Package ‘rbtc’
September 12, 2018

Title Bitcoin API
Version 0.1-5
Description Implementation of the RPC-JSON API for Bitcoin and utility functions for address creation and content analysis of the blockchain.
Depends R (>= 3.4.0)
License GPL-3
Encoding UTF-8
LazyData true
RoxygenNote 6.1.0
Imports methods, rjson, httr, openssl, gmp
NeedsCompilation no
Author Bernhard Pfaff [aut, cre]
Maintainer Bernhard Pfaff <bernhard@pfaffikus.de>
Repository CRAN
Repository/R-Forge/Project rbtc
Repository/R-Forge/Revision 7
Repository/R-Forge/DateTimeStamp 2018-09-06 18:39:17
Date/Publication 2018-09-12 15:10:07 UTC

R topics documented:

addnode ................................................. 3
ANSRPC-class ......................................... 4
base58CheckDecode ................................. 5
base58CheckEncode .................................. 6
bkfee ............................................... 7
blockattime ......................................... 7
blockstats .......................................... 8
BTCADR-class ....................................... 9
clearbanned ........................................ 9
concatHex .................................................. 10
conrpc ......................................................... 11
CONRPC-class .................................................. 12
containsPoint .................................................. 12
createBtcAdr ................................................... 13
createPrivateKey ............................................. 14
date2int ....................................................... 15
decodeHex ...................................................... 15
decoderawtransaction ....................................... 16
decodescript .................................................... 17
disconnectnode ............................................... 18
ecoperators ..................................................... 19
ecpam ............................................................ 20
ECPARAM-class ............................................... 21
EcparamOrNull-class ......................................... 21
ecpoint .......................................................... 22
ECPOINT-class ............................................... 23
getaddednodeinfo ............................................ 24
getbestblockhash ............................................ 24
getblock ....................................................... 25
getblockchaininfo ........................................... 26
getblockcount .................................................. 27
getblockhash ................................................... 27
getblockheader ................................................ 28
getchaintips .................................................... 29
getchaintxstats ............................................... 30
getconnectioncount ......................................... 31
getdifficulty ................................................... 31
gethelp ........................................................ 32
getinfo .......................................................... 33
getmempoolancestors ....................................... 34
getmempooldescendants ..................................... 35
getmempoolentry ............................................. 36
getmempoolinfo .............................................. 37
getnettotals .................................................... 37
getnetworkinfo ............................................... 38
getpeerinfo ..................................................... 39
getrawmempool ............................................... 40
getrawtransaction .......................................... 41
gettxout ........................................................ 42
gettxoutproof ............................................... 43
gettxoutsetinfo .............................................. 44
getwalletinfo .................................................. 44
hash160 ........................................................ 45
hash256 ........................................................ 46
int2date .......................................................... 47
intMaxDay ...................................................... 48
intMinDay ...................................................... 48
addnode

**Description**

Attempts to add or remove a node from the addnode list. Or try a connection to a node once.

**Usage**

```
addnode(con, node, command = c("add", "remove", "onetry"))
```
ANSRPC-class

Arguments

- **con** object of class CONRPC.
- **node** character the node (see getpeerinfo() for nodes).
- **command** character 'add' to add a node to the list, 'remove' to remove a node from the list, 'onetry' to try a connection to the node once.

Value

A S4-object of class ANSRPC.

Author(s)

Bernhard Pfaff

References


See Also

Other Network RPCs: clearbanned, disconnectnode, getaddednodeinfo, getconnectioncount, getnettotals, getnetworkinfo, getpeerinfo, listbanned, ping, setnetworkactive

ANSRPC-class The ANSRPC class

Description

This class definition is employed to cast the JSON-objects returned by API-calls to bitcoind.

Slots

- **rpcname** character the name of the API.
- **result** ANY the output/result of the API.
- **ecode** NullOrInteger the error code, in case of no error NULL.
- **emessage** NullOrIntegerCharacter the error message, in case of no error NULL.
- **id** character identifier to API-call.

See Also

Other bitcoind functions: CONRPC-class, NullOrCharacter-class, NullOrInteger-class, conrpc, rpcpost, startbtc, stopbtc
Description

This is a modified binary-to-text decoding used for decoding Bitcoin addresses, aka Base58Check. If this is applied to a WIF address and the first and last four bytes are dropped, the result is the corresponding private key.

Usage

base58CheckDecode(x)

Arguments

x character, string in hex format.

Value

list, the decoded elements of the string.

Author(s)

Bernhard Pfaff

References

https://en.bitcoin.it/wiki/Wallet_import_format,
https://en.bitcoin.it/wiki/Address,
https://en.bitcoin.it/wiki/Base58Check_encoding

See Also

Other BtcAddresses: BTCADR-class,PrivKey2PubKey,PrivKey2Wif,PubHash2BtcAdr,PubKey2PubHash,Wif2PrivKey,base58CheckEncode,concatHex,createBtcAdr,createPrivateKey,decodeHex,hash160,hash256,validBtcAdr
base58CheckEncode  Base 58 binary-to-text-encoding

Description

This is a modified binary-to-text encoding used for encoding Bitcoin addresses, aka Base58Check. If this is applied to an extended private key with its trailing check sum, then the result is the Wallet Import Format, (WIF).

Usage

base58CheckEncode(x)

Arguments

x  character, string in hex format.

Value

character, the encoded string.

Author(s)

Bernhard Pfaff

References

https://en.bitcoin.it/wiki/Wallet_import_format,
https://en.bitcoin.it/wiki/Address,
https://en.bitcoin.it/wiki/Base58Check_encoding

See Also

Other BtcAdresses: BTCADR-class, PrivKey2PubKey, PrivKey2Wif, PubHash2BtcAddr, PubKey2PubHash, Wif2PrivKey, base58CheckDecode, concatHex, createBtcAddr, createPrivateKey, decodeHex, hash160, hash256, validBtcAddr
Compute fee in a block

Description

This function returns the fee of the coinbase transaction. Hereby, the mining reward has been deducted. Initially, the mining reward was 50 BTC and is halved every 210,000 blocks.

Usage

bkfee(con, height)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>con</td>
<td>CONRPC, configuration object.</td>
</tr>
<tr>
<td>height</td>
<td>integer, the height of the block.</td>
</tr>
</tbody>
</table>

Value

numeric

Author(s)

Bernhard Pfaff

See Also

Other UtilityFuncs: blockattime, blockstats, date2int, int2date, intMaxDay, intMinDay, intRangeDay, intRangePeriod, timeofblock, txfee, txids, txinids, txstats, utxoage, utxotype, utxovalue

Block height at time

Description

This function returns the block heights closest to a provided date/time (time zone is GMT).

Usage

blockattime(con, targetdate)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>con</td>
<td>CONRPC, configuration object.</td>
</tr>
<tr>
<td>targetdate</td>
<td>POSIXct, the date/time of closest block heights.</td>
</tr>
</tbody>
</table>
Value

data.frame: the heights, the times and the time differences (in minutes) to the provided date/time.

Author(s)

Bernhard Pfaff

See Also

Other UtilityFuncs: bkfee, blockstats, date2int, int2date, intMaxDay, intMinDay, intRangeDay, intRangePeriod, timeofblock, txfee, txids, txinids, txstats, utxoage, utxotype, utxovalue

---

blockstats

Obtaining statistics of a block

Description

This function returns key statistics of a block’s content, such as the time, the count of transactions, and summary statistics of the UTXOs.

Usage

blockstats(con, height, excoinbase = TRUE)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>con</td>
<td>CONRPC, configuration object.</td>
</tr>
<tr>
<td>height</td>
<td>integer, the block’s height.</td>
</tr>
<tr>
<td>excoinbase</td>
<td>logical, whether coinbase transaction should be excluded (default is TRUE).</td>
</tr>
</tbody>
</table>

Value

An object of class data.frame

Author(s)

Bernhard Pfaff

See Also

Other UtilityFuncs: bkfee, blockatime, date2int, int2date, intMaxDay, intMinDay, intRangeDay, intRangePeriod, timeofblock, txfee, txids, txinids, txstats, utxoage, utxotype, utxovalue
BTCADR-class

Description

S4-class for BTC addresses, ordinarily created by a call to createBtcAddr().

Slots

- privkey character, the private key.
- wif character, the WIF.
- pubkey character, the 512-bit public key.
- pubhash character, the hashed public key.
- btcadr character, the BTC address.
- mainnet logical, whether mainnet or testnet.

Author(s)

Bernhard Pfaff

References

https://en.bitcoin.it/wiki/Address

See Also

Other Btc Addresses: PrivKey2PubKey, PrivKey2Wif, PubHash2BtcAddr, PubKey2PubHash, Wif2PrivKey, base58CheckDecode, base58CheckEncode, concatHex, createBtcAddr, createPrivateKey, decodeHex, hash160, hash256, validBtcAddr

clearbanned

RPC-JSON API: clearbanned

Description

Clear all banned IPs.

Usage

clearbanned(con)

Arguments

con object of class CONFPC.
Concatenate two hex strings

This function concatenates two hex strings, provided without the 0x prefix, and returns a list object of the associated integers.

Usage

```
concatHex(hex1, hex2)
```

Arguments

- **hex1**: character, a hex string.
- **hex2**: character, a hex string.

Value

list

Author(s)

Bernhard Pfaff

References

- https://en.bitcoin.it/wiki/Wallet_import_format
- https://en.bitcoin.it/wiki/Address
See Also

Other bitcoind functions: BTCADR-class, PrivKey2PubKey, PrivKey2Wif, PubHash2BtcAdr, PubKey2PubHash, Wif2PrivKey, base58CheckDecode, base58CheckEncode, createBtcAdr, createPrivateKey, decodeHex, hash160, hash256, validBtcAdr

Examples

h1 <- "80"
h2 <- createPrivateKey()
concatHex(h1, h2)
CONRPC-class  

Description  
S4-class for curl connections to RPC-JSON.

Details  
The slots rpcuse and rpcpwd are required in the call to curl. Furthermore, the fully qualified path to bitcoin.conf (slot config) is required for starting and stopping bitcoind as daemon.

See Also  
Other bitcoind functions: ANSRPC-class, NullOrCharacter-class, NullOrInteger-class, conrpc, rpcpost, startbtc, stopbtc

containsPoint  

Description  
Checks whether a point is on a defined elliptic curve.

Usage  
containsPoint(curve, x, y)

## S4 method for signature 'ECPARAM, bigz, bigz'
containsPoint(curve, x, y)

## S4 method for signature 'ECPARAM, integer, integer'
containsPoint(curve, x, y)

## S4 method for signature 'ECPARAM, character, character'
containsPoint(curve, x, y)

Arguments  

curve  
an S4-object of class ECPARAM.

x  
an S4-object of class bigz, the x-coordinate.

y  
an S4-object of class bigz, the y-coordinate.

Value  
logical
**createBtcAdr**

**Author(s)**
Bernhard Pfaff

**References**
https://en.bitcoin.it/wiki/Secp256k1

**See Also**
Other EllipticCurve: ECPARAM-class, ECPPOINT-class, EcparamOrNull-class, eoperators, ecpam, ecpoint, isNull

**Examples**
```r
p <- "0xffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff"
b <- "0x79be667ef9d8aca155a0e6295ce870d07098fcdbb20d8095f2815b16f81798"
a <- "0x79be667ef9d8aca155a0e6295ce870d07098fcdbb20d8095f2815b16f81798"
curve256 <- ecpam(p, a, b)
Gx <- "0x483ada7726a3c4655da4fbbfc0e1108a8fd17c448a68554199c47d08ffbf10d4b8"
Gy <- "0x483ada7726a3c4655da4fbbfc0e1108a8fd17c448a68554199c47d08ffbf10d4b8"
containsPoint(curve256, Gx, Gy)
```

**createBtcAdr**

Create BTC addresses

**Description**
This function creates an object of S4-class BTCADR.

**Usage**
```r
createBtcAdr(privkey, mainnet = TRUE)
```

**Arguments**
- **privkey** character, a private key.
- **mainnet** logical, for which net the keys should belong to.

**Value**
Object of S4-class BTCADR

**Author(s)**
Bernhard Pfaff

**References**
https://en.bitcoin.it/wiki/Address
See Also

Other BtcAddresses: BTCADR-class, PrivKey2PubKey, PrivKey2Wif, PubHash2BtcAdr, PubKey2PubHash, Wif2PrivKey, base58CheckDecode, base58CheckEncode, concatHex, createPrivateKey, decodeHex, hash160, hash256, validBtcAdr

createPrivateKey

Creation of a private key

Description

Returns a random 256-bit private key in hex notation.

Usage

createPrivateKey()

Value

character.

Author(s)

Bernhard Pfaff

References

https://en.bitcoin.it/wiki/Wallet_import_format,
https://en.bitcoin.it/wiki/Address

See Also

Other BtcAddresses: BTCADR-class, PrivKey2PubKey, PrivKey2Wif, PubHash2BtcAdr, PubKey2PubHash, Wif2PrivKey, base58CheckDecode, base58CheckEncode, concatHex, createBtcAdr, decodeHex, hash160, hash256, validBtcAdr

Examples

createPrivateKey()
**date2int**

Convert date/time to integer

**Description**

This function returns the associated integer time for a given date/time object (coercible as POSIXct object).

**Usage**

date2int(x)

**Arguments**

- **x** POSIXct, date/time object.

**Value**

integer

**Author(s)**

Bernhard Pfaff

**See Also**

Other UtilityFuncs: bkfee, blockattime, blockstats, int2date, intMaxDay, intMinDay, intRangeDay, intRangePeriod, timeofblock, txfee, txids, txinids, txstats, utxoage, utxotype, utxovalue

**Examples**

d <- "2017-03-15"
date2int(d)

**decodeHex**

Decoding of a hex string

**Description**

This function converts a hex string, whereby the string must not contain the 0x prefix, to a list object with the associated integers as its elements.

**Usage**

decodeHex(s)
Arguments

s character, the hex string.

Value

list

Author(s)

Bernhard Pfaff

References

https://en.bitcoin.it/wiki/Wallet_import_format,
https://en.bitcoin.it/wiki/Address

See Also

Other BtcAddresses: BTCADR-class,PrivKey2PubKey,PrivKey2Wif,PubHash2BtcAdr,PubKey2PubHash,
Wif2PrivKey,base58CheckDecode,base58CheckEncode,concatHex,createBtcAdr,createPrivateKey,
hash160,hash256,validBtcAdr

Examples

pk <- createPrivateKey()
decodeHex(pk)

---

decoderawtransaction  

RPC-JSOn API: decoderawtransaction

Description

Return a JSON object representing the serialized, hex-encoded transaction.

Usage

decoderawtransaction(con, hexstring)

Arguments

con object of class CONRPC.

hexstring character, the transaction hex string.

Value

A S4-object of class ANSRPC.
The `decodescript` RPC decodes a hex-encoded P2SH redeem script.

**Usage**

```
decodescript(con, redeem)
```

**Arguments**

- `con` object of class `conrpc`.
- `redeem` character, the P2SH.

**Value**

A S4-object of class `ansrpc`.

**Author(s)**

Bernhard Pfaff

**References**


**See Also**

Other RawTransactions RPCs: `getrawtransaction`

Other Blockchain RPCs: `getbestblockhash`, `getblockchaininfo`, `getblockcount`, `getblockhash`, `getblockheader`, `getblock`, `getchaintips`, `getchaintxstats`, `getdifficulty`, `getmempoolancestors`, `getmempooldescendants`, `getmempoolentry`, `getmempoolinfo`, `getrawmempool`, `gettxoutproof`, `gettxoutsetinfo`, `gettxout`, `preciousblock`, `pruneblockchain`, `verifychain`, `verifytxoutproof`
disconnectnode

**RPC-JSON API: disconnectnode**

**Description**

Immediately disconnects from the specified peer node. Strictly one out of address and nodeid can be provided to identify the node.

**Usage**

\[
\text{disconnectnode}(\text{con}, \text{address} = \text{NULL}, \text{nodeid} = \text{NULL})
\]

**Arguments**

- \text{con} \text{object of class CONRPC.}
- \text{address} \text{character the IP address/port of the node.}
- \text{nodeid} \text{character The node ID (see getpeerinfo() for node IDs).}

**Value**

A $4$-object of class ANSRPC.

**Author(s)**

Bernhard Pfaff

**References**


**See Also**

Other Network RPCs: addnode, clearbanned, getaddednodeinfo, getconnectioncount, getnettotals, getnetworkinfo, getpeerinfo, listbanned, ping, setnetworkactive
**Elliptic curve operators**

**Description**

The following operations for EC points are available:

- doubleUp: multiplying a point by itself
- +: point addition
- leftmostBit: highest bit value of an integer
- AND: logical and-operator for two integers
- *: multiplication of an integer scalar with an EC point

**Usage**

```plaintext
doubleUp(ecp)

## S4 method for signature 'ECPOINT'
doubleUp(ecp)

## S4 method for signature 'ECPOINT,ECPOINT'
e1 + e2

leftmostBit(x)

## S4 method for signature 'bigz'
leftmostBit(x)

AND(x, y)

## S4 method for signature 'bigz,bigz'
AND(x, y)

## S4 method for signature 'ECPOINT,bigz'
e1 * e2

## S4 method for signature 'bigz,ECPOINT'
e1 * e2
```

**Arguments**

- `ecp`: point on elliptic curve
- `e1`: point on elliptic curve, or integer
- `e2`: point on elliptic curve, or integer
- `x`: integer
- `y`: integer
ecparam

Author(s)
Bernhard Pfaff

References
https://en.bitcoin.it/wiki/Secp256k1

See Also
Other EllipticCurve: ECPARAM-class, ECPOINT-class, EcparamOrNull-class, containsPoint, ecparam, ecpoint, isNull

ecparam

Creating objects of class ECPARAM

Description
This function returns an object of S4-class ECPARAM, that does contain the parametrization of an elliptic curve.

Usage
ecparam(p, a, b)

Arguments
p integer
a integer
b integer

Value
An object of S4-class ECPARAM

Author(s)
Bernhard Pfaff

References
https://en.bitcoin.it/wiki/Secp256k1

See Also
Other EllipticCurve: ECPARAM-class, ECPOINT-class, EcparamOrNull-class, containsPoint, ecoperators, ecpoint, isNull
Examples

```r
p <- "0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF"
b <- "0x00000000000000000000000000000000000000000000000000000000000000000"
a <- "0x00000000000000000000000000000000000000000000000000000000000000000"
curve256 <- ecparam(p, a, b)
curve256
```

Description

S4-class for elliptic curve parameters. Objects of this class do contain the big integer parameters of elliptic curves. Instances of this class are ordinarily created by a call to `ecparam`.

Slots

- `p bigz`, curve dimension.
- `a bigz`, parameter.
- `b bigz`, parameter.

Author(s)

Bernhard Pfaff

References

https://en.bitcoin.it/wiki/Secp256k1

See Also

Other EllipticCurve: `ECPOINT-class`, `EcparamOrNull-class`, `containsPoint`, `ecoperators`, `ecparam`, `ecpoint`, `isNull`

EcparamOrNull-class

S4 Class Union ECPARAM or NULL

Description

S4-class union of NULL or ECPARAM.

Author(s)

Bernhard Pfaff
Creating objects of class ECPOINT

Description
This function returns an object of S4-class ECPOINT, that does represent a point on an elliptic curve.

Usage
ecpoint(ecparam = NULL, x, y, r = NULL)

Arguments
- ecparam integerECPARAM
- x x-coordinate, to be coercible to bigz.
- y y-coordinate, to be coercible to bigz.
- r the order of the base point.

Value
An object of S4-class ECPOINT

Author(s)
Bernhard Pfaff

References
https://en.bitcoin.it/wiki/Secp256k1

See Also
Other EllipticCurve: ECPARAM-class, ECPOINT-class, containsPoint, ecoperators, ecpam, ecpoint, isNull
Examples

```r
p <- "0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEFC2F"
b <- "0x0000000000000000000000000000000000000000000000000000000000000007"
a <- "0x0000000000000000000000000000000000000000000000000000000000000009"
r <- "0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEFE6AEF"
x <- "0x79BE667EF9DCBBAC55A06295CE870B07029BFCDB2DCE280D645F6B69B706"
y <- "0x483ada7726a3c4655da4fbfc0e1108a8fd17b448a68554199c47d08ff1b1d4b8"
curve256 <- ecp(p, a, b)
ecp <- ecpoint(curve256, x, y, r)
ecp
```

---

**Description**

S4-class for a point on an elliptic curve. Ordinarily, objects are created by calling `ecpoint`.

**Slots**

- `ecparam` `ECPARAM`
- `x` `bigz`
- `y` `bigz`
- `r` `bigz`

**Author(s)**

Bernhard Pfaff

**References**

[https://en.bitcoin.it/wiki/Secp256k1](https://en.bitcoin.it/wiki/Secp256k1)

**See Also**

Other EllipticCurve: `ECPARAM-class`, `EcparamOrNull-class`, `containsPoint`, `ecoperators`, `ecparam`, `ecpoint`, `isNull`
getaddednodeinfo  

**RPC-JSON API: getaddednodeinfo**

### Description
Returns information about the given added node, or all added nodes (note that onetry addnodes are not listed here)

### Usage
getaddednodeinfo(con, node = NULL)

### Arguments
- **con**
  - object of class `CONRPC`.
- **node**
  - character the node (see `getpeerinfo() for nodes`).

### Value
A S4-object of class `ANSRPC`.

### Author(s)
Bernhard Pfaff

### References

### See Also
- Other Network RPCs: `addnode`, `clearbanned`, `disconnectnode`, `getconnectioncount`, `getnettotals`, `getnetworkinfo`, `getpeerinfo`, `listbanned`, `ping`, `setnetworkactive`

getbestblockhash  

**RPC-JSON API: getbestblockhash**

### Description
Returns the hash of the best (tip) block in the longest blockchain.

### Usage
getbestblockhash(con)
getblock

**Arguments**
- `con` object of class CONRPC.

**Value**
A S4-object of class ANSRPC.

**Author(s)**
Bernhard Pfaff

**References**

**See Also**
Other Blockchain RPCs: decodescript, getblockchaininfo, getblockcount, getblockhash, getblockheader, getblock, getchaintips, getchaintxstats, getdifficulty, getmempoolancestors, getmempooldescendants, getmempoolentry, getmempoolinfo, getrawmempool, gettxoutproof, gettxoutsetinfo, gettxout, preciousblock, pruneblockchain, verifychain, verifytxoutproof

---

**getblock**

**RPC-JSON API: getblock**

**Description**
Returns information of a block hash. The returned level of details depends on the argument verbosity.

**Usage**
getblock(con, blockhash, verbosity = c("l1", "l0", "l2"))

**Arguments**
- `con` object of class CONRPC.
- `blockhash` character, the block hash.
- `verbosity` character, level of returned details.

**Value**
A S4-object of class ANSRPC.

Details If verbosity is 'l0', returns a string that is serialized, hex-encoded data for block 'hash'. If verbosity is 'l1' (the default), returns an object with information about block <hash>. If verbosity is 'l2', returns an object with information about block <hash> and information about each transaction.
getblockchaininfo

**Author(s)**
Bernhard Pfaff

**References**

**See Also**
Other Blockchain RPCs: descript, getbestblockhash, getblockchaininfo, getblockcount, getblockhash, getblockheader, getchaintips, getchaintxstats, getdifficulty, getmempoolancestors, getmempooldescendants, getmempoolentry, getmempoolinfo, getrawmempool, gettxoutproof, gettxoutsetinfo, gettxout, preciousblock, pruneblockchain, verifychain, verifytxoutproof

---

**Description**
Returns an object containing various state info regarding blockchain processing.

**Usage**
getblockchaininfo(con)

**Arguments**
con object of class CONRPC.

**Value**
A S4-object of class ANSRPC.

**Author(s)**
Bernhard Pfaff

**References**

**See Also**
Other Blockchain RPCs: descript, getbestblockhash, getblockcount, getblockhash, getblockheader, getblock, getchaintips, getchaintxstats, getdifficulty, getmempoolancestors, getmempooldescendants, getmempoolentry, getmempoolinfo, getrawmempool, gettxoutproof, gettxoutsetinfo, gettxout, preciousblock, pruneblockchain, verifychain, verifytxoutproof
getblockcount

**Description**

Returns the number of blocks in the longest blockchain.

**Usage**

```r
getblockcount(con)
```

**Arguments**

- `con` object of class `CONRPC`.

**Value**

A S4-object of class `ANSRPC`.

**Author(s)**

Bernhard Pfaff

**References**


**See Also**

Other Blockchain RPCs: `decodescript`, `getbestblockhash`, `getblockchaininfo`, `getblockhash`, `getblockheader`, `getblock`, `getchainstats`, `getdifficulty`, `getmempoolancestors`, `getmempooldescendants`, `getmempoolentry`, `getmempoolinfo`, `getrawmempool`, `gettxoutproof`, `getxoutsetinfo`, `getxout`, `preciousblock`, `pruneblockchain`, `verifychain`, `verifytxoutproof`
**getblockheader**

**Arguments**

- **con**  
  object of class `CONRPC`.
- **height**  
  integer the height index.

**Value**

A S4-object of class `ANSRPC`.

**Author(s)**

Bernhard Pfaff

**References**


**See Also**

Other Blockchain RPCs: decodecript, getbestblockhash, getblockchaininfo, getblockcount,  
getblockheader, getblock, getchaintips, getchaintxstats, getdifficulty, getmempoolancestors,  
getmempoolancestords, getmempoolentry, getmempoolinfo, getrawmempool, gettxoutproof,  
getxoutsetinfo, getxout, preciousblock, pruneblockchain, verifychain, verifytxoutproof

---

**Description**

Returns the block header for a given hash string.

**Usage**

```
getblockheader(con, hash, verbose = TRUE)
```

**Arguments**

- **con**  
  object of class `CONRPC`.
- **hash**  
  character the block hash.
- **verbose**  
  logical TRUE for a json object, FALSE for the hex encoded data.

**Value**

A S4-object of class `ANSRPC`.
**getchaintips**

**Details**

If verbose is false, returns a string that is serialized, hex-encoded data for blockheader 'hash'. If verbose is true, returns an Object with information about blockheader <hash>.

**Author(s)**

Bernhard Pfaff

**References**


**See Also**

Other Blockchain RPCs: decodescript, getbestblockhash, getblockchaininfo, getblockcount, getblockhash, getblock, getchaintips, getchaintxstats, getdifficulty, getmempoolancestors, getmempooldescendants, getmempoolentry, getmempoolinfo, getrawmempool, gettxoutproof, getxoutsetinfo, getxout, preciousblock, pruneblockchain, verifychain, verifytxoutproof

---

**getchaintips**  
*RPC-JSON API: getchaintips*

**Description**

Return information about all known tips in the block tree, including the main chain as well as orphaned branches.

**Usage**

getchaintips(con)

**Arguments**

*con*  
oBJECT OF CLASS CONRPC.

**Value**

A S4-object of class ANSRPC.

**Author(s)**

Bernhard Pfaff

**References**

getchaintxstats

See Also

Other Blockchain RPCs: decodescript, getbestblockhash, getblockchaininfo, getblockcount, getblockhash, getblockheader, getblock, getchaintxstats, getdifficulty, getmempoolancestors, getmempoolancestors, getmempooldescendants, getmempoolentry, getmempoolinfo, getrawmempool, gettxoutproof, gettxoutsetinfo, gettxout, preciousblock, pruneblockchain, verifychain, verifytxoutproof

---

getchaintxstats  
**RPC-JSON API:** getchaintxstats

---

**Description**

Compute statistics about the total number and rate of transactions in the chain.

**Usage**

getchaintxstats(con, nblocks = NULL, blockhash = NULL)

**Arguments**

- **con**  
  object of class CONRPC.

- **nblocks**  
  integer optional, size of the window in number of blocks (default: one month).

- **blockhash**  
  character optional, the hash of the block that ends the window.

**Value**

A S4-object of class ANSRPC.

**Author(s)**

Bernhard Pfaff

**References**


**See Also**

Other Blockchain RPCs: decodescript, getbestblockhash, getblockchaininfo, getblockcount, getblockhash, getblockheader, getblock, getchaintxstats, getdifficulty, getmempoolancestors, getmempoolancestors, getmempoolancestors, getmempooldescendants, getmempoolentry, getmempoolinfo, getrawmempool, gettxoutproof, gettxoutsetinfo, gettxout, preciousblock, pruneblockchain, verifychain, verifytxoutproof
getconnectioncount

**Description**

Returns the number of connections to other nodes.

**Usage**

```plaintext
getconnectioncount(con)
```

**Arguments**

- `con`: object of class `CONRPC`.

**Value**

A S4-object of class `ANSRPC`.

**Author(s)**

Bernhard Pfaff

**References**


**See Also**

Other Network RPCs: `addnode`, `clearbanned`, `disconnectnode`, `getaddednodeinfo`, `getnettotals`, `getnetworkinfo`, `getpeerinfo`, `listbanned`, `ping`, `setnetworkactive`

---

getdifficulty

**Description**

Returns the proof-of-work difficulty as a multiple of the minimum difficulty.

**Usage**

```plaintext
getdifficulty(con)
```

**Arguments**

- `con`: object of class `CONRPC`. 
Value
A S4-object of class ANSRPC.

Author(s)
Bernhard Pfaff

References

See Also
Other Blockchain RPCs: decodescript, getbestblockhash, getblockchaininfo, getblockcount, getblockhash, getblockheader, getblock, getchaintips, getchaintxstats, getmempoolancestors, getmempooldescendants, getmempoolentry, getmempoolinfo, getrawmempool, gettxoutproof, gettxoutsetinfo, gettxout, preciousblock, pruneblockchain, verifychain, verifytxoutproof

Description
Returning information about RPC functions.

Usage
gethelp(con, rpc = "")

Arguments
con        object of class CONRPC.
 rpc        character, name of RPC function.

Value
A S4-object of class ANSRPC.

Author(s)
Bernhard Pfaff

References
getinfo

See Also

Other Control RPCs: getinfo, getwalletinfo

---

getinfo  \(\text{RPC-JS\text{"ON API: getinfo}}\)

Description

Returning information about bitcoin configuration and settings.

Usage

getinfo(con)

Arguments

con \(\text{object of class CONRPC.}\)

Details

WARNING: getinfo is deprecated and will be fully removed in 0.16. Projects should transition to using getblockchaininfo, getnetworkinfo, and getwalletinfo before upgrading to 0.16.

Value

A S4-object of class ANSRPC.

Author(s)

Bernhard Pfaff

References


See Also

Other Control RPCs: gethelp, getwalletinfo
getmempoolancestors  

RPC-JSON API: getmempoolancestors

Description

If txid is in the mempool, returns all in-mempool ancestors.

Usage

getmempoolancestors(con, txid, verbose = FALSE)

Arguments

- con: object of class CONRPC.
- txid: character, the transaction id (must be in mempool).
- verbose: logical, TRUE for a json object, FALSE for array of transaction ids (default).

Value

A S4-object of class ANSRPC.

Author(s)

Bernhard Pfaff

References


See Also

Other Blockchain RPCs: decodescript, getbestblockhash, getblockchaininfo, getblockcount, getblockhash, getblockheader, getblock, getchaintips, getchaintxstats, getdifficulty, getmempooldescendants, getmempoolentry, getmempoolinfo, getrawmempool, gettxoutproof, gettxoutsetinfo, gettxout, preciousblock, pruneblockchain, verifychain, verifytxoutproof
**Description**

If txid is in the mempool, returns all in-mempool descendants.

**Usage**

```r
getmempooldescendants(con, txid, verbose = FALSE)
```

**Arguments**

- `con` object of class `CON_RPC`.
- `txid` character, the transaction id (must be in mempool).
- `verbose` logical, TRUE for a json object, FALSE for array of transaction ids (default).

**Value**

A S4-object of class ANSRPC.

**Author(s)**

Bernhard Pfaff

**References**


**See Also**

Other Blockchain RPCs: decodeScript, getbestblockhash, getblockchaininfo, getblockcount, getblockhash, getblockheader, getblock, getchaintips, getchaintxstats, getdifficulty, getmempooldescendants, getmempooldentry, getmempoolinfo, getrawmempool, gettxoutproof, gettxoutsetinfo, gettxout, preciousblock, pruneblockchain, verifychain, verifytxoutproof
getmempoolentry  

**RPC-JSON API:** getmempoolentry

---

**Description**

Returns mempool data for given transaction.

**Usage**

`getmempoolentry(con, txid)`

**Arguments**

- `con` object of class CONRPC.
- `txid` character, the transaction id (must be in mempool).

**Value**

A S4-object of class ANSRPC.

**Author(s)**

Bernhard Pfaff

**References**


**See Also**

Other Blockchain RPCs: decodescript, getbestblockhash, getblockchaininfo, getblockcount, getblockhash, getblockheader, getblock, getchaintips, getchaintxstats, getdifficulty, getmempoolancestors, getmempooldescendants, getmempoolinfo, getrawmempool, gettxoutproof, gettxoutsetinfo, gettxout, preciousblock, pruneblockchain, verifychain, verifytxoutproof
getmempoolinfo

RPC-JSON API: getmempoolinfo

Description
Returns details on the active state of the TX memory pool.

Usage
getmempoolinfo(con)

Arguments
con object of class CONRPC.

Value
A S4-object of class ANSRPC.

Author(s)
Bernhard Pfaff

References

See Also
Other Blockchain RPCs: decodescript, getbestblockhash, getblockchaininfo, getblockcount, getblockhash, getblockheader, getblock, getchaintips, getchaintxstats, getdifficulty, getmempoolancestors, getmempooldescendants, getmempoolentry, getrawmempool, gettxoutproof, getxoutsetinfo, gettxout, preciousblock, pruneblockchain, verifychain, verifytxoutproof

getnettotals

RPC-JSON API: getnettotals

Description
Returns information about network traffic, including bytes in, bytes out, and current time.

Usage
getnettotals(con)
getnetworkinfo

Arguments

   con  object of class CONRPC.

Value

   A S4-object of class ANSRPC.

Author(s)

   Bernhard Pfaff

References


See Also

   Other Network RPCs: addnode, clearbanned, disconnectnode, getaddednodeinfo, getconnectioncount, getnetworkinfo, getpeerinfo, listbanned, ping, setnetworkactive

getnetworkinfo  RPC-JSON API: getnetworkinfo

Description

   Returns an object containing various state info regarding P2P networking.

Usage

   getnetworkinfo(con)

Arguments

   con  object of class CONRPC.

Value

   A S4-object of class ANSRPC.

Author(s)

   Bernhard Pfaff

References

getpeerinfo

See Also

Other Network RPCs: addnode, clearbanned, disconnectnode, getaddednodeinfo, getconnectioncount, getnettotals, getpeerinfo, listbanned, ping, setnetworkactive

---

getpeerinfo  

**RPC-JSON API:** getpeerinfo

**Description**

Returns data about each connected network node as a json array of objects.

**Usage**

`getpeerinfo(con)`

**Arguments**

`con`  
object of class CONRPC.

**Value**

A S4-object of class ANSRPC.

**Author(s)**

Bernhard Pfaff

**References**


**See Also**

Other Network RPCs: addnode, clearbanned, disconnectnode, getaddednodeinfo, getconnectioncount, getnettotals, getnetworkinfo, listbanned, ping, setnetworkactive
getrawmempool

RPC-JSON API: getrawmempool

Description

Returns all transaction ids in memory pool as a json array of string transaction ids. Hint: use getmempoolentry to fetch a specific transaction from the mempool.

Usage

getrawmempool(con, verbose = TRUE)

Arguments

con  object of class CONRPC.
verbose  logical, TRUE for a json object, FALSE for array of transaction ids

Value

A S4-object of class ANSRPC.

Author(s)

Bernhard Pfaff

References


See Also

Other Blockchain RPCs: decodescript, getbestblockhash, getblockchaininfo, getblockcount, getblockhash, getblockheader, getblock, getchaintips, getchaintxstats, getdifficulty, getmempoolancestors, getmempooldescendants, getmempoolentry, getmempoolinfo, gettxoutproof, gettxoutsetinfo, gettxout, preciousblock, pruneblockchain, verifychain, verifytxoutproof
getrawtransaction  

**RPC-JSON API: getrawtransaction**

**Description**

Returns the raw transaction data.

**Usage**

getrawtransaction(con, txid, verbose = FALSE)

**Arguments**

- **con** object of class `CONRPC`.
- **txid** character, the transaction id.
- **verbose** logical, type of output.

**Value**

A S4-object of class A`NSRPC`.

Details By default this function only works for mempool transactions. If the -txindex option is enabled, it also works for blockchain transactions. DEPRECATED: for now, it also works for transactions with unspent outputs. If verbose is 'true', returns an object with information about 'txid'. If verbose is 'false' or omitted, returns a string that is serialized, hex-encoded data for 'txid'.

**Author(s)**

Bernhard Pfaff

**References**


**See Also**

Other RawTransactions RPCs: decoderawtransaction
gettxout  

**RPC-JSON API: gettxout**

---

**Description**

Returns details about an unspent transaction output.

**Usage**

```
gettxout(con, txid, n, incmempool = TRUE)
```

**Arguments**

- `con`: object of class `conrpc`.
- `txid`: character the transaction id.
- `n`: integer vout number.
- `incmempool`: logical whether to include the mempool (default TRUE).

**Details**

Note that an unspent output that is spent in the mempool won’t appear.

**Value**

A S4-object of class `ansrpc`.

**Author(s)**

Bernhard Pfaff

**References**


**See Also**

Other Blockchain RPCs: `decodescript, getbestblockhash, getblockchaininfo, getblockcount, getblockhash, getblockheader, getblock, getchaintips, getchaintxstats, getdifficulty, getmempoolancestors, getmempooldescendants, getmempoolentry, getmempoolinfo, getrawmempool, gettxoutproof, gettxoutsetinfo, preciousblock, pruneblockchain, verifychain, verifytxoutproof`
gettxoutproof  

RPC-JSON API: gettxoutproof

Description

Returns a hex-encoded proof that "txid" was included in a block.

Usage

gettxoutproof(con, txids, blockhash = NULL)

Arguments

- con: object of class CONRPC.
- txids: character a json array of txids to filter.
- blockhash: integer looks for txid in the block with this hash, (optional, default NULL).

Details

NOTE: By default this function only works sometimes. This is when there is an unspent output in the utxo for this transaction. To make it always work, you need to maintain a transaction index, using the -txindex command line option or specify the block in which the transaction is included manually (by blockhash).

Value

A S4-object of class ANSRPC.

Author(s)

Bernhard Pfaff

References


See Also

Other Blockchain RPCs: decodescript, getbestblockhash, getblockchaininfo, getblockcount, getblockhash, getblockheader, getblock, getchaintips, getchaintxstats, getdifficulty, getmempoolancestors, getmempooldescendants, getmempoolentry, getmempoolinfo, getrawmempool, gettxoutsetinfo, gettxout, preciousblock, pruneblockchain, verifychain, verifytxoutproof
gettxoutsetinfo  

**RPC-JSON API: gettxoutsetinfo**

**Description**

Returns statistics about the unspent transaction output set. Note this call may take some time.

**Usage**

gettxoutsetinfo(con)

**Arguments**

- con object of class CONRPC.

**Value**

A S4-object of class ANSRPC.

**Author(s)**

Bernhard Pfaff

**References**


**See Also**

Other Blockchain RPCs: decodescript, getbestblockhash, getblockchaininfo, getblockcount, getblockhash, getblockheader, getblock, getchaintips, getchaintxstats, getdifficulty, getmempoolancestors, getmempooldescendants, getmempoolentry, getmempoolinfo, getrawmempool, gettxoutproof, gettxout, preciousblock, pruneblockchain, verifychain, verifytxoutproof

getwalletinfo  

**RPC-JSON API: getwalletinfo**

**Description**

Returning information about bitcoin wallet.

**Usage**

getwalletinfo(con)
**hash160**

**Arguments**

- `con`: object of class `CONRPC`.

**Value**

- A S4-object of class `ANSRPC`.

**Author(s)**

Bernhard Pfaff

**References**


**See Also**

- Other Control RPCs: `gethelp`, `getinfo`

---

**Description**

This function returns the hash by applying the sha256 hashing first and then to the resulting hash the ripemd160 algorithm.

**Usage**

```r
hash160(d)
```

**Arguments**

- `d`: raw, vector.

**Value**

- character, the value of d hashed with sha256 and ripemd160.

**Author(s)**

Bernhard Pfaff

**References**

- https://en.bitcoin.it/wiki/Address
hash256

See Also

Other BtcAdresses: BTCADR-class, PrivKey2PubKey, PrivKey2Wif, PubHash2BtcAdr, PubKey2PubHash, Wif2PrivKey, base58CheckDecode, base58CheckEncode, concatHex, createBtcAdr, createPrivateKey, decodeHex, hash256, validBtcAdr

| hash256   | BTC hash256 |

Description

This function returns the hash by applying the sha256 hashing algorithm twice to a raw object.

Usage

hash256(d)

Arguments

d raw, vector.

Value

color, the value of d hashed twice.

Author(s)

Bernhard Pfaff

References

https://en.bitcoin.it/wiki/Address

See Also

Other BtcAdresses: BTCADR-class, PrivKey2PubKey, PrivKey2Wif, PubHash2BtcAdr, PubKey2PubHash, Wif2PrivKey, base58CheckDecode, base58CheckEncode, concatHex, createBtcAdr, createPrivateKey, decodeHex, hash160, validBtcAdr
int2date  

Convert time stamp to POSIX

Description

This function returns the associated POSIXct time to the time stamp integer in a block header.

Usage

int2date(x)

Arguments

x  integer, the block header time stamp

Value

An object of class POSIXct, POSIXt

Author(s)

Bernhard Pfaff

References

https://en.bitcoin.it/wiki/Block_timestamp

See Also

Other UtilityFuncs: bkfee, blockattime, blockstats, date2int, intMaxDay, intMinDay, intRangeDay, intRangePeriod, timeofblock, txfee, txids, txinids, txstats, utxoage, utxotype, utxovalue

Examples

ts <- 1532954868
int2date(ts)
intMaxDay

Description

This function returns the associated integer time for the end of a specific day (i.e., 23:59:59 time).

Usage

intMaxDay(x)

Arguments

x POSIXct, date/time object.

Value

integer

Author(s)

Bernhard Pfaff

See Also

Other UtilityFuncs: bkfee, blockattime, blockstats, date2int, int2date, intMinDay, intRangeDay, intRangePeriod, timeofblock, txfee, txids, txinids, txstats, utxoage, utxotype, utxovalue

Examples

d1 <- "2017-03-15"
d1 <- intMaxDay(d1)
d2 <- "2017-03-15 23:59:59"
d2 <- intMaxDay(d2)
identical(d1,d2)

intMinDay

Description

This function returns the associated integer time for the start of a specific day (i.e., 00:00:00 time).

Usage

intMinDay(x)
intRangeDay

Arguments

x POSIXct, date/time object.

Value

integer

Author(s)

Bernhard Pfaff

See Also

Other UtilityFunctions: bkfee, blockattime, blockstats, date2int, int2date, intMaxDay, intRangeDay, intRangePeriod, timeofblock, txfee, txids, txinids, txstats, utxoage, utxotype, utxovalue

Examples

d1 <- "2017-03-15"
d1 <- intMinDay(d1)
d2 <- "2017-03-15 00:00:00"
d2 <- intMinDay(d2)
identical(d1,d2)

---

intRangeDay Integer range within a day

Description

This function returns the associated integer times for the start and end of a specific day.

