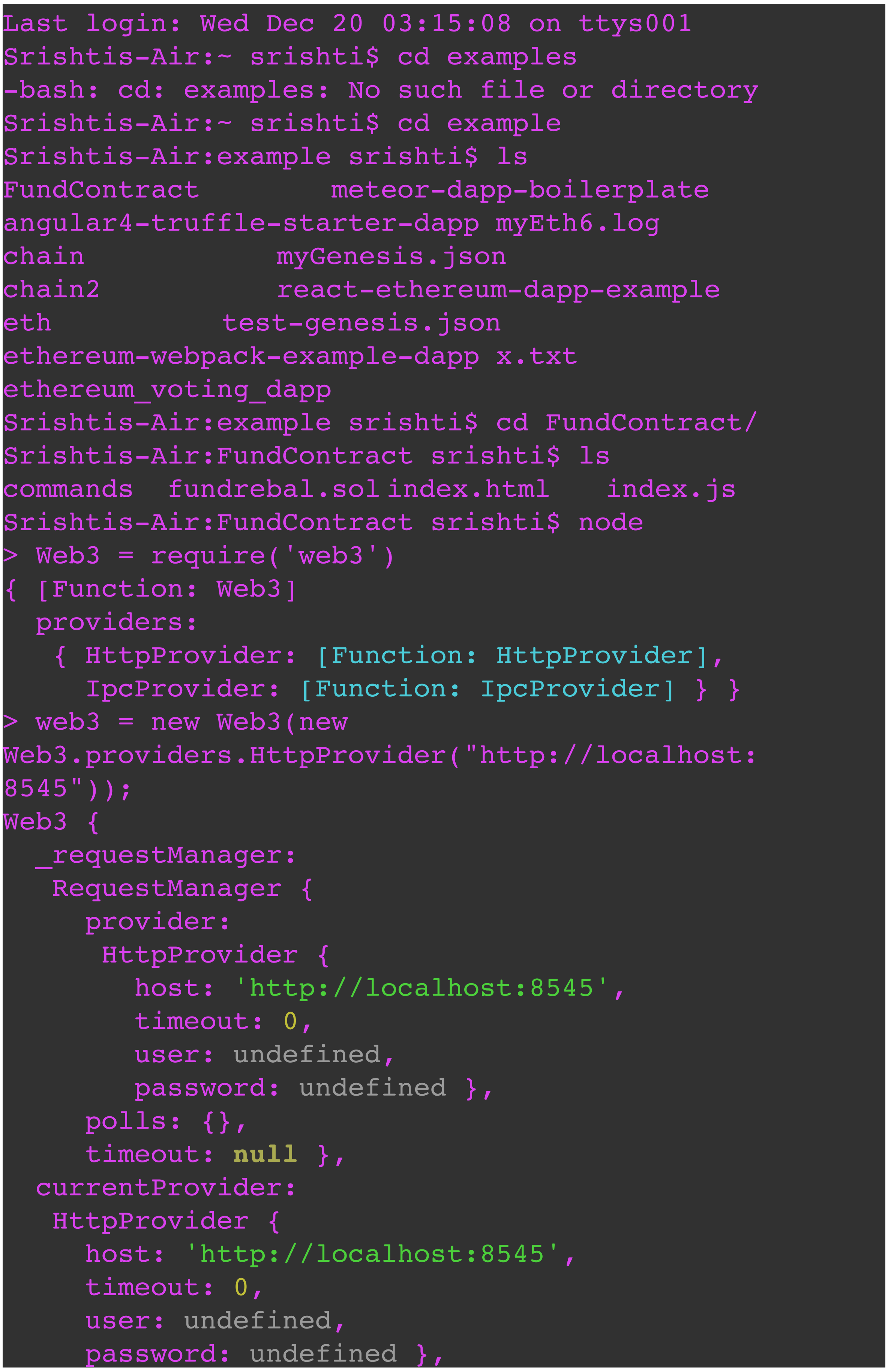
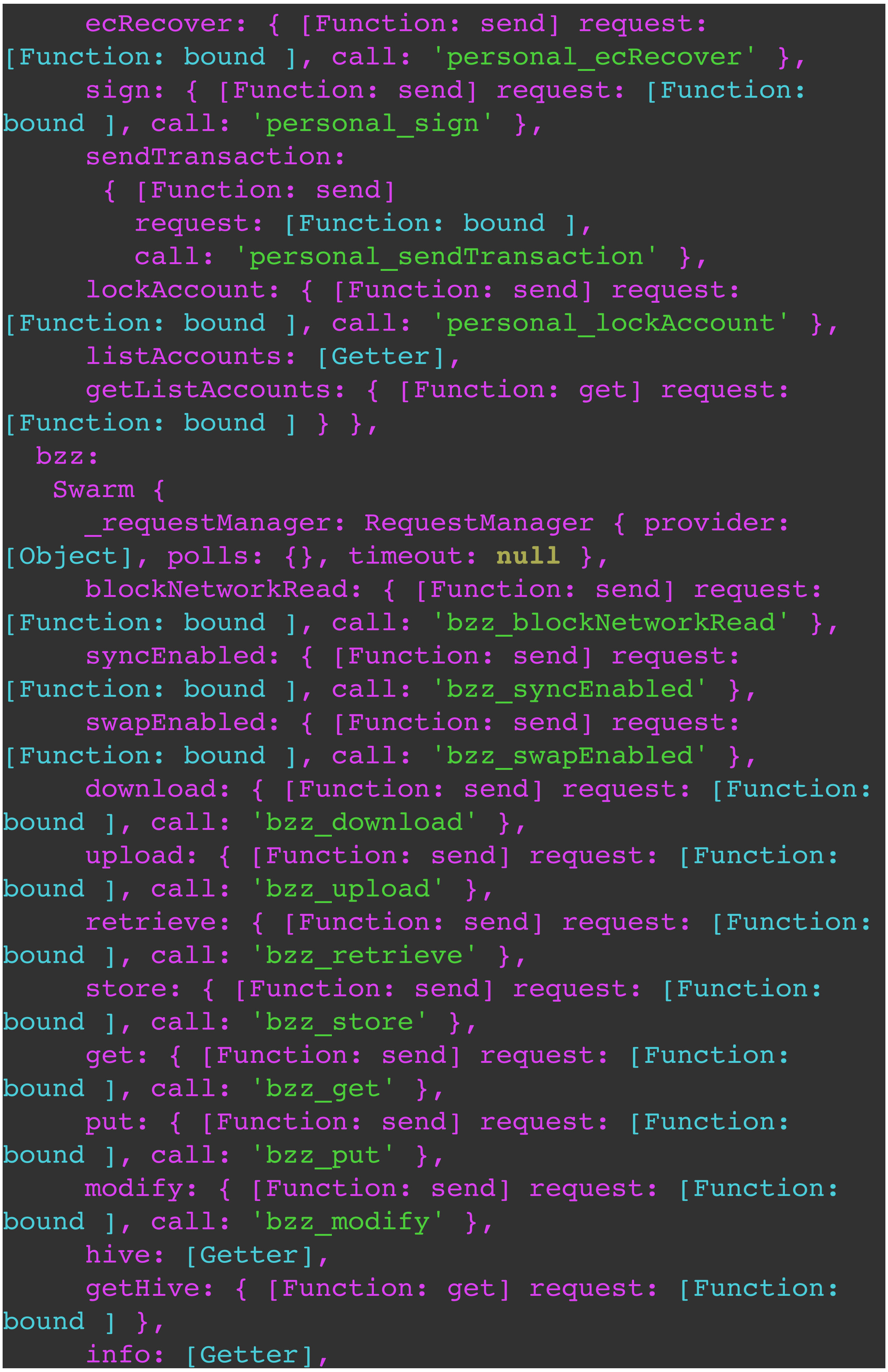


```
Last login: Wed Dec 20 03:15:08 on ttys001
Srishtis-Air:~ srishti$ cd examples
-bash: cd: examples: No such file or directory
Srishtis-Air:~ srishti$ cd example
Srishtis-Air:example srishti$ ls
FundContract        meteor-dapp-boilerplate
angular4-truffle-starter-dapp myEth6.log
chain             myGenesis.json
chain2            react-ethereum-dapp-example
eth           test-genesis.json
ethereum-webpack-example-dapp x.txt
ethereum_voting_dapp
Srishtis-Air:example srishti$ cd FundContract/
Srishtis-Air:FundContract srishti$ ls
commands  fundrebal.sol index.html   index.js
Srishtis-Air:FundContract srishti$ node
> Web3 = require('web3')
{ [Function: Web3]
  providers:
   { HttpProvider: [Function: HttpProvider],
     IpcProvider: [Function: IpcProvider] } }
> web3 = new Web3(new
Web3.providers.HttpProvider("http://localhost:
8545"));
Web3 {
  _requestManager:
   RequestManager {
     provider:
      HttpProvider {
        host: 'http://localhost:8545',
        timeout: 0,
        user: undefined,
        password: undefined },
     polls: {},
     timeout: null },
  currentProvider:
   HttpProvider {
     host: 'http://localhost:8545',
     timeout: 0,
     user: undefined,
     password: undefined },
  eth:
   Eth {
     _requestManager: RequestManager { provider:
[Object], polls: {}, timeout: null },
     getBalance: { [Function: send] request:
[Function: bound ], call: 'eth_getBalance' },
     getStorageAt: { [Function: send] request:
[Function: bound ], call: 'eth_getStorageAt' },
     getCode: { [Function: send] request: [Function:
bound ], call: 'eth_getCode' },
     getBlock: { [Function: send] request: [Function:
bound ], call: [Function: blockCall] },
     getUncle: { [Function: send] request: [Function:
bound ], call: [Function: uncleCall] },
     getCompilers: { [Function: send] request:
[Function: bound ], call: 'eth_getCompilers' },
     getBlockTransactionCount:
      { [Function: send]
        request: [Function: bound ],
        call: [Function:
getBlockTransactionCountCall] },
     getBlockUncleCount:
      { [Function: send]
        request: [Function: bound ],
        call: [Function: uncleCountCall] },
     getTransaction:
      { [Function: send]
        request: [Function: bound ],
        call: 'eth_getTransactionByHash' },
     getTransactionFromBlock:
      { [Function: send]
        request: [Function: bound ],
        call: [Function: transactionFromBlockCall] },
     getTransactionReceipt:
      { [Function: send]
        request: [Function: bound ],
        call: 'eth_getTransactionReceipt' },
     getTransactionCount: { [Function: send] request:
[Function: bound ], call:
'eth_getTransactionCount' },
     call: { [Function: send] request: [Function:
```

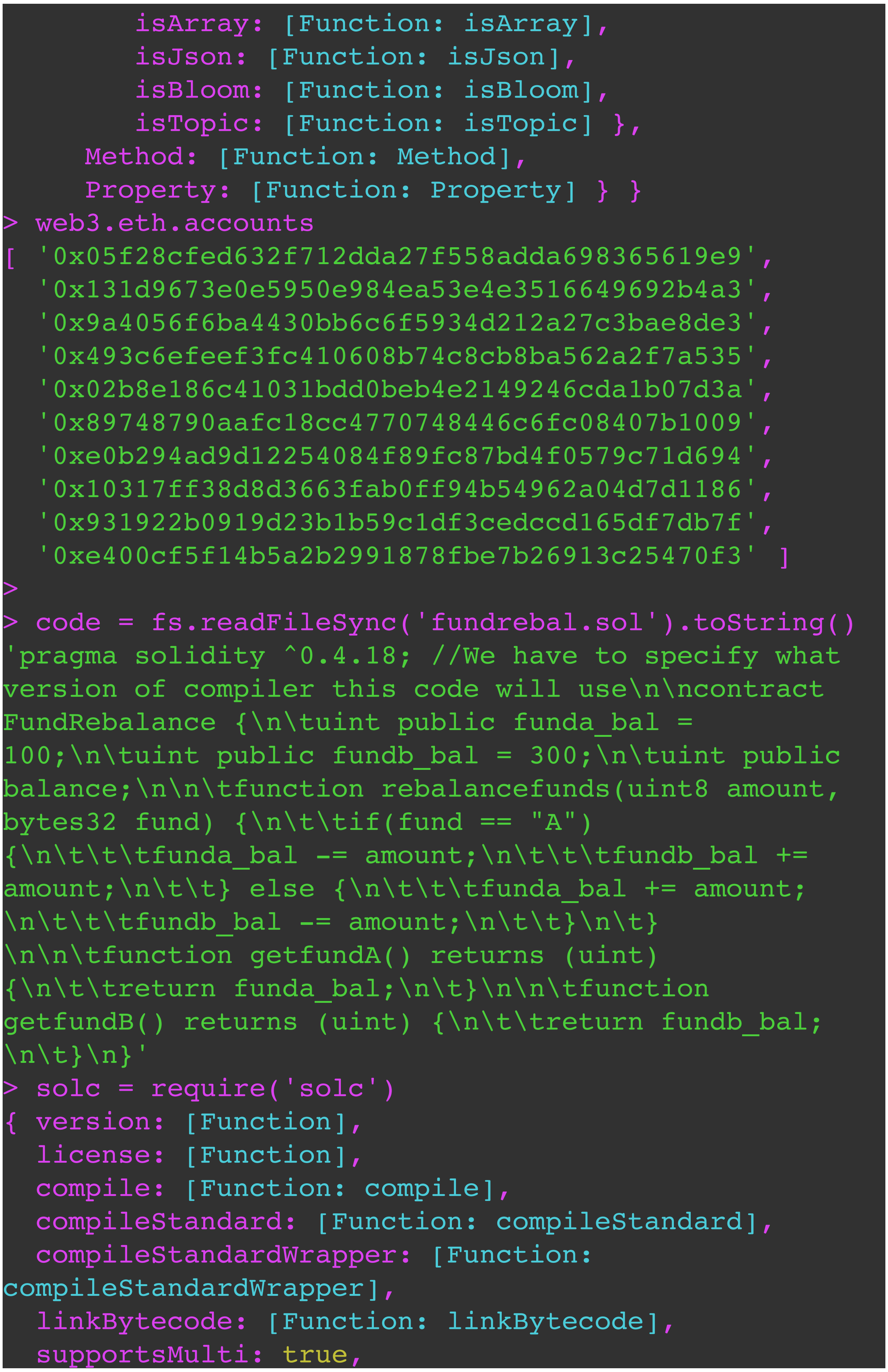
```
bound ], call: 'eth_call' },
     estimateGas: { [Function: send] request:
[Function: bound ], call: 'eth_estimateGas' },
     sendRawTransaction: { [Function: send] request:
[Function: bound ], call: 'eth_sendRawTransaction' },
     signTransaction: { [Function: send] request:
[Function: bound ], call: 'eth_signTransaction' },
     sendTransaction: { [Function: send] request:
[Function: bound ], call: 'eth_sendTransaction' },
     sign: { [Function: send] request: [Function:
bound ], call: 'eth_sign' },
     compile: { solidity: [Object], lll: [Object],
serpent: [Object] },
     submitWork: { [Function: send] request:
[Function: bound ], call: 'eth_submitWork' },
     getWork: { [Function: send] request: [Function:
bound ], call: 'eth_getWork' },
     coinbase: [Getter],
     getCoinbase: { [Function: get] request:
[Function: bound ] },
     mining: [Getter],
     getMining: { [Function: get] request: [Function:
bound ] },
     hashrate: [Getter],
     getHashrate: { [Function: get] request:
[Function: bound ] },
     syncing: [Getter],
     getSyncