profile to file.

Description

debug_writeBlockProfile writes a goroutine blocking profile to the given file.

Usage

```
debug_writeBlockProfile(file)
```

Arguments

file	String - File to write the data.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRat debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeMemProfile, gethr

Examples

```
debug_writeBlockProfile('file.log')
```

debug_writeMemProfile *Allocation profile to file.*

Description

debug_writeMemProfile writes an allocation profile to the given file.

Usage

```
debug_writeMemProfile(file)
```

Arguments

file	String - File to write the data.

See Also

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRat debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, gethr

Examples

```
debug_writeMemProfile('file.log')
```

dec_to_hex *Decimal to hexadecimal conversion.*

Description

dec_to_hex returns the value in hexadecimal.

Usage

```
dec_to_hex(number)
```

Arguments

number	Integer - Value in decimal.

Value

Data - Value in hexadecimal.

See Also

Other utils functions: get_network_id, get_post, get_rpc_address, hex_to_dec, hex_to_text, is.wholenumber, process_block, process_log, process_receipt, process_transaction, set_network_id, set_rpc_address, text_to_hex

Examples

```
dec_to_hex(50)
```

---

ether.toEther *Conversion to Ether.*

Description

ether.toEther returns the value of the cryptocurrency in Ether.

Usage

```
ether.toEther(amount, type)
```

Arguments

amount	Integer - Amount of the cryptocurrency to convert from.
type	String - Unit of the cryptocurrency to convert from: wei, kwei, mwei, gwei, szabo, finney, ether, kether, mether, gether or tether.

Value

Integer - Value in Ether.

See Also

Other ether functions: ether.toFinney, ether.toGether, ether.toGwei, ether.toKether, ether.toKwei, ether.toMether, ether.toMwei, ether.toSzabo, ether.toTether, ether.toWei, gethr

Examples

```
ether.toEther(50, 'wei')
```

ether.toFinney *Conversion to Finney.*

Description

`ether.toFinney` returns the value of the cryptocurrency in Finney.

Usage

```
ether.toFinney(amount, type)
```

Arguments

`amount`	Integer - Amount of the cryptocurrency to convert from.
`type`	String - Unit of the cryptocurrency to convert from: wei, kwei, mwei, gwei, szabo, finney, ether, kether, mether, gether or tether.

Value

Integer - Value in Finney.

See Also

Other ether functions: `ether.toEther`, `ether.toGether`, `ether.toGwei`, `ether.toKether`, `ether.toKwei`, `ether.toMether`, `ether.toMwei`, `ether.toSzabo`, `ether.toTether`, `ether.toWei`, `gethr`

Examples

```
ether.toFinney(50, 'ether')
```

ether.toGether *Conversion to Gether.*

Description

`ether.toGether` returns the value of the cryptocurrency in Gether.

Usage

```
ether.toGether(amount, type)
```

Arguments

`amount`	Integer - Amount of the cryptocurrency to convert from.
`type`	String - Unit of the cryptocurrency to convert from: wei, kwei, mwei, gwei, szabo, finney, ether, kether, mether, gether or tether.

Value

Integer - Value in Gether.

See Also

Other ether functions: ether.toEther, ether.toFinney, ether.toGwei, ether.toKether, ether.toKwei, ether.toMether, ether.toMwei, ether.toSzabo, ether.toTether, ether.toWei, gethr

Examples

```
ether.toGether(50, 'ether')
```

---

ether.toGwei *Conversion to Gwei.*

Description

ether.toGwei returns the value of the cryptocurrency in Gwei.

Usage

```
ether.toGwei(amount, type)
```

Arguments

amount	Integer - Amount of the cryptocurrency to convert from.
type	String - Unit of the cryptocurrency to convert from: wei, kwei, mwei, gwei, szabo, finney, ether, kether, mether, gether or tether.

Value

Integer - Value in Gwei.

See Also

Other ether functions: ether.toEther, ether.toFinney, ether.toGether, ether.toKether, ether.toKwei, ether.toMether, ether.toMwei, ether.toSzabo, ether.toTether, ether.toWei, gethr

Examples

```
ether.toGwei(50, 'ether')
```

ether.toKether *Conversion to Kether.*

Description

`ether.toKether` returns the value of the cryptocurrency in Kether.

Usage

```
ether.toKether(amount, type)
```

Arguments

`amount`	Integer - Amount of the cryptocurrency to convert from.
`type`	String - Unit of the cryptocurrency to convert from: wei, kwei, mwei, gwei, szabo, finney, ether, kether, mether, gether or tether.

Value

Integer - Value in Kether.

See Also

Other ether functions: `ether.toEther`, `ether.toFinney`, `ether.toGether`, `ether.toGwei`, `ether.toKwei`, `ether.toMether`, `ether.toMwei`, `ether.toSzabo`, `ether.toTether`, `ether.toWei`, `gethr`

Examples

```
ether.toKether(50, 'ether')
```

ether.toKwei *Conversion to Kwei.*

Description

`ether.toKwei` returns the value of the cryptocurrency in Kwei.

Usage

```
ether.toKwei(amount, type)
```

Arguments

`amount`	Integer - Amount of the cryptocurrency to convert from.
`type`	String - Unit of the cryptocurrency to convert from: wei, kwei, mwei, gwei, szabo, finney, ether, kether, mether, gether or tether.

Value

Integer - Value in Kwei.

See Also

Other ether functions: ether.toEther, ether.toFinney, ether.toGether, ether.toGwei, ether.toKether, ether.toMether, ether.toMwei, ether.toSzabo, ether.toTether, ether.toWei, gethr

Examples

```
ether.toKwei(50, 'ether')
```

---

ether.toMether *Conversion to Mether.*

Description

ether.toMether returns the value of the cryptocurrency in Mether.

Usage

```
ether.toMether(amount, type)
```

Arguments

amount	Integer - Amount of the cryptocurrency to convert from.
type	String - Unit of the cryptocurrency to convert from: wei, kwei, mwei, gwei, szabo, finney, ether, kether, mether, gether or tether.

Value

Integer - Value in Mether.

See Also

Other ether functions: ether.toEther, ether.toFinney, ether.toGether, ether.toGwei, ether.toKether, ether.toKwei, ether.toMwei, ether.toSzabo, ether.toTether, ether.toWei, gethr

Examples

```
ether.toMether(50, 'ether')
```

ether.toMwei *Conversion to Mwei.*

Description

`ether.toMei` returns the value of the cryptocurrency in Mwei.

Usage

```
ether.toMwei(amount, type)
```

Arguments

`amount`	Integer - Amount of the cryptocurrency to convert from.
`type`	String - Unit of the cryptocurrency to convert from: wei, kwei, mwei, gwei, szabo, finney, ether, kether, mether, gether or tether.

Value

Integer - Value in Mwei.

See Also

Other ether functions: `ether.toEther`, `ether.toFinney`, `ether.toGether`, `ether.toGwei`, `ether.toKether`, `ether.toKwei`, `ether.toMether`, `ether.toSzabo`, `ether.toTether`, `ether.toWei`, `gethr`

Examples

```
ether.toMwei(50, 'ether')
```

ether.toSzabo *Conversion to Szabo.*

Description

`ether.toSzabo` returns the value of the cryptocurrency in Szabo.

Usage

```
ether.toSzabo(amount, type)
```

Arguments

`amount`	Integer - Amount of the cryptocurrency to convert from.
`type`	String - Unit of the cryptocurrency to convert from: wei, kwei, mwei, gwei, szabo, finney, ether, kether, mether, gether or tether.

Value

Integer - Value in Szabo.

See Also

Other ether functions: ether.toEther, ether.toFinney, ether.toGether, ether.toGwei, ether.toKether, ether.toKwei, ether.toMether, ether.toMwei, ether.toTether, ether.toWei, gethr

Examples

```
ether.toSzabo(50, 'ether')
```

---

ether.toTether *Conversion to Tether.*

Description

ether.toTether returns the value of the cryptocurrency in Tether.

Usage

```
ether.toTether(amount, type)
```

Arguments

amount	Integer - Amount of the cryptocurrency to convert from.
type	String - Unit of the cryptocurrency to convert from: wei, kwei, mwei, gwei, szabo, finney, ether, kether, mether, gether or tether.

Value

Integer - Value in Tether.

See Also

Other ether functions: ether.toEther, ether.toFinney, ether.toGether, ether.toGwei, ether.toKether, ether.toKwei, ether.toMether, ether.toMwei, ether.toSzabo, ether.toWei, gethr

Examples

```
ether.toTether(50, 'ether')
```

ether.toWei *Conversion to Wei.*

Description

`ether.toWei` returns the value of the cryptocurrency in Wei.

Usage

```
ether.toWei(amount, type)
```

Arguments

`amount`	Integer - Amount of the cryptocurrency to convert from.
`type`	String - Unit of the cryptocurrency to convert from: wei, kwei, mwei, gwei, szabo, finney, ether, kether, mether, gether or tether.

Value

Integer - Value in Wei.

See Also

Other ether functions: `ether.toEther`, `ether.toFinney`, `ether.toGether`, `ether.toGwei`, `ether.toKether`, `ether.toKwei`, `ether.toMether`, `ether.toMwei`, `ether.toSzabo`, `ether.toTether`, `gethr`

Examples

```
ether.toWei(50, 'ether')
```

eth_accounts *Addresses owned by client.*

Description

`eth_accounts` returns a list of addresses owned by the client.

Usage

```
eth_accounts()
```

Value

Array of Address - Addresses owned by the client.

See Also

Other eth functions: eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHash, eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_accounts()
```

---

eth_blockNumber *Current block number.*

Description

`eth_blockNumber` returns the number of most recent block.

Usage

```
eth_blockNumber()
```

Value

Integer - Current block number the client is on

See Also

Other eth functions: eth_accounts, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHash, eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_blockNumber()
```

`eth_call` *New message call.*

Description

`eth_call` executes a new message call immediately without creating a transaction on the block chain.

Usage

```
eth_call(from = -1, to, gas = 30000, gas_price = -1, value = -1,
  data = -1, number = "latest")
```

Arguments

`from`	Address - Address the call is send from.
`to`	Address - Address the call is send to.
`gas`	Integer - Gas provided for the call execution. `eth_call` consumes zero gas, but this parameter may be needed by some executions.
`gas_price`	Integer - Value of the gas for this call.
`value`	Integer - Value sent with the call
`data`	Data - Hash of the method signature and encoded parameters. For details see Ethereum Contract ABI https://solidity.readthedocs.io/en/develop/abi-spec.html.
`number`	Integer|Tag - Integer block number, or the string 'latest', 'earliest' or 'pending', see the default block parameter https://github.com/ethereum/wiki/wiki/JSON-RPC#the-default-block-parameter

Value

Data - Return value of executed call.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHash`, `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberAndIndex`, `eth_getTransactionByHash`, `eth_getTransactionCount`,

eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_call(to = '0xcaf9a0356ddfa779fdbb55c45b22d35673550f30', data = '0x8f8a0d1900
0000000000000000000000000000000000000000000000000000000000000060000000000000000000
0000000000000000000000000000000000000000000000000000000000000002')
```

---

eth_coinbase *Coinbase address.*

Description

eth_coinbase returns the client coinbase address.

Usage

```
eth_coinbase()
```

Value

Address - Current coinbase address.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHash, eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_coinbase()
```

`eth_estimateGas` *Estimation of the gas.*

Description

`eth_estimateGas` returns an estimate of how much gas is necessary to allow the transaction to complete.

Usage

```
eth_estimateGas(to, data)
```

Arguments

`to`	Address - Address the call is send to.
`data`	Data - Hash of the method signature and encoded parameters. For details see Ethereum Contract ABI https://solidity.readthedocs.io/en/develop/abi-spec.html.

Details

The transaction will not be added to the blockchain. Note that the estimate may be significantly more than the amount of gas actually used by the transaction, for a variety of reasons including EVM mechanics and node performance.

Value

Integer - Amount of gas estimated.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHash`, `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberAndIndex`, `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAndIndex`, `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter`, `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_estimateGas(to = '0xcaf9a0356ddfa779fdbb55c45b22d35673550f30', data =
'0x8f8a0d19000000000000000000000000000000000000000000000000000000000000
06000000000000000000000000000000000000000000000000000000000000000002')
```

`eth_gasPrice` *Gas price.*

Description

`eth_gasPrice` returns the current price per gas in `weis`.

Usage

```
eth_gasPrice()
```

Value

Integer - Current gas price in wei.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHash`, `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberAndIndex`, `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAndIndex`, `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter`, `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_gasPrice()
```

`eth_getBalance` *Balance of an account.*

Description

`eth_getBalance` returns the balance of the account of the given address.

Usage

```
eth_getBalance(address = NULL, number = "latest")
```

Arguments

`address`	Address - Address to check the balance.
`number`	Integer|Tag - Block number, or the string 'latest', 'earliest' or 'pending'.

Value

Integer - Current balance in wei.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHash`, `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberAndIndex`, `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAndIndex`, `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter`, `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_getBalance()
eth_getBalance('0xb117a8bc3ecf2c3f006b89da6826e49b4193977a')
eth_getBalance('0xb117a8bc3ecf2c3f006b89da6826e49b4193977a', 'earliest')
eth_getBalance('0xb117a8bc3ecf2c3f006b89da6826e49b4193977a', 5)
```

`eth_getBlockByHash` *Block information based on its hash.*

Description

`eth_getBlockByHash` returns information about a block by hash.

Usage

```
eth_getBlockByHash(hash, full = TRUE, hex = TRUE)
```

Arguments

`hash`	Hash - Hash of the block.
`full`	Boolean - If `true` it returns the full transaction objects, if `false` only the hashes of the transactions.
`hex`	Boolean - `true` to get the response in hexadecimal, `false` to get a readable response.

Value

Object - A block object, or `null` when no block was found.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHash`, `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberAndIndex`, `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAndIndex`, `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter`, `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_getBlockByHash('0xb69e76f3997f318f4385f31885576aa43cb40ad4ed8938718e1503
20ff48f528')
eth_getBlockByHash('0xb69e76f3997f318f4385f31885576aa43cb40ad4ed8938718e1503
20ff48f528', FALSE, FALSE)
```

`eth_getBlockByNumber` *Block information based on its number.*

Description

`eth_getBlockByNumber` returns information about a block by block number.

Usage

```
eth_getBlockByNumber(number = "latest", full = TRUE, hex = TRUE)
```

Arguments

`number`	Integer|Tag - Block number, or the string 'latest', 'earliest' or 'pending'.
`full`	Boolean - If `true` it returns the full transaction objects, if `false` only the hashes of the transactions.
`hex`	Boolean - `true` to get the response in hexadecimal, `false` to get a readable response.

Value

Object - A block object, or `null` when no block was found.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockTransactionCountByHash`, `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberAndIndex`, `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAndIndex`, `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter`, `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_getBlockByNumber(42364)
eth_getBlockByNumber(42364, FALSE, FALSE)
```

eth_getBlockTransactionCountByHash
Transactions in a block given a hash.

Description

eth_getBlockTransactionCountByHash returns the number of transactions in a block from a block matching the given block hash.

Usage

```
eth_getBlockTransactionCountByHash(hash)
```

Arguments

hash	Hash - Hash of the block.

Value

Integer - Number of transactions in the block.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByNu eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHa eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_getBlockTransactionCountByHash('0x6e4670b7fda89b5960e684d4c809f7e7e9d9c0
ee70b43849405efe78aa3c0d24')
```

`eth_getBlockTransactionCountByNumber` *Transactions in a block given a number.*

Description

`eth_getBlockTransactionCountByNumber` returns the number of transactions in a block from a block matching the given block number.

Usage

```
eth_getBlockTransactionCountByNumber(number)
```

Arguments

`number`	Integer - Number of the block.

Value

Integer - Number of transactions in the block.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHa` `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberAndIndex`, `eth_getTransactionByHa` `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAndIndex`, `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter` `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_getBlockTransactionCountByNumber(38038)
```

eth_getCode *Code at an address.*

Description

eth_getCode returns code at a given address.

Usage

```
eth_getCode(address, number = "latest")
```

Arguments

address	Address - Address to get the code.
number	Integer|Tag - Block number, or the string 'latest', 'earliest' or 'pending'.

Value

Data - Code from the given address.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberA eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAn eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_getCode('0xcaf9a0356ddfa779fdbb55c45b22d35673550f30')
eth_getCode('0xcaf9a0356ddfa779fdbb55c45b22d35673550f30', 'earliest')
eth_getCode('0xcaf9a0356ddfa779fdbb55c45b22d35673550f30', 25)
```

`eth_getFilterChanges` *Filter information since last poll.*

Description

`eth_getFilterChanges` returns an array of logs which occurred since last poll.

Usage

```
eth_getFilterChanges(id, hex = TRUE)
```

Arguments

`id`	Integer - Filter Id.
`hex`	Boolean - `true` to get the response in hexadecimal, `false` to get a readable response.

Value

Array of Object - Logs for a given filter since last poll.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHa` `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberA` `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAn` `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter` `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_getFilterChanges('0x75c1c2893a789a4cfb8e146464ea622b')
```

`eth_getFilterLogs` *Filter information.*

Description

`eth_getFilterLogs` returns an array of all logs matching filter with given id.

Usage

```
eth_getFilterLogs(id, hex = TRUE)
```

Arguments

`id`	Integer - Filter Id.
`hex`	Boolean - `true` to get the response in hexadecimal, `false` to get a readable response.

Value

Array of Object - Logs for a given filter.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHa` `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberA` `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAn` `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter` `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_getFilterLogs('0x75c1c2893a789a4cfb8e146464ea622b')
```

eth_getLogs *New filter and its logs.*

Description

eth_getLogs returns an array of all logs matching a given filter object.

Usage

```
eth_getLogs(from_block = "earliest", to_block = "latest", address,
  topics = -1, block_hash = -1)
```

Arguments

from_block	Integer|Tag - Block number, or the string 'earliest', 'latest' or 'pending'.
to_block	Integer|Tag - Block number, or the string 'earliest', 'latest' or 'pending'.
address	Address - Contract address or a list of addresses from which logs should originate.
topics	Array of Data - Topics are order-dependent. Each topic can also be an array of DATA with 'or' options.
block_hash	Hash Single block to return logs. If blockHash is present in the filter criteria, then neither fromBlock nor toBlock are allowed.

Value

Hash - A filter Id.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumber/ eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAr eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_getLogs(0, 100, '0xcaf9a0356ddfa779fdbb55c45b22d35673550f30',
list('0x977f31fe2eae427d123315e068c90016b9f8c44b9c8d0818a740f06d2dc10f95',
'0x0000000000000000000000000000000000000000000000000000000000000003'))
eth_getLogs(address = '0x8655bd257db96eb2aca7154f845d6b1d67689219')
```

eth_getProof *Value from storage position.*

Description

eth_getProof returns the account and storage values of the specified account including the Merkle-proof.

Usage

```
eth_getProof(address, keys, number = "latest")
```

Arguments

address	Address - Address of the account or contract.
keys	Array of Data - Storage-keys which should be proofed and included.
number	Integer|Tag - Block number, or the string 'latest', 'earliest' or 'pending'.

Value

Data - Value at this storage position.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAr eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAr eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_getProof('0x8655bd257db96eb2aca7154f845d6b1d67689219',
list('0x0000000000000000000000000000000000000000000000000000000000000000'))
```

`eth_getStorageAt` *Value from storage position.*

Description

`eth_getStorageAt` returns the value from a storage position at a given address.

Usage

```
eth_getStorageAt(address, position, number = "latest")
```

Arguments

`address`	Address - Address of the storage.
`position`	Integer - Position in the storage.
`number`	Integer|Tag - Block number, or the string 'latest', 'earliest' or 'pending'.

Value

Data - Value at this storage position.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHa` `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberAndInd` `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAr` `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter` `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_getStorageAt('0xcaf9a0356ddfa779fdbb55c45b22d35673550f30', '0x0')
eth_getStorageAt('0xcaf9a0356ddfa779fdbb55c45b22d35673550f30', '0x0', 'latest')
eth_getStorageAt('0xcaf9a0356ddfa779fdbb55c45b22d35673550f30', '0x0', 10)
```

`eth_getTransactionByBlockHashAndIndex` *Transaction information given a block hash and an index position.*

Description

`eth_getTransactionByBlockHashAndIndex` returns information about a transaction by block hash and transaction index position.

Usage

```
eth_getTransactionByBlockHashAndIndex(block_hash, trans_index,
  hex = TRUE)
```

Arguments

`block_hash`	Hash - Hash of a block.
`trans_index`	Integer - Transaction index position.
`hex`	Boolean - `true` to get the response in hexadecimal, `false` to get a readable response.

Value

Object - A transaction object, or *null* when no transaction was found.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHa` `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockNumberAndIndex`, `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAr` `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter` `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_getTransactionByBlockHashAndIndex('0xb69e76f3997f318f4385f31885576aa43cb
40ad4ed8938718e150320ff48f528', 0)
eth_getTransactionByBlockHashAndIndex('0x9f965e1d082ffe457247befe92089d8fc2e
63015c12f373bd0d61d3ab76c6d05', 10, FALSE)
```

`eth_getTransactionByBlockNumberAndIndex` *Transaction information given a block number and an index position.*

Description

`eth_getTransactionByBlockHashAndIndex` returns information about a transaction by block number and transaction index position.

Usage

```
eth_getTransactionByBlockNumberAndIndex(block_number, trans_index,
  hex = TRUE)
```

Arguments

`block_number`	Integer|Tag - Block number, or the string 'earliest', 'latest' or 'pending'.
`trans_index`	Integer - Transaction index position.
`hex`	Boolean - `true` to get the response in hexadecimal, `false` to get a readable response.

Value

Object - A transaction object, or *null* when no transaction was found.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHa` `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAr` `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter` `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_getTransactionByBlockNumberAndIndex(42364, 0)
eth_getTransactionByBlockNumberAndIndex(42364, 1, FALSE)
eth_getTransactionByBlockNumberAndIndex(500000, 10, FALSE)
eth_getTransactionByBlockNumberAndIndex(42365, 2)
```

eth_getTransactionByHash *Transaction information given a transaction hash.*

Description

eth_getTransactionByHash returns the information about a transaction requested by transaction hash.

Usage

```
eth_getTransactionByHash(hash, hex = TRUE)
```

Arguments

hash	Hash - Hash of the transaction.
hex	Boolean - true to get the response in hexadecimal, false to get a readable response.

Value

Object - A transaction object, or *null* when no transaction was found.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_getTransactionByHash('0xb61a9ca11109646bfd056f8be9e1e183a1b1bea3c281e73c
c4f17d332fa69a05')
eth_getTransactionByHash('0xcb33fb7850764aefd3cedd3dcae186cbd8bda74ca2822e4c
59115b1c6b5c48bf', FALSE)
```

`eth_getTransactionCount` *Transactions an address has sent.*

Description

`eth_getTransactionCount` returns the number of transactions sent from an address.

Usage

```
eth_getTransactionCount(address = NULL, number = "latest")
```

Arguments

`address`	Address - Address of the storage.
`number`	Integer|Tag - Block number, or the string 'latest', 'earliest' or 'pending'.

Value

Integer - Number of transactions send from this address.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHa` `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberAndIndex`, `eth_getTransactionByHash`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAndIndex`, `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter`, `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_getTransactionCount()
eth_getTransactionCount('0xb117a8bc3ecf2c3f006b89da6826e49b4193977a')
eth_getTransactionCount('0xb117a8bc3ecf2c3f006b89da6826e49b4193977a', 'pending')
eth_getTransactionCount('0xb117a8bc3ecf2c3f006b89da6826e49b4193977a', 10)
```

`eth_getTransactionReceipt` *Transaction receipt.*

Description

`eth_getTransactionReceipt` returns the receipt of a transaction by transaction hash.

Usage

```
eth_getTransactionReceipt(hash, hex = TRUE)
```

Arguments

`hash`	Hash - Hash of a transaction.
`hex`	Boolean - `true` to get the response in hexadecimal, `false` to get a readable response.

Value

Object - A transaction receipt, or *null* when no receipt was found.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHa` `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberAndIndex`, `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getUncleByBlockHashAndIndex`, `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter`, `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_getTransactionReceipt('0xb61a9ca11109646bfd056f8be9e1e183a1b1bea3c281e
73cc4f17d332fa69a05')
eth_getTransactionReceipt('0xb61a9ca11109646bfd056f8be9e1e183a1b1bea3c281e
73cc4f17d332fa69a05', FALSE)
```

`eth_getUncleByBlockHashAndIndex` *Uncle information given a block hash and an index position.*

Description

`eth_getUncleByBlockHashAndIndex` returns information about a uncle of a block by hash and uncle index position.

Usage

```
eth_getUncleByBlockHashAndIndex(block_hash, uncle_index, hex = TRUE)
```

Arguments

`block_hash`	Hash - Hash of a block.
`uncle_index`	Integer - Uncle index position.
`hex`	Boolean - `true` to get the response in hexadecimal, `false` to get a readable response.

Value

Object - A block object, or *null* when no block was found.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHa` `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberAndIndex`, `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter`, `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_getUncleByBlockHashAndIndex('0xb69e76f3997f318f4385f31885576aa43cb40ad4ed
8938718e150320ff48f528', 1)
```

eth_getUncleByBlockNumberAndIndex *Uncle information given a block number and an index position.*

Description

`eth_getUncleByBlockHashAndIndex` returns information about a uncle of a block by number and uncle index position.

Usage

```
eth_getUncleByBlockNumberAndIndex(block_number, uncle_index, hex = TRUE)
```

Arguments

`block_number`	Integer|Tag - Block number, or the string 'earliest', 'latest' or 'pending'.
`uncle_index`	Integer - Uncle index position.
`hex`	Boolean - `true` to get the response in hexadecimal, `false` to get a readable response.

Value

Object - A block object, or *null* when no block was found.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHa` `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberAndIndex`, `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter`, `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_getUncleByBlockNumberAndIndex(42364, 345)
```

`eth_getUncleCountByBlockHash` *Uncles in a block given a hash.*

Description

`eth_getUncleCountByBlockHash` returns the number of uncles in a block from a block matching the given block hash.

Usage

```
eth_getUncleCountByBlockHash(hash)
```

Arguments

`hash`	Hash - Hash of the block.

Value

Integer - Number of uncles in the block.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHa` `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberAndIndex`, `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAndIndex`, `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter`, `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_getUncleCountByBlockHash('0x6e4670b7fda89b5960e684d4c809f7e7e9d9c0ee70b
43849405efe78aa3c0d24')
```

`eth_getUncleCountByBlockNumber` *Uncles in a block given a number.*

Description

`eth_getUncleCountByBlockNumber` returns the number of uncles in a block from a block matching the given block number.

Usage

```
eth_getUncleCountByBlockNumber(number)
```

Arguments

`number`	Integer - Number of the block.

Value

Integer - Number of uncles in the block.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHa` `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberAndIndex`, `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAndIndex`, `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter`, `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_getUncleCountByBlockNumber(38038)
```

eth_getWork *Work performed by the current block*

Description

eth_getWork returns the hash of the current block, the seedHash, and the boundary condition to meet the target.

Usage

```
eth_getWork()
```

Value

Object - Information about the block header pow-hash, the seed hash used for the DAG and the boundary condition / target.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_getWork()
```

eth_hashrate *Hashes per second that are mined.*

Description

eth_hashrate returns the number of hashes per second that the node is mining with.

Usage

```
eth_hashrate()
```

Value

Integer - Number of hashes per second.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_hashrate()
```

---

eth_mining *Whether client is mining or not.*

Description

`eth_mining` returns `true` if client is actively mining new blocks.

Usage

```
eth_mining()
```

Value

Boolean - `true` if the client is mining, otherwise `false`.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion,

eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_mining()
```

`eth_newBlockFilter` *New block filter.*

Description

`eth_newBlockFilter` creates a filter in the node, to notify when a new block arrives.

Usage

```
eth_newBlockFilter()
```

Value

Hash - A filter Id.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_newBlockFilter()
```

`eth_newFilter` *New filter.*

Description

`eth_newFilter` creates a filter object, based on filter options, to notify when the state changes (logs).

Usage

```
eth_newFilter(from_block = "earliest", to_block = "latest", address,
  topics = -1)
```

Arguments

`from_block`	Integer|Tag - Block number, or the string 'earliest', 'latest' or 'pending'.
`to_block`	Integer|Tag - Block number, or the string 'earliest', 'latest' or 'pending'.
`address`	Address - Contract address or a list of addresses from which logs should originate.
`topics`	Array of Data - Topics are order-dependent. Each topic can also be an array of DATA with 'or' options.

Value

Hash - A filter Id.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHa`
`eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberAndIndex`, `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAndIndex`, `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newPendingTransactionFilter`, `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `eth_uninstallFilter`, `gethr`, `personal_sendTransaction`

Examples

```
eth_newFilter(0, 100, '0xcaf9a0356ddfa779fdbb55c45b22d35673550f30',
list('0x977f31fe2eae427d123315e068c90016b9f8c44b9c8d0818a740f06d2dc10f95',
'0x0000000000000000000000000000000000000000000000000000000000000003'))
eth_newFilter(address = '0x8655bd257db96eb2aca7154f845d6b1d67689219')
```

eth_newPendingTransactionFilter *New pending transaction filter.*

Description

eth_newPendingTransactionFilter creates a filter in the node, to notify when new pending transactions arrive.

Usage

```
eth_newPendingTransactionFilter()
```

Value

Hash - A filter Id.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_newPendingTransactionFilter()
```

eth_protocolVersion *Ethereum protocol version.*

Description

eth_protocolVersion returns the current Ethereum protocol version.

Usage

```
eth_protocolVersion()
```

Value

Integer - Current Ethereum protocol version.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_protocolVersion()
```

---

eth_sendRawTransaction

New raw transaction

Description

`eth_sendRawTransaction` creates new message call transaction or a contract creation for signed transactions.

Usage

```
eth_sendRawTransaction(data)
```

Arguments

`data`	Data - Signed transaction data.

Value

Data - The transaction hash, or the zero hash if the transaction is not yet available.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_sendRawTransaction('0xd46e8dd67c5d32be8d46e8dd67c5d32be8058bb8eb970870f
072445675058bb8eb970870f072145675')
```

---

eth_sendTransaction *New transaction*

Description

eth_sendTransaction creates new message call transaction or a contract creation, if the data field contains code.

Usage

```
eth_sendTransaction(from, data = -1, to = -1, gas = 90000,
  gas_price = -1, value = -1, nonce = -1)
```

Arguments

from	Address - Address the transaction is send from.
data	Data - Compiled code of a contract OR the hash of the invoked method signature and encoded parameters.
to	Address - Address the transaction is send to.
gas	Integer - Gas provided for the transaction execution. It will return unused gas.
gas_price	Integer - Value of the gas for this transaction.
value	Integer - Value sent with the transaction.
nonce	Integer - Value of the nonce. This allows to overwrite your own pending transactions that use the same nonce.

Value

Data - The transaction hash, or the zero hash if the transaction is not yet available.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion, eth_sendRawTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_sendTransaction(from = '0xb117a8bc3ecf2c3f006b89da6826e49b4193977a',
data = '0x8f8a0d190000000000000000000000000000000000000000000000000000000000000000
0000000060000000000000000000000000000000000000000000000000000000000000002')
eth_sendTransaction(from = '0xb117a8bc3ecf2c3f006b89da6826e49b4193977a',
to = '0x84159f53c71ef9e8df944225430da49371cbd67b', value = 10)
```

---

eth_sign *Ethereum specific signature*

Description

eth_sign returns an Ethereum specific signature with Keccak256.

Usage

```
eth_sign(address, message)
```

Arguments

address	Address - Address of the user.
message	String - Message to sign.

Details

By adding a prefix to the message makes the calculated signature recognisable as an Ethereum specific signature. This prevents misuse where a malicious DApp can sign arbitrary data (e.g. transaction) and use the signature to impersonate the victim. Note the address to sign with must be unlocked.

Value

Data - Signature.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_sign('0xb117a8bc3ecf2c3f006b89da6826e49b4193977a', 'hello world')
```

---

eth_submitHashrate *Mining hashrate submission.*

Description

`eth_submitHashrate` submits mining hashrate.

Usage

```
eth_submitHashrate(hashrate, id)
```

Arguments

`hashrate`	Integer - Hashrate to be submitted.
`id`	Data - A random hexadecimal ID identifying the client.

Value

Boolean - returns `true` if the provided solution is valid, otherwise `false`.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_submitHashrate(5200050, '0x1234567890abcdef1234567890abcdef12345678
90abcdef1234567890abcdef')
```

`eth_submitWork` *Proof-of-work submission*

Description

`eth_submitWork` submits a proof-of-work solution.

Usage

```
eth_submitWork(nonce, pow_hash, mix_digest)
```

Arguments

`nonce`	Data - Nonce found.
`pow_hash`	Data - Header's pow-hash.
`mix_digest`	Data - Mix digest.

Value

Boolean - returns `true` if the provided solution is valid, otherwise `false`.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_syncing, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_submitWork('0x0000000000000001', '0x1234567890abcdef1234567890abcde
f1234567890abcdef1234567890abcdef', '0xD1FE5700000000000000000000000000
D1FE5700000000000000000000000000')
```

`eth_syncing` *Sync status.*

Description

`eth_syncing` returns an object with data about the sync status or `false`.

Usage

```
eth_syncing()
```

Value

Object|Boolean - An object with sync status data or `false`, when not syncing.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_uninstallFilter, gethr, personal_sendTransaction

Examples

```
eth_syncing()
```

`eth_uninstallFilter` *Filter removal.*

Description

`eth_uninstallFilter` uninstalls a filter with given id. Should always be called when watch is no longer needed.

Usage

```
eth_uninstallFilter(id)
```

Arguments

`id`	Integer - Filter Id.

Value

Boolean - `true` if the filter can be unistalled, otherwise `false`.

See Also

Other eth functions: `eth_accounts`, `eth_blockNumber`, `eth_call`, `eth_coinbase`, `eth_estimateGas`, `eth_gasPrice`, `eth_getBalance`, `eth_getBlockByHash`, `eth_getBlockByNumber`, `eth_getBlockTransactionCountByHa` `eth_getBlockTransactionCountByNumber`, `eth_getCode`, `eth_getFilterChanges`, `eth_getFilterLogs`, `eth_getLogs`, `eth_getProof`, `eth_getStorageAt`, `eth_getTransactionByBlockHashAndIndex`, `eth_getTransactionByBlockNumberAndIndex`, `eth_getTransactionByHash`, `eth_getTransactionCount`, `eth_getTransactionReceipt`, `eth_getUncleByBlockHashAndIndex`, `eth_getUncleByBlockNumberAndIndex`, `eth_getUncleCountByBlockHash`, `eth_getUncleCountByBlockNumber`, `eth_getWork`, `eth_hashrate`, `eth_mining`, `eth_newBlockFilter`, `eth_newFilter`, `eth_newPendingTransactionFilter`, `eth_protocolVersion`, `eth_sendRawTransaction`, `eth_sendTransaction`, `eth_sign`, `eth_submitHashrate`, `eth_submitWork`, `eth_syncing`, `gethr`, `personal_sendTransaction`

Examples

```
eth_uninstallFilter('0x5f74d97db50bb631ed0efe8b3b3697c8')
```

gethr *gethr: Access to Ethereum-based Blockchains through Geth Nodes Using the R Language*

Description

The gethr package provides full access to the Geth command line interface for running full Ethereum nodes. With gethr it is possible to carry out different tasks such as mine ether, transfer funds, create contacts, explore block history, etc. The package also provides access to all the available APIs. The officially exposed by Ethereum blockchains (`eth`, `shh`, `web3`, `net`) and some provided directly by Geth (`admin`, `debug`, `miner`, `personal`, `txpool`).

See Also

Other admin functions: admin_addPeer, admin_datadir, admin_nodeInfo, admin_peers, admin_setSolc, admin_startRPC, admin_startWS, admin_stopRPC, admin_stopWS

Other debug functions: debug_backtraceAt, debug_blockProfile, debug_cpuProfile, debug_dumpBlock, debug_gcStats, debug_getBlockRlp, debug_goTrace, debug_memStats, debug_seedHash, debug_setBlockProfileRat debug_setHead, debug_stacks, debug_startCPUProfile, debug_startGoTrace, debug_stopCPUProfile, debug_stopGoTrace, debug_traceBlockByHash, debug_traceBlockByNumber, debug_traceBlockFromFile, debug_traceBlock, debug_traceTransaction, debug_verbosity, debug_vmodule, debug_writeBlockProfile, debug_writeMemProfile

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, personal_sendTransaction

Other ether functions: ether.toEther, ether.toFinney, ether.toGether, ether.toGwei, ether.toKether, ether.toKwei, ether.toMether, ether.toMwei, ether.toSzabo, ether.toTether, ether.toWei

Other miner functions: miner_setEtherBase, miner_setExtra, miner_setGasPrice, miner_start, miner_stop

Other net functions: net_listening, net_peerCount, net_version

Other personal functions: personal_ecRecover, personal_importRawKey, personal_listAccounts, personal_lockAccount, personal_newAccount, personal_sign, personal_unlockAccount

Other shh functions: shh_addPrivatekey, shh_addSymKey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPassword, shh_getPrivateKey, shh_getPublicKey, shh_getSymKey, shh_hasKeyPair, shh_hasSymKey, shh_info, shh_markTrustedPeer, shh_newKeyPair, shh_newMessageFilter, shh_newSymKey, shh_post, shh_setMaxMessageSize, shh_setMinPoW, shh_version

Other txpool functions: txpool_content, txpool_inspect, txpool_status

Other web3 functions: web3_clientVersion, web3_sha3

get_network_id *ID of the network.*

Description

`get_network_id` returns the ID of the network that is being used.

Usage

```
get_network_id()
```

Value

String - Network ID.

See Also

Other utils functions: `dec_to_hex`, `get_post`, `get_rpc_address`, `hex_to_dec`, `hex_to_text`, `is.wholenumber`, `process_block`, `process_log`, `process_receipt`, `process_transaction`, `set_network_id`, `set_rpc_address`, `text_to_hex`

Examples

```
get_network_id()
```

get_post *RPC call helper method.*

Description

`get_post` returns the response of the RPC call in the Geth node.

Usage

```
get_post(method, params = list())
```

Arguments

`method`	String - Method to call in the Geth node.
`params`	Array of Strings - Params that are passed to the method.

Value

Object - Response from the Geth node.

See Also

Other utils functions: dec_to_hex, get_network_id, get_rpc_address, hex_to_dec, hex_to_text, is.wholenumber, process_block, process_log, process_receipt, process_transaction, set_network_id, set_rpc_address, text_to_hex

Examples

```
get_post('eth_accounts')
get_post('debug_verbosity', list(3))
get_post('debug_goTrace', list('file.log', 5))
```

---

get_rpc_address *RPC address of the node.*

Description

get_rpc_address returns the RPC address that is being used to connect to the Geth node.

Usage

```
get_rpc_address()
```

Value

String - RPC Address.

See Also

Other utils functions: dec_to_hex, get_network_id, get_post, hex_to_dec, hex_to_text, is.wholenumber, process_block, process_log, process_receipt, process_transaction, set_network_id, set_rpc_address, text_to_hex

Examples

```
get_rpc_address()
```

hex_to_dec *Hexadecimal to decimal conversion.*

Description

hex_to_dec returns the value in decimal.

Usage

```
hex_to_dec(hex)
```

Arguments

hex	Data - Value in hexadecimal.

Value

Integer - Value in decimal.

See Also

Other utils functions: dec_to_hex, get_network_id, get_post, get_rpc_address, hex_to_text, is.wholenumber, process_block, process_log, process_receipt, process_transaction, set_network_id, set_rpc_address, text_to_hex

Examples

```
hex_to_dec('0x28')
hex_to_dec('0xd9190')
```

hex_to_text *Hexadecimal to string conversion.*

Description

hex_to_text returns the plain text.

Usage

```
hex_to_text(msg)
```

Arguments

msg	Data - Value in hexadecimal.

Value

String - Value in plain text.

See Also

Other utils functions: dec_to_hex, get_network_id, get_post, get_rpc_address, hex_to_dec, is.wholenumber, process_block, process_log, process_receipt, process_transaction, set_network_id, set_rpc_address, text_to_hex

Examples

```
hex_to_text('0x68656c6c6f20776f726c64')
```

---

is.wholenumber *Whole numbers identification.*

Description

`is.wholenumber` returns whether a value is a whole number.

Usage

```
is.wholenumber(x, tol = .Machine$double.eps^0.5)
```

Arguments

`x`	Data - Element to check whether it is a whole number.
`tol`	Double - Range to determine if the number is a whole number.

Value

Boolean - `true` if the input is a whole number, `false` otherwise.

See Also

Other utils functions: dec_to_hex, get_network_id, get_post, get_rpc_address, hex_to_dec, hex_to_text, process_block, process_log, process_receipt, process_transaction, set_network_id, set_rpc_address, text_to_hex

Examples

```
is.wholenumber(20)
is.wholenumber(20.3)
is.wholenumber('no')
```

miner_setEtherBase *Etherbase for mining.*

Description

miner_setEtherBase sets the etherbase, where mining rewards will go.

Usage

```
miner_setEtherBase(address)
```

Arguments

address	Address - Address to send the rewards when mining.

Value

Boolean - true if changed, otherwise false.

See Also

Other miner functions: gethr, miner_setExtra, miner_setGasPrice, miner_start, miner_stop

Examples

```
miner_setEtherBase('0xb117a8bc3ecf2c3f006b89da6826e49b4193977a')
```

miner_setExtra *Extra data for mined blocks.*

Description

miner_setExtra sets the extra data a miner can include when miner blocks. This is capped at 32 bytes.

Usage

```
miner_setExtra(data)
```

Arguments

data	String - Data to include in mined blocks.

Value

Boolean - `true` if changed, otherwise `false`.

See Also

Other miner functions: `gethr`, `miner_setEtherBase`, `miner_setGasPrice`, `miner_start`, `miner_stop`

Examples

```
miner_setExtra('My info')
```

`miner_setGasPrice` *Minimal gas price for mining.*

Description

`miner_setGasPrice` sets the minimal accepted gas price when mining transactions. Any transactions that are below this limit are excluded from the mining process.

Usage

```
miner_setGasPrice(price)
```

Arguments

`price`	Integer - Value of the gas price.

Value

Boolean - `true` if changed, otherwise `false`.

See Also

Other miner functions: `gethr`, `miner_setEtherBase`, `miner_setExtra`, `miner_start`, `miner_stop`

Examples

```
miner_setGasPrice(10)
```

miner_start *Mining process to be started.*

Description

`miner_start` starts the CPU mining process with the given number of threads.

Usage

```
miner_start(threads = 1)
```

Arguments

`threads`	Integer - Given number of threads used for mining.

See Also

Other miner functions: `gethr`, `miner_setEtherBase`, `miner_setExtra`, `miner_setGasPrice`, `miner_stop`

Examples

```
miner_start()
miner_start(4)
```

miner_stop *Mining process to be stopped.*

Description

`miner_stop` stops the CPU mining operation.

Usage

```
miner_stop()
```

See Also

Other miner functions: `gethr`, `miner_setEtherBase`, `miner_setExtra`, `miner_setGasPrice`, `miner_start`

Examples

```
miner_stop()
```

`net_listening` *Whether client is listening or not.*

Description

`net_listening` returns `true` if client is actively listening for network connections.

Usage

```
net_listening()
```

Value

Boolean - `true` when listening, otherwise `false`.

See Also

Other net functions: `gethr`, `net_peerCount`, `net_version`

Examples

```
net_listening()
```

`net_peerCount` *Number of peers connected.*

Description

`net_peerCount` returns the number of peers currently connected to the client.

Usage

```
net_peerCount()
```

Value

Integer - Number of connected peers.

See Also

Other net functions: gethr, net_listening, net_version

Examples

```
net_peerCount()
```

---

net_version *Current network id.*

Description

net_version returns the current network id.

Usage

```
net_version()
```

Value

String - Current network id.

See Also

Other net functions: gethr, net_listening, net_peerCount

Examples

```
net_version()
```

personal_ecRecover *Signatory address.*

Description

personal_ecRecover returns the address associated with the private key that was used to calculate a signature.

Usage

```
personal_ecRecover(message, signature)
```

Arguments

message	String - Message that has been signed.
signature	Data - Signature of the message that has been signed.

Value

Address - Address of the account that has signed a message.

See Also

Other personal functions: gethr, personal_importRawKey, personal_listAccounts, personal_lockAccount, personal_newAccount, personal_sign, personal_unlockAccount

Examples

```
personal_ecRecover('hello world', '0x1dd3657c91d95f350ab25f17ee7cbcdbccd3f5bc
52976bfd4dd03bd6bc29d2ac23e656bee509ca33b921e0e6b53eb64082be1bb3c69c3a4adccd9
93b1d667f8d1b')
```

personal_importRawKey *New account creation giving the private key.*

Description

personal_importRawKey imports the given unencrypted private key (hex string) into the key store, encrypting it with the passphrase.

Usage

```
personal_importRawKey(keydata, password)
```

Arguments

keydata	Data - Message that has been signed.
password	String - Password of the account.

Value

Address - Address of the new account.

See Also

Other personal functions: gethr, personal_ecRecover, personal_listAccounts, personal_lockAccount, personal_newAccount, personal_sign, personal_unlockAccount

Examples

```
personal_importRawKey('a5e3d0b2bb3011d00a139e5cdc4ae13144962752d6af7916bf2b
d271a240094e', 'password')
```

---

personal_listAccounts *Addresses owned by client.*

Description

personal_lockAccount returns a list of addresses owned by client.

Usage

```
personal_listAccounts()
```

Value

Array of Address - Addresses owned by the client.

See Also

Other personal functions: gethr, personal_ecRecover, personal_importRawKey, personal_lockAccount, personal_newAccount, personal_sign, personal_unlockAccount

Examples

```
personal_listAccounts()
```

personal_lockAccount *Account deletion.*

Description

personal_lockAccount removes the private key with the given address from memory. The account can no longer be used to send transactions.

Usage

```
personal_lockAccount(address)
```

Arguments

address	Address - Address of account to be deleted.

Value

Boolean - true if the account has been deleted.

See Also

Other personal functions: gethr, personal_ecRecover, personal_importRawKey, personal_listAccounts, personal_newAccount, personal_sign, personal_unlockAccount

Examples

```
personal_lockAccount('0xf1b76d9a65b532dbdc3899dee6e117b52c85a536')
```

personal_newAccount *New account creation.*

Description

personal_newAccount generates a new private key and stores it in the key store directory. The key file is encrypted with the given passphrase.

Usage

```
personal_newAccount(password)
```

Arguments

password	String - Password of the account.

Value

Address - Address of the new account.

See Also

Other personal functions: gethr, personal_ecRecover, personal_importRawKey, personal_listAccounts, personal_lockAccount, personal_sign, personal_unlockAccount

Examples

```
personal_newAccount('password')
```

---

personal_sendTransaction

New transaction

Description

personal_sendTransaction creates new message call transaction or a contract if the data field contains code.

Usage

```
personal_sendTransaction(from, data = -1, to = -1, gas = 90000,
  gas_price = -1, value = -1, nonce = -1, password)
```

Arguments

from	Address - Address the transaction is send from.
data	Data - Compiled code of a contract OR the hash of the invoked method signature and encoded parameters.
to	Address - Address the transaction is send to.
gas	Integer - Gas provided for the transaction execution. It will return unused gas.
gas_price	Integer - Value of the gas for this transaction.
value	Integer - Value sent with the transaction.
nonce	Integer - Value of the nonce. This allows to overwrite your own pending transactions that use the same nonce.
password	String - Password of the account.

Value

Data - The transaction hash, or the zero hash if the transaction is not yet available.

See Also

Other eth functions: eth_accounts, eth_blockNumber, eth_call, eth_coinbase, eth_estimateGas, eth_gasPrice, eth_getBalance, eth_getBlockByHash, eth_getBlockByNumber, eth_getBlockTransactionCountByHa eth_getBlockTransactionCountByNumber, eth_getCode, eth_getFilterChanges, eth_getFilterLogs, eth_getLogs, eth_getProof, eth_getStorageAt, eth_getTransactionByBlockHashAndIndex, eth_getTransactionByBlockNumberAndIndex, eth_getTransactionByHash, eth_getTransactionCount, eth_getTransactionReceipt, eth_getUncleByBlockHashAndIndex, eth_getUncleByBlockNumberAndIndex, eth_getUncleCountByBlockHash, eth_getUncleCountByBlockNumber, eth_getWork, eth_hashrate, eth_mining, eth_newBlockFilter, eth_newFilter, eth_newPendingTransactionFilter, eth_protocolVersion, eth_sendRawTransaction, eth_sendTransaction, eth_sign, eth_submitHashrate, eth_submitWork, eth_syncing, eth_uninstallFilter, gethr

Examples

```
personal_sendTransaction(from = '0xb117a8bc3ecf2c3f006b89da6826e49b4193977a'
, data = '0x8f8a0d190000000000000000000000000000000000000000000000000000000000000000
0000000060000000000000000000000000000000000000000000000000000000000000002',
password = 'password')
personal_sendTransaction(from = '0xb117a8bc3ecf2c3f006b89da6826e49b4193977a'
, to = '0x84159f53c71ef9e8df944225430da49371cbd67b', value = 10, password =
'password')
```

`personal_sign` *Ethereum specific signature*

Description

`personal_sign` returns an Ethereum specific signature with `keccak256`.

Usage

```
personal_sign(address, message, password)
```

Arguments

`address`	Address - Address of the user.
`message`	String - Message to sign.
`password`	String - Password of the account.

Value

Data - Signature.

See Also

Other personal functions: gethr, personal_ecRecover, personal_importRawKey, personal_listAccounts, personal_lockAccount, personal_newAccount, personal_unlockAccount

Examples

```
personal_sign('0xb117a8bc3ecf2c3f006b89da6826e49b4193977a', 'hello world',
'password')
```

---

personal_unlockAccount

Unlocking the key.

Description

personal_unlockAccount unlocks an account to be used for signing and transactions.

Usage

```
personal_unlockAccount(address, password, duration = 3000)
```

Arguments

address	Address - Address to unlock.
password	String - Password of the account.
duration	Integer - Seconds the account will be unlocked.

Value

Boolean - true if the account has been unlocked.

See Also

Other personal functions: gethr, personal_ecRecover, personal_importRawKey, personal_listAccounts, personal_lockAccount, personal_newAccount, personal_sign

Examples

```
personal_unlockAccount('0xb117a8bc3ecf2c3f006b89da6826e49b4193977a',
'password', 60000)
```

process_block *Values of the blocks in plain text or decimal instead of hexadecimal.*

Description

process_block returns the values of the block in plain text or decimal if possible. The values are obtained from the Geth node.

Usage

```
process_block(block)
```

Arguments

block	Objects - Information of the block in hexadecimal.

Value

Object - Information of the block with values in plain text or decimal if possible.

See Also

Other utils functions: dec_to_hex, get_network_id, get_post, get_rpc_address, hex_to_dec, hex_to_text, is.wholenumber, process_log, process_receipt, process_transaction, set_network_id, set_rpc_address, text_to_hex

process_log *Values of the logs in decimal instead of hexadecimal.*

Description

process_block returns the values of the log in decimal if possible. The values are obtained from the Geth node.

Usage

```
process_log(log)
```

Arguments

log	Objects - Information of the log in hexadecimal.

Value

Object - Information of the log with values in decimal if possible.

See Also

Other utils functions: dec_to_hex, get_network_id, get_post, get_rpc_address, hex_to_dec, hex_to_text, is.wholenumber, process_block, process_receipt, process_transaction, set_network_id, set_rpc_address, text_to_hex

---

process_receipt *Values of the receipts in plain text or decimal instead of hexadecimal.*

Description

process_block returns the values of the receipts in plain text or decimal if possible. The values are obtained from the Geth node.

Usage

```
process_receipt(receipt)
```

Arguments

receipt	Objects - Information of the receipt in hexadecimal.

Value

Object - Information of the receipt with values in plain text or decimal if possible.

See Also

Other utils functions: dec_to_hex, get_network_id, get_post, get_rpc_address, hex_to_dec, hex_to_text, is.wholenumber, process_block, process_log, process_transaction, set_network_id, set_rpc_address, text_to_hex

---

process_transaction *Values of the transactions in plain text or decimal instead of hexadecimal.*

Description

process_block returns the values of the transactions in plain text or decimal if possible. The values are obtained from the Geth node.

Usage

```
process_transaction(trans)
```

Arguments

trans	Objects - Information of the transaction in hexadecimal.

Value

Object - Information of the transaction with values in plain text or decimal if possible.

See Also

Other utils functions: dec_to_hex, get_network_id, get_post, get_rpc_address, hex_to_dec, hex_to_text, is.wholenumber, process_block, process_log, process_receipt, set_network_id, set_rpc_address, text_to_hex

---

set_network_id *ID of the network update.*

Description

set_network_id sets the ID of the network that is being used.

Usage

```
set_network_id(id)
```

Arguments

id	String|Integer - ID of the network.

See Also

Other utils functions: dec_to_hex, get_network_id, get_post, get_rpc_address, hex_to_dec, hex_to_text, is.wholenumber, process_block, process_log, process_receipt, process_transaction, set_rpc_address, text_to_hex

Examples

```
set_network_id(7000)
set_network_id('my_network_id')
```

set_rpc_address *Query of the RPC address of the node.*

Description

`set_rpc_address` sets the RPC address that is being used to connect to the Geth node.

Usage

```
set_rpc_address(url, port)
```

Arguments

`url`	String - URL of the Geth node.
`port`	Integer - Port of the Geth node.

See Also

Other utils functions: `dec_to_hex`, `get_network_id`, `get_post`, `get_rpc_address`, `hex_to_dec`, `hex_to_text`, `is.wholenumber`, `process_block`, `process_log`, `process_receipt`, `process_transaction`, `set_network_id`, `text_to_hex`

Examples

```
set_rpc_address('http://153.35.91.1', 8600)
set_rpc_address('http://localhost', 8545)
```

shh_addPrivatekey *Key pair storage.*

Description

`shh_addPrivatekey` stores the key pair, and returns its ID.

Usage

```
shh_addPrivatekey(key)
```

Arguments

`key`	String - Key as HEX bytes.

Value

String - Key pair ID on sucess.

See Also

Other shh functions: gethr, shh_addSymKey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPasswo
shh_getPrivateKey, shh_getPublicKey, shh_getSymKey, shh_hasKeyPair, shh_hasSymKey,
shh_info, shh_markTrustedPeer, shh_newKeyPair, shh_newMessageFilter, shh_newSymKey,
shh_post, shh_setMaxMessageSize, shh_setMinPoW, shh_version

Examples

```
shh_addPrivatekey('0x6d795f707269766174655f6b6579')
```

---

shh_addSymKey *Symmetric key storage.*

Description

shh_addSymKey stores the symmetric key, and returns its ID.

Usage

```
shh_addSymKey(key)
```

Arguments

key	String - Key as HEX bytes.

Value

String - Symmetric key ID on sucess.

See Also

Other shh functions: gethr, shh_addPrivatekey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPa
shh_getPrivateKey, shh_getPublicKey, shh_getSymKey, shh_hasKeyPair, shh_hasSymKey,
shh_info, shh_markTrustedPeer, shh_newKeyPair, shh_newMessageFilter, shh_newSymKey,
shh_post, shh_setMaxMessageSize, shh_setMinPoW, shh_version

Examples

```
shh_addSymKey('0xf6dcf21ed6a17bd78d8c4c63195ab997b3b65ea683705501eae82d32667adc92')
```

shh_deleteKeyPair *Key pair deletion.*

Description

shh_deleteKeyPair deletes the specific key pair if it exists.

Usage

```
shh_deleteKeyPair(id)
```

Arguments

id	String - ID of the key pair.

Value

Boolean - true on success.

See Also

Other shh functions: gethr, shh_addPrivatekey, shh_addSymKey, shh_deleteSymKey, shh_generateSymKeyFromPasswo
shh_getPrivateKey, shh_getPublicKey, shh_getSymKey, shh_hasKeyPair, shh_hasSymKey,
shh_info, shh_markTrustedPeer, shh_newKeyPair, shh_newMessageFilter, shh_newSymKey,
shh_post, shh_setMaxMessageSize, shh_setMinPoW, shh_version

Examples

```
shh_deleteKeyPair('7ade0794bf59b9a4508a3c5d7d3408f910fc575fb2f31845da7611cae6664448')
```

shh_deleteSymKey *Symmetric key deletion.*

Description

shh_deleteSymKey deletes the specific symmetric key if it exists.

Usage

```
shh_deleteSymKey(id)
```

Arguments

id	String - ID of the symmetric key.

Value

Boolean - `true` on success.

See Also

Other shh functions: `gethr`, `shh_addPrivatekey`, `shh_addSymKey`, `shh_deleteKeyPair`, `shh_generateSymKeyFromPassw`
`shh_getPrivateKey`, `shh_getPublicKey`, `shh_getSymKey`, `shh_hasKeyPair`, `shh_hasSymKey`,
`shh_info`, `shh_markTrustedPeer`, `shh_newKeyPair`, `shh_newMessageFilter`, `shh_newSymKey`,
`shh_post`, `shh_setMaxMessageSize`, `shh_setMinPoW`, `shh_version`

Examples

```
shh_deleteSymKey('8d7b2dff569d14308a8e74ca1475dd93ba8dd42a9a74e97638796d5d6c8751ac')
```

---

shh_generateSymKeyFromPassword

Symmetric key generation and storage.

Description

`shh_generateSymKeyFromPassword` stores the symmetric key, and returns its ID.

Usage

```
shh_generateSymKeyFromPassword(password)
```

Arguments

`password`	String - Password used to generate the symmetric key.

Value

String - Symmetric key ID on sucess.

See Also

Other shh functions: `gethr`, `shh_addPrivatekey`, `shh_addSymKey`, `shh_deleteKeyPair`, `shh_deleteSymKey`,
`shh_getPrivateKey`, `shh_getPublicKey`, `shh_getSymKey`, `shh_hasKeyPair`, `shh_hasSymKey`,
`shh_info`, `shh_markTrustedPeer`, `shh_newKeyPair`, `shh_newMessageFilter`, `shh_newSymKey`,
`shh_post`, `shh_setMaxMessageSize`, `shh_setMinPoW`, `shh_version`

Examples

```
shh_generateSymKeyFromPassword('my_password')
```

shh_getPrivateKey *Private key given a key pair ID.*

Description

shh_getPrivateKey returns the private key given an ID.

Usage

```
shh_getPrivateKey(id)
```

Arguments

id	String - ID of the key pair.

Value

String - Private key that corresponds to a key pair ID.

See Also

Other shh functions: gethr, shh_addPrivatekey, shh_addSymKey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPassword, shh_getPublicKey, shh_getSymKey, shh_hasKeyPair, shh_hasSymKey, shh_info, shh_markTrustedPeer, shh_newKeyPair, shh_newMessageFilter, shh_newSymKey, shh_post, shh_setMaxMessageSize, shh_setMinPoW, shh_version

Examples

```
shh_getPrivateKey('7ade0794bf59b9a4508a3c5d7d3408f910fc575fb2f31845da7611cae6664448')
```

shh_getPublicKey *Public key given a key pair ID.*

Description

shh_getPublicKey returns the public key given an ID.

Usage

```
shh_getPublicKey(id)
```

Arguments

id	String - ID of the key pair.

Value

String - Public key that corresponds to a key pair ID.

See Also

Other shh functions: gethr, shh_addPrivatekey, shh_addSymKey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPassword, shh_getPrivateKey, shh_getSymKey, shh_hasKeyPair, shh_hasSymKey, shh_info, shh_markTrustedPeer, shh_newKeyPair, shh_newMessageFilter, shh_newSymKey, shh_post, shh_setMaxMessageSize, shh_setMinPoW, shh_version

Examples

```
shh_getPublicKey('7ade0794bf59b9a4508a3c5d7d3408f910fc575fb2f31845da7611cae6664448')
```

---

shh_getSymKey *Symmetric key given a symmetric key ID.*

Description

shh_getSymKey returns the symmetric key given an ID.

Usage

```
shh_getSymKey(id)
```

Arguments

id	String - ID of the symmetric key.

Value

String - Symmetric key that corresponds to a symmetric key ID.

See Also

Other shh functions: gethr, shh_addPrivatekey, shh_addSymKey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPassword, shh_getPrivateKey, shh_getPublicKey, shh_hasKeyPair, shh_hasSymKey, shh_info, shh_markTrustedPeer, shh_newKeyPair, shh_newMessageFilter, shh_newSymKey, shh_post, shh_setMaxMessageSize, shh_setMinPoW, shh_version

Examples

```
shh_getSymKey('8d7b2dff569d14308a8e74ca1475dd93ba8dd42a9a74e97638796d5d6c8751ac')
```

shh_hasKeyPair *Key pair existence confirmation.*

Description

shh_hasKeyPair checks if the whisper node has a key pair given an ID.

Usage

```
shh_hasKeyPair(id)
```

Arguments

id	String - ID of the key pair.

Value

Boolean - true if the key pair exists, otherwise false.

See Also

Other shh functions: gethr, shh_addPrivatekey, shh_addSymKey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPassword, shh_getPrivateKey, shh_getPublicKey, shh_getSymKey, shh_hasSymKey, shh_info, shh_markTrustedPeer, shh_newKeyPair, shh_newMessageFilter, shh_newSymKey, shh_post, shh_setMaxMessageSize, shh_setMinPoW, shh_version

Examples

```
shh_hasKeyPair('7ade0794bf59b9a4508a3c5d7d3408f910fc575fb2f31845da7611cae6664448')
```

shh_hasSymKey *Symmetric key existence confirmation.*

Description

shh_hasSymKey checks if the whisper node has a symmetric key given an ID.

Usage

```
shh_hasSymKey(id)
```

Arguments

id	String - ID of the symmetric key.

Value

Boolean - `true` if the symmetric key exists, otherwise `false`.

See Also

Other shh functions: gethr, shh_addPrivatekey, shh_addSymKey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPassword, shh_getPrivateKey, shh_getPublicKey, shh_getSymKey, shh_hasKeyPair, shh_info, shh_markTrustedPeer, shh_newKeyPair, shh_newMessageFilter, shh_newSymKey, shh_post, shh_setMaxMessageSize, shh_setMinPoW, shh_version

Examples

```
shh_hasSymKey('4c5e37602b120904c592181e75917949c07b0d2f3111d41bb554b604e532c197')
```

---

shh_info *Information about the whisper.*

Description

`shh_info` returns diagnostic information about the whisper node.

Usage

```
shh_info()
```

Value

Object - Diagnostic information about current minimum PoW requirement, message size limit in bytes, memory size of the messages and number of current messages.

See Also

Other shh functions: gethr, shh_addPrivatekey, shh_addSymKey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPassword, shh_getPrivateKey, shh_getPublicKey, shh_getSymKey, shh_hasKeyPair, shh_hasSymKey, shh_markTrustedPeer, shh_newKeyPair, shh_newMessageFilter, shh_newSymKey, shh_post, shh_setMaxMessageSize, shh_setMinPoW, shh_version

Examples

```
shh_info()
```

shh_markTrustedPeer *Indication of trusted peers.*

Description

shh_markTrustedPeer marks specific peer trusted, which will allow it to send historic (expired) messages.

Usage

```
shh_markTrustedPeer(enode)
```

Arguments

enode	String - Enode of the trusted peer.

Value

Boolean - true on success.

See Also

Other shh functions: gethr, shh_addPrivatekey, shh_addSymKey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPassword, shh_getPrivateKey, shh_getPublicKey, shh_getSymKey, shh_hasKeyPair, shh_hasSymKey, shh_info, shh_newKeyPair, shh_newMessageFilter, shh_newSymKey, shh_post, shh_setMaxMessageSize, shh_setMinPoW, shh_version

Examples

```
shh_markTrustedPeer('enode://c1a07558238c0b31657450dd34a558752d63150ce334f3
e99b9187262b612f48a713a083cd1601bfe3bba761a908264320885633fa81d6d6ca0ef7a6e
84a2bcd@127.0.0.1:30301')
```

shh_newKeyPair *Key pair creation and storage.*

Description

shh_newKeyPair generates a new public and private key pair for message decryption and encryption.

Usage

```
shh_newKeyPair()
```

Value

String - Key pair ID on sucess.

See Also

Other shh functions: gethr, shh_addPrivatekey, shh_addSymKey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPassword, shh_getPrivateKey, shh_getPublicKey, shh_getSymKey, shh_hasKeyPair, shh_hasSymKey, shh_info, shh_markTrustedPeer, shh_newMessageFilter, shh_newSymKey, shh_post, shh_setMaxMessageSize, shh_setMinPoW, shh_version

Examples

```
shh_newKeyPair()
```

---

shh_newMessageFilter *New filter in the node.*

Description

shh_newMessageFilter create a new filter within the node. This filter can be used to poll for new messages that match the set of criteria.

Usage

```
shh_newMessageFilter(symKeyID = NULL, privateKeyID = NULL,
  sig = NULL, minPow = NULL, topics = NULL, allowP2P = NULL)
```

Arguments

symKeyID	String - ID of the symmetric key for message decryption.
privateKeyID	String - ID of the key pair for message decryption.
sig	String - Public key of the signature.
minPow	Integer - Minimal PoW requirement for incomming messages.
topics	Array of Data - Possible topics (or partial topics).
allowP2P	Boolean - Indicates if this filter allows processing of direct peer-to-peer messages (which are not to be forwarded any further, because they might be expired).

Value

String - Filter identifier.

See Also

Other shh functions: gethr, shh_addPrivatekey, shh_addSymKey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPassword, shh_getPrivateKey, shh_getPublicKey, shh_getSymKey, shh_hasKeyPair, shh_hasSymKey, shh_info, shh_markTrustedPeer, shh_newKeyPair, shh_newSymKey, shh_post, shh_setMaxMessageSize, shh_setMinPoW, shh_version

Examples

```
shh_newMessageFilter(symKeyID = '8e975ab6e0427396d3afb748678a09b036ebe389
c1c5a6d39560adabdfdb08ca')
shh_newMessageFilter(privateKeyID = '3794e3b08a2962b066e19869283974dc6b39
6cfc87cdcd69a2d269f5f1bab3cb', minPow = 0.3, allowP2P = TRUE)
```

---

shh_newSymKey *Symmetric key creation and storage.*

Description

shh_newSymKey generates a random symmetric key and stores it under an ID, which is then returned. It can be used encrypting and decrypting messages where the key is known to both parties.

Usage

```
shh_newSymKey()
```

Value

String - Symmetric key ID on sucess.

See Also

Other shh functions: gethr, shh_addPrivatekey, shh_addSymKey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPassword, shh_getPrivateKey, shh_getPublicKey, shh_getSymKey, shh_hasKeyPair, shh_hasSymKey, shh_info, shh_markTrustedPeer, shh_newKeyPair, shh_newMessageFilter, shh_post, shh_setMaxMessageSize, shh_setMinPoW, shh_version

Examples

```
shh_newSymKey()
```

shh_post *New whisper message.*

Description

shh_post creates a whisper message and injects it into the network for distribution.

Usage

```
shh_post(symKeyID = NULL, pubKey = NULL, sig = NULL, ttl,
  topic = NULL, payload, padding = NULL, powTime, powTarget,
  targetPeer = NULL)
```

Arguments

symKeyID	String - ID of the symmetric key for message encryption.
pubKey	String - Public key for message encryption.
sig	String - ID of the signing key.
ttl	Integer - Time-to-live in seconds.
topic	String - Message topic (mandatory when key is symmetric).
payload	String - Payload to be encrypted.
padding	String - Optional padding (byte array of arbitrary length).
powTime	Integer - Maximal time in seconds to be spent on proof of work.
powTarget	Integer - Minimal PoW target required for this message.
targetPeer	String - Optional peer ID (for peer-to-peer message only).

Value

Boolean - true if the message was send, otherwise false.

See Also

Other shh functions: gethr, shh_addPrivatekey, shh_addSymKey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPassword, shh_getPrivateKey, shh_getPublicKey, shh_getSymKey, shh_hasKeyPair, shh_hasSymKey, shh_info, shh_markTrustedPeer, shh_newKeyPair, shh_newMessageFilter, shh_newSymKey, shh_setMaxMessageSize, shh_setMinPoW, shh_version

Examples

```
shh_post(symKeyID = '7bc23b46b61e8223ef49241bd23b94921fb1e2dd8fd7bcb58df41
59e1f4d3759', ttl = 7, topic = '0x07678231', payload = '0x68656c6c6f',
powTime = 2, powTarget = 3)
shh_post(pubKey = '0x0425670405b102c0ce487cefae7aa2bfd7b474b76bc8433499bec
777bb15d6d8a6b95e3001d16de259bf3170ec4cff38f00321eedc8a808a2f2e67bec6b254a
```

```
1b1', ttl = 7, payload = '0x68656c6c6f', powTime = 2, powTarget = 3)
```

---

shh_setMaxMessageSize *Maximal message size allowed by this node.*

Description

shh_setMaxMessageSize sets the maximal message size allowed by this node. Incoming and outgoing messages with a larger size will be rejected. Whisper message size can never exceed the limit imposed by the underlying P2P protocol (10 Mb).

Usage

```
shh_setMaxMessageSize(size)
```

Arguments

size	Integer - Message size in bytes

Value

Boolean - true on success.

See Also

Other shh functions: gethr, shh_addPrivatekey, shh_addSymKey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPassword, shh_getPrivateKey, shh_getPublicKey, shh_getSymKey, shh_hasKeyPair, shh_hasSymKey, shh_info, shh_markTrustedPeer, shh_newKeyPair, shh_newMessageFilter, shh_newSymKey, shh_post, shh_setMinPoW, shh_version

Examples

```
shh_setMaxMessageSize(1048576)
```

shh_setMinPoW *Minimal Pow.*

Description

shh_setMinPoW sets the minimal PoW required by this node.

Usage

```
shh_setMinPoW(pow)
```

Arguments

pow	Integer - The new PoW requirement.

Value

Boolean - true on success.

See Also

Other shh functions: gethr, shh_addPrivatekey, shh_addSymKey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPassword, shh_getPrivateKey, shh_getPublicKey, shh_getSymKey, shh_hasKeyPair, shh_hasSymKey, shh_info, shh_markTrustedPeer, shh_newKeyPair, shh_newMessageFilter, shh_newSymKey, shh_post, shh_setMaxMessageSize, shh_version

Examples

```
shh_setMinPoW(0.2)
```

shh_version *Current whisper protocol.*

Description

shh_version returns the current whisper protocol version.

Usage

```
shh_version()
```

Value

String - Current whisper protocol version.

See Also

Other shh functions: gethr, shh_addPrivatekey, shh_addSymKey, shh_deleteKeyPair, shh_deleteSymKey, shh_generateSymKeyFromPassword, shh_getPrivateKey, shh_getPublicKey, shh_getSymKey, shh_hasKeyPair, shh_hasSymKey, shh_info, shh_markTrustedPeer, shh_newKeyPair, shh_newMessageFilter, shh_newSymKey, shh_post, shh_setMaxMessageSize, shh_setMinPoW

Examples

```
shh_version()
```

---

text_to_hex *String to hexadecimal conversion.*

Description

text_to_hex returns the hexadecimal.

Usage

```
text_to_hex(text)
```

Arguments

text	String - Value in plain text.

Value

Data - Value in hexadecimal.

See Also

Other utils functions: dec_to_hex, get_network_id, get_post, get_rpc_address, hex_to_dec, hex_to_text, is.wholenumber, process_block, process_log, process_receipt, process_transaction, set_network_id, set_rpc_address

Examples

```
text_to_hex('hello world')
```

txpool_content *Information about pending and queued transactions.*

Description

`txpool_content` returns a list with the exact details of all the transactions currently pending for inclusion in the next block(s), as well as the ones that are being scheduled for future execution only.

Usage

```
txpool_content()
```

Value

Object - Information about the pending and queued transactions to be inserted in the blockchain.

See Also

Other txpool functions: `gethr`, `txpool_inspect`, `txpool_status`

Examples

```
txpool_content()
```

txpool_inspect *Summary of the information about pending and queued transactions.*

Description

`txpool_inspect` returns a list with a textual summary of all the transactions currently pending for inclusion in the next block(s), as well as the ones that are being scheduled for future execution only. This is a method specifically tailored to developers to quickly see the transactions in the pool and find any potential issues.

Usage

```
txpool_inspect()
```

Value

Object - Summary of the information about the pending and queued transactions to be inserted in the blockchain.

See Also

Other txpool functions: gethr, txpool_content, txpool_status

Examples

```
txpool_inspect()
```

`txpool_status` *Number of pending and queued transactions.*

Description

`txpool_status` returns the number of transactions currently pending for inclusion in the next block(s), as well as the ones that are being scheduled for future execution only.

Usage

```
txpool_status()
```

Value

Integer - Number of pending and queued transactions.

See Also

Other txpool functions: gethr, txpool_content, txpool_inspect

Examples

```
txpool_status()
```

web3_clientVersion *Current client version.*

Description

web3_clientVersion returns the current client version.

Usage

```
web3_clientVersion()
```

Value

String - Current client version.

See Also

Other web3 functions: gethr, web3_sha3

Examples

```
web3_clientVersion()
```

web3_sha3 *Keccak-256 value of the data.*

Description

web3_sha3 returns Keccak-256 (not the standardized SHA3-256) of the given data.

Usage

```
web3_sha3(data)
```

Arguments

data	String - Data to convert into a SHA3 hash.

Value

Data - SHA3 result of the given string.

See Also

Other web3 functions: gethr, web3_clientVersion

Examples

```
web3_sha3('hello')
```

Index