Usage

intRangeDay(x)

Arguments

x POSIXct, date/time object.

Value

integer

Author(s)

Bernhard Pfaff
**intRangePeriod**

**See Also**

Other UtilityFuncs: *bkfee, blockattime, blockstats, date2int, int2date, intMaxDay, intMinDay, intRangePeriod, timeofblock, txfee, txids, txinids, txstats, utxoage, utxotype, utxovalue*

**Examples**

```r
d1 <- "2017-03-15"
intRangeDay(d1)
intMinDay(d1)
intMaxDay(d1)
```

---

**Description**

This function returns the associated integer times for the start of date \(d_1\) and the end of date \(d_2\).

**Usage**

```r
intRangePeriod(d1, d2)
```

**Arguments**

- **d1** POSIXct, date/time object.
- **d2** POSIXct, date/time object.

**Value**

integer

**Author(s)**

Bernhard Pfaff

**See Also**

Other UtilityFuncs: *bkfee, blockattime, blockstats, date2int, int2date, intMaxDay, intMinDay, intRangeDay, timeofblock, txfee, txids, txinids, txstats, utxoage, utxotype, utxovalue*

**Examples**

```r
d1 <- "2017-03-15"
d2 <- "2017-04-15"
intRangePeriod(d1, d2)
intMinDay(d1)
intMaxDay(d2)
```
isNull

Test for empty EC point

Description
Checks whether an EC point does exist.

Usage
isNull(x)

## S4 method for signature 'ECPOINT'
isNull(x)

Arguments
x object

Value
logical

Author(s)
Bernhard Pfaff

References
https://en.bitcoin.it/wiki/Secp256k1

See Also
Other EllipticCurve: ECPARAM-class, ECPPOINT-class, EcpParamOrNull-class, containsPoint, ecoperators, ecparam, ecpoint

listbanned

RPC-JSON API: listbanned

Description
List all banned IPs/Subnets.

Usage
listbanned(con)
NullOrInteger-class

Arguments

con object of class CONRPC.

Value

A S4-object of class ANSRPC.

Author(s)

Bernhard Pfaff

References


See Also

Other Network RPCs: addnode, clearbanned, disconnectnode, getaddednodeinfo, getconnectioncount, getnettotals, getnetworkinfo, getpeerinfo, ping, setnetworkactive

---

NullOrCharacter-class  S4 Class Union NULL or character

Description

S4-class union of NULL or character.

See Also

Other bitcoind functions: ANSRPC-class, CONRPC-class, NullOrInteger-class, conrpc, rpcpost, startbtc, stopbtc

---

NullOrInteger-class  S4 Class Union NULL or integer

Description

S4-class union of NULL or integer.

See Also

Other bitcoind functions: ANSRPC-class, CONRPC-class, NullOrCharacter-class, conrpc, rpcpost, startbtc, stopbtc
**ping**  

**Description**  
Requests that a ping be sent to all other nodes, to measure ping time. Results provided in getpeerinfo, pingtime and pingwait fields are decimal seconds. Ping command is handled in queue with all other commands, so it measures processing backlog, not just network ping.

**Usage**  

```
ping(con)
```

**Arguments**  

con: object of class `CONRPC`.

**Value**  
A S4-object of class `ANSRPC`.

**Author(s)**  
Bernhard Pfaff

**References**  


**See Also**  
Other Network RPCs: `addnode`, `clearbanned`, `disconnectnode`, `getaddednodeinfo`, `getconnectioncount`, `getnettotals`, `getnetworkinfo`, `getpeerinfo`, `listbanned`, `setnetworkactive`

---

**preciousblock**  

**Description**  
Treats a block as if it were received before others with the same work. A can override the effect of an earlier one. The effects of preciousblock are not retained across restarts.

**Usage**  

```
preciousblock(con, blockhash)
```
Arguments

- **con**: object of class **CONRPC**.
- **blockhash**: character, the hash of the block to mark as precious.

Value

A S4-object of class **ANSRPC**.

Author(s)

Bernhard Pfaff

References


See Also


---

**PrivKey2PubKey**

*Create public key from private key*

Description

This function creates the 512-bit public key corresponding to a private key.

Usage

`PrivKey2PubKey(privkey, mainnet = TRUE)`

Arguments

- **privkey**: character, the private key.
- **mainnet**: logical, whether the WIF should correspond to the mainnet or testnet.

Value

character, the public key.

Author(s)

Bernhard Pfaff
PrivKey2Wif

Description

Returns the corresponding WIF key from a private key

Usage

PrivKey2Wif(privkey, mainnet = TRUE)

Arguments

- privkey: character, a private key.
- mainnet: logical, whether the WIF should correspond to the mainnet or testnet.

Value

character, the WIF key

Author(s)

Bernhard Pfaff

References

https://en.bitcoin.it/wiki/Wallet_import_format,
https://en.bitcoin.it/wiki/Address

See Also

Other BtcAdresses: BTCADR-class, PrivKey2PubKey, PubHash2BtcAdr, PubKey2PubHash, Wif2PrivKey, base58CheckDecode, base58CheckEncode, concatHex, createBtcAdr, createPrivateKey, decodeHex, hash160, hash256, validBtcAdr

Examples

```r
pk <- createPrivateKey()
PrivKey2Wif(pk)
```
**Description**

Pruning of blockchain.

**Usage**

```r
pruneblockchain(con, height)
```

**Arguments**

- `con` object of class `CONRPC`
- `height` integer The block height to prune up to.

**Value**

A S4-object of class `ANSRPC`.

**Details**

May be set to a discrete height, or a unix timestamp to prune blocks whose block time is at least 2 hours older than the provided timestamp.

**Author(s)**

Bernhard Pfaff

**References**


**See Also**

Other Blockchain RPCs: `decodescript, getbestblockhash, getblockchaininfo, getblockcount, getblockhash, getblockheader, getblock, getchaintips, getchaintxstats, getdifficulty, getmempoolancestors, getmempooldescendants, getmempoolentry, getmempoolinfo, getrawmempool, gettxoutproof, gettxoutsetinfo, gettxout, preciousblock, verifychain, verifytxoutproof`
**PubHash2BtcAdr**

Create BTC address from public key hash

**Description**

This function returns the corresponding BTC address from a hashed public key.

**Usage**

```
PubHash2BtcAdr(pubhash)
```

**Arguments**

- **pubhash**: character, the public key hash.

**Value**

character, the BTC address

**Author(s)**

Bernhard Pfaff

**References**

https://en.bitcoin.it/wiki/Address

**See Also**

Other BtcAdresses: BTCADR-class, PrivKey2PubKey, PrivKey2Wif, PubKey2PubHash, Wif2PrivKey, base58CheckDecode, base58CheckEncode, concatHex, createBtcAdr, createPrivateKey, decodeHex, hash160, hash256, validBtcAdr

---

**PubKey2PubHash**

Create public key hash from 512-bit public key

**Description**

This function returns the associated public key hash from a 512-bit public key by using the hash160() function.

**Usage**

```
PubKey2PubHash(pubkey, mainnet = TRUE)
```
Arguments

- **pubkey**: character, the public key.
- **mainnet**: logical, whether the WIF should correspond to the mainnet or testnet.

Value

- character, the hash of a public key

Author(s)

Bernhard Pfaff

References

[https://en.bitcoin.it/wiki/Address](https://en.bitcoin.it/wiki/Address)

See Also

Other BtcAddresses: BTCADR-class, PrivKey2PubKey, PrivKey2Wif, PubHash2BtcAddr, Wif2PrivKey, base58CheckDecode, base58CheckEncode, concatHex, createBtcAddr, createPrivateKey, decodeHex, hash160, hash256, validBtcAddr

---

### rpcpost

*HTTP post of RPC-JS0N*

Description

This function executes an RPC-JS0N post.

Usage

```r
rpcpost(con, api, plist = list())
```

Arguments

- **con**: CONRPC object, returned from conrpc().
- **api**: character the name of the RPC function.
- **plist**: list a named list object of the parameters for api

Value

A list object, coerced JSON answer from RPC.

Author(s)

Bernhard Pfaff
setnetworkactive

**See Also**

Other bitcoind functions: ANSRPC-class, CONRPC-class, NullOrCharacter-class, NullOrInteger-class, conrpc, startbtc, stopbtc

---

setnetworkactive **RPC-JSON API: setnetworkactive**

**Description**

Disable/enable all p2p network activity.

**Usage**

setnetworkactive(con, state = TRUE)

**Arguments**

con object of class CONRPC.

state logical the network state.

**Value**

A S4-object of class ANSRPC.

**Author(s)**

Bernhard Pfaff

**References**


**See Also**

Other Network RPCs: addnode, clearbanned, disconnectnode, getaddednodeinfo, getconnectioncount, getnettotals, getnetworkinfo, getpeerinfo, listbanned, ping
**Description**

Defined show-methods for S4-classes.

**Usage**

```r
## S4 method for signature 'ANSRPC'
show(object)

## S4 method for signature 'BTCADR'
show(object)

## S4 method for signature 'ECPARAM'
show(object)
```

**Arguments**

- `object` a S4-class object.

---

**startbtc**  
*Start bitcoind server process*

**Description**

This function does start the bitcoind-server process. It should only be called when no suitable RPC-JSON process is running.

**Usage**

`startbtc(confbtc)`

**Arguments**

- `confbtc` CONRPC object, returned from `conrpc()`.

**Details**

The process is started by calling `system()`. Hereby, the options: `rpcuser`, `rpcpassword` and `conf` are used in the call to `bitcoind`.

**Value**

NULL
stopbtc

Author(s)
Bernhard Pfaff

See Also
Other bitcoind functions: ANSRPC-class, CONRPC-class, NullOrCharacter-class, NullOrInteger-class, conrpc, rpcpost, stopbtc

stopbtc Stop bitcoind server process

Description
This function stops a running bitcoind process. It calls bitcoin-cli stop via the R function system().

Usage
stopbtc(confbtc)

Arguments
confbtc CONRPC object, returned from conrpc().

Author(s)
Bernhard Pfaff

See Also
Other bitcoind functions: ANSRPC-class, CONRPC-class, NullOrCharacter-class, NullOrInteger-class, conrpc, rpcpost, startbtc

timeofblock Time of a block

Description
This function returns the time of a block in GMT.

Usage
timeofblock(con, height)
txfee

Arguments

- **con**: CONRPC, configuration object.
- **height**: integer, the height of the block.

Value

- POSIXct

Author(s)

Bernhard Pfaff

See Also

Other UtilityFuncs: bkfee, blockattime, blockstats, date2int, int2date, intMaxDay, intMinDay, intRangeDay, intRangePeriod, txfee, txids, txinids, txstats, utxoage, utxotype, utxovalue

```
taxfee(con, txid)  # Compute fee of a transaction
```

Description

This function returns the implicit fee of a transaction, by computing the difference between the sum of its inputs and the sum of its outputs.

Usage

taxfee(con, txid)

Arguments

- **con**: CONRPC, configuration object.
- **txid**: character, the id of the transaction.

Value

numeric

Author(s)

Bernhard Pfaff

See Also

Other UtilityFuncs: bkfee, blockattime, blockstats, date2int, int2date, intMaxDay, intMinDay, intRangeDay, intRangePeriod, timeofblock, txids, txinids, txstats, utxoage, utxotype, utxovalue
txids  Retrieve TX Ids in block

Description
This function retrieves the transaction IDs in a block.

Usage
\[ \text{txids}(\text{con, height, excoinbase = TRUE}) \]

Arguments
- \text{con} \quad \text{CONRPC, configuration object.}
- \text{height} \quad \text{integer, the block's height.}
- \text{excoinbase} \quad \text{logical, whether coinbase transaction should be excluded (default is TRUE).}

Value
character

Author(s)
Bernhard Pfaff

See Also
Other UtilityFuncs: \text{bkfee, blockattime, blockstats, date2int, int2date, intMaxDay, intMinDay, intRangeDay, intRangePeriod, timeofblock, txfee, txinids, txstats, utxoage, utxotype, utxovalue}

txinids  Retrieving the input transaction IDs

Description
This function returns the transaction IDs of the inputs for a given transaction.

Usage
\[ \text{txinids}(\text{con, txid}) \]

Arguments
- \text{con} \quad \text{CONRPC, configuration object.}
- \text{txid} \quad \text{character, the id of the transaction.}
txstats

Value

data.frame, the transaction ID(s) and the position(s) of the previous UTXO(s).

Author(s)

Bernhard Pfaff

See Also

Other UtilityFuncs: bkfee, blockattime, blockstats, date2int, int2date, intMaxDay, intMinDay, intRangeDay, intRangePeriod, timeofblock, txfee, txids, txstats, utxoage, utxotype, utxovalue

---

**txstats**

*Statistics of a transaction*

---

Description

This function returns key statistics/characteristics of a transaction.

Usage

```r
txstats(con, txid)
```

Arguments

- **con**: CONRPC, configuration object.
- **txid**: character, the id of the transaction.

Value

data.frame

Author(s)

Bernhard Pfaff

See Also

Other UtilityFuncs: bkfee, blockattime, blockstats, date2int, int2date, intMaxDay, intMinDay, intRangeDay, intRangePeriod, timeofblock, txfee, txids, txinids, utxoage, utxotype, utxovalue
utxoage  Age of UTXOs

Description
This function returns a difftime object measuring the elapsed time(s) between the UTXO(s) in a transaction and its input(s) (previous UTXO(s)).

Usage
utxoage(con, txid, units = c("auto", "secs", "mins", "hours", "days", "weeks"))

Arguments
- con: CONRPC, configuration object.
- txid: character, the id of the transaction.
- units: character, the time difference units; passed to difftime().

Value
difftime

Author(s)
Bernhard Pfaff

See Also
Other UtilityFuncs: bkfee, blockattime, blockstats, date2int, int2date, intMaxDay, intMinDay, intRangeDay, intRangePeriod, timeofblock, txfee, txids, txinids, txstats, utxotype, utxovalue

utxotype  Retrieving types of UTXOs

Description
This function returns the types of the UTXO(s) in a transaction.

Usage
utxotype(con, txid)
Arguments

con: CONRPC, configuration object.
txid: character, the id of the transaction.

Value

character

Author(s)

Bernhard Pfaff

See Also

Other UtilityFuncs: bkfee, blockattime, blockstats, date2int, int2date, intMaxDay, intMinDay, intRangeDay, intRangePeriod, timeofblock, txfee, txids, txinids, txstats, utxoage, utxotype

utxovalue

Retrieving values of UTXOs

Description

This function returns the values of UTXO(s) in a transaction.

Usage

utxovalue(con, txid)

Arguments

con: CONRPC, configuration object.
txid: character, the id of the transaction.

Value

numeric

Author(s)

Bernhard Pfaff

See Also

Other UtilityFuncs: bkfee, blockattime, blockstats, date2int, int2date, intMaxDay, intMinDay, intRangeDay, intRangePeriod, timeofblock, txfee, txids, txinids, txstats, utxoage, utxotype
validBtcAdr

validBtcAdr  Validate S4-class BTCADR

Description

This function validates objects of S4-class BTCADR. Hereby, checks are conducted with respect to the first character of the addresses; their consistency with the net version and the correspondence of the checksums.

Usage

validBtcAdr(object)

Arguments

object  BTCADR object

Author(s)

Bernhard Pfaff

References

https://en.bitcoin.it/wiki/Address

See Also

Other BtcAdresses: BTCADR-class, PrivKey2PubKey, PrivKey2Wif, PubHash2BtcAdr, PubKey2PubHash, Wif2PrivKey, base58CheckDecode, base58CheckEncode, concatHex, createBtcAdr, createPrivatekey, decodeHex, hash160, hash256

verifychain  RPC-JSOn API: verifychain

Description

Verifies blockchain database.

Usage

verifychain(con, checklevel = NULL, nblocks = NULL)
verifytxoutproof

RPC-JSON API: verifytxoutproof

Description
Verifies that a proof points to a transaction in a block, returning the transaction it commits to and throwing an RPC error if the block is not in our best chain.

Usage
verifytxoutproof(con, proof)

Arguments
- **con** object of class CONRPC.
- **proof** character the hex-encoded proof generated by gettxoutproof.

Value
A S4-object of class ANSRPC.
Wif2PrivKey

Author(s)
Bernhard Pfaff

References

See Also
Other Blockchain RPCs: decodescript, getbestblockhash, getblockchaininfo, getblockcount, getblockhash, getblockheader, getblock, getchaintips, getchaintxstats, getdifficulty, getmempoolancestors, getmempooldescendants, getmempoolentry, getmempoolinfo, getrawmempool, gettxoutproof, gettxoutsetinfo, gettxout, preciousblock, pruneblockchain, verifychain

Wif2PrivKey Create private key from WIF

Description
Returns the corresponding private key from a WIF key.

Usage
Wif2PrivKey(wif)

Arguments
wif character, a WIF key.

Value
character, the corresponding private key.

Author(s)
Bernhard Pfaff

References
https://en.bitcoin.it/wiki/Wallet_import_format,
https://en.bitcoin.it/wiki/Address

See Also
Other BtcAdresses: BTCADR-class, PrivKey2PubKey, PrivKey2Wif, PubHash2BtcAadr, PubKey2PubHash, base58CheckDecode, base58CheckEncode, concatHex, createBtcAadr, createPrivateKey, decodeHex, hash160, hash256, validBtcAadr
Examples

pk1 <- createPrivateKey()
wif <- PrivKey2Wif(pk1)
pk2 <- Wif2PrivKey(wif)
identical(pk1, pk2)
### Index

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>*,ECPOINT, bigz-method (ecoperators)</td>
<td>19</td>
</tr>
<tr>
<td>*,bigz, ECPOINT-method (ecoperators)</td>
<td>19</td>
</tr>
<tr>
<td>+,ECPOINT, ECPOINT-method (ecoperators)</td>
<td>19</td>
</tr>
<tr>
<td>addnode</td>
<td>3, 10, 18, 24, 31, 38, 39, 52, 53, 59</td>
</tr>
<tr>
<td>AND (ecoperators)</td>
<td>19</td>
</tr>
<tr>
<td>AND, bigz, bigz-method (ecoperators)</td>
<td>19</td>
</tr>
<tr>
<td>ANSRPC-class</td>
<td>4</td>
</tr>
<tr>
<td>base58CheckDecode</td>
<td>5, 6, 9, 11, 14, 16, 46, 55, 57, 58, 67, 69</td>
</tr>
<tr>
<td>base58CheckEncode</td>
<td>5, 6, 9, 11, 14, 16, 46, 55, 57, 58, 67, 69</td>
</tr>
<tr>
<td>bkfee</td>
<td>7, 8, 15, 47–50, 62–66</td>
</tr>
<tr>
<td>blockattime</td>
<td>7, 7, 8, 15, 47–50, 62–66</td>
</tr>
<tr>
<td>blockstatts</td>
<td>7, 8, 15, 47–50, 62–66</td>
</tr>
<tr>
<td>BTCADR-class</td>
<td>9</td>
</tr>
<tr>
<td>clearbanned</td>
<td>4, 9, 18, 24, 31, 38, 39, 52, 53, 59</td>
</tr>
<tr>
<td>concatHex</td>
<td>5, 6, 9, 10, 14, 16, 46, 55, 57, 58, 67, 69</td>
</tr>
<tr>
<td>conrpc</td>
<td>4, 11, 12, 52, 59, 61</td>
</tr>
<tr>
<td>CONRPC-class</td>
<td>12</td>
</tr>
<tr>
<td>containsPoint, ecparam, bigz, bigz-method</td>
<td>12</td>
</tr>
<tr>
<td>(containsPoint)</td>
<td>12</td>
</tr>
<tr>
<td>containsPoint, ECPARAM, character, character-method (containsPoint)</td>
<td>12</td>
</tr>
<tr>
<td>containsPoint, ECPARAM, integer, integer-method (containsPoint)</td>
<td>12</td>
</tr>
<tr>
<td>createBtcAddr</td>
<td>5, 6, 9, 11, 13, 14, 16, 46, 55, 57, 58, 67, 69</td>
</tr>
<tr>
<td>createPrivateKey</td>
<td>5, 6, 9, 11, 14, 14, 16, 46, 55, 57, 58, 67, 69</td>
</tr>
<tr>
<td>date2int</td>
<td>7, 8, 15, 47–50, 62–66</td>
</tr>
<tr>
<td>decodeHex</td>
<td>5, 6, 9, 11, 14, 15, 46, 55, 57, 58, 67, 69</td>
</tr>
<tr>
<td>decoderawtransaction</td>
<td>16, 41</td>
</tr>
<tr>
<td>disconnectnode</td>
<td>4, 10, 18, 24, 31, 38, 39, 52, 53, 59</td>
</tr>
<tr>
<td>doubleUp</td>
<td>19</td>
</tr>
<tr>
<td>doubleUp, ECPOINT-method (ecoperators)</td>
<td>19</td>
</tr>
<tr>
<td>ecparam</td>
<td>13, 19, 20–23, 51</td>
</tr>
<tr>
<td>ECPARAM-class</td>
<td>21</td>
</tr>
<tr>
<td>EcparamOrNull-class</td>
<td>21</td>
</tr>
<tr>
<td>ecpoint</td>
<td>13, 20–22, 22, 23, 51</td>
</tr>
<tr>
<td>ECPOINT-class</td>
<td>23</td>
</tr>
<tr>
<td>getaddednodeinfo</td>
<td>4, 10, 18, 24, 31, 38, 39, 52, 53, 59</td>
</tr>
<tr>
<td>getbestblockhash</td>
<td>17, 24, 26–30, 32, 34–37, 40, 42–44, 54, 56, 68, 69</td>
</tr>
<tr>
<td>getblock</td>
<td>17, 25, 25, 26–30, 32, 34–37, 40, 42–44, 54, 56, 68, 69</td>
</tr>
<tr>
<td>getblockchaininfo</td>
<td>17, 25, 26, 27–30, 32, 34–37, 40, 42–44, 54, 56, 68, 69</td>
</tr>
<tr>
<td>getblockcount</td>
<td>17, 25, 26, 27, 28–30, 32, 34–37, 40, 42–44, 54, 56, 68, 69</td>
</tr>
<tr>
<td>getblockhash</td>
<td>17, 25–27, 27, 29, 30, 32, 34–37, 40, 42–44, 54, 56, 68, 69</td>
</tr>
<tr>
<td>gethelp</td>
<td>32, 33, 45</td>
</tr>
<tr>
<td>getInfo</td>
<td>33, 33, 45</td>
</tr>
</tbody>
</table>

71
getmempoolancestors, 17, 25–30, 32, 34, 35–37, 40, 42–44, 54, 56, 68, 69
getmempooldescendants, 17, 25–30, 32, 34, 35, 36, 37, 40, 42–44, 54, 56, 68, 69
getmempoolentry, 17, 25–30, 32, 34, 35, 36, 37, 40, 42–44, 54, 56, 68, 69
getnettotals, 4, 10, 18, 24, 31, 37, 39, 52, 53, 59
getnetworkinfo, 4, 10, 18, 24, 31, 38, 39, 52, 53, 59
getpeerinfo, 4, 10, 18, 24, 31, 38, 39, 52, 53, 59
getrawtransaction, 17, 41
gettxout, 17, 25–30, 32, 34–37, 40, 42, 43, 44, 54, 56, 68, 69
gettxoutpremature, 17, 25–30, 32, 34–37, 40, 42, 43, 44, 54, 56, 68, 69
gettxoutsetinfo, 17, 25–30, 32, 34–37, 40, 42, 43, 44, 54, 56, 68, 69
getwalletinfo, 33, 44
hash160, 5, 6, 9, 11, 14, 16, 45, 46, 55, 57, 58, 67, 69
hash256, 5, 6, 9, 11, 14, 16, 46, 46, 55, 57, 58, 67, 69
int2date, 7, 8, 15, 47, 48–50, 62–66
intMaxDay, 7, 8, 15, 47, 48, 49, 50, 62–66
intMinDay, 7, 8, 15, 47, 48, 49, 50, 62–66
intRangeDay, 7, 8, 15, 47–49, 49, 50, 62–66
intRangePeriod, 7, 8, 15, 47–50, 50, 62–66
isNull, 13, 20–23, 51
isNull, EPOINT-method (isNull), 51
leftmostBit (ecoperators), 19
leftmostBit, bigz-method (ecoperators), 19
lstbanned, 4, 10, 18, 24, 31, 38, 39, 51, 53, 59
NullOrCharacter-class, 52
NullOrInteger-class, 52
ping, 4, 10, 18, 24, 31, 38, 39, 52, 53, 59
PrivKey2PubKey, 5, 6, 9, 11, 14, 16, 46, 54, 55, 57, 58, 67, 69
PrivKey2Wif, 5, 6, 9, 11, 14, 16, 46, 55, 55, 57, 58, 67, 69
PubHash2BtcAddr, 5, 6, 9, 11, 14, 16, 46, 55, 57, 58, 67, 69
PubKey2PubHash, 5, 6, 9, 11, 14, 16, 46, 55, 57, 57, 67, 69
rpcpost, 4, 11, 12, 52, 58, 61
setnetworkactive, 4, 10, 18, 24, 31, 38, 39, 52, 53, 59
show, 60
show, ANSrpc-method (show), 60
show, BTCADR-method (show), 60
show, ECPARAM-method (show), 60
startbtc, 4, 11, 12, 52, 59, 60, 61
stopbtc, 4, 11, 12, 52, 59, 61, 61
timofblock, 7, 8, 15, 47–50, 61, 62–66
txfee, 7, 8, 15, 47–50, 62, 62, 63–66
txids, 7, 8, 15, 47–50, 62, 63, 64–66
txinids, 7, 8, 15, 47–50, 62, 63, 63, 64–66
txstats, 7, 8, 15, 47–50, 62–64, 64, 65, 66
utxoage, 7, 8, 15, 47–50, 62–64, 65, 66
utxotype, 7, 8, 15, 47–50, 62–65, 65, 66
utxovalue, 7, 8, 15, 47–50, 62–66, 66
validBtcAddr, 5, 6, 9, 11, 14, 16, 46, 55, 57, 58, 67, 69
Wif2PrivKey, 5, 6, 9, 11, 14, 16, 46, 55, 57, 58, 67, 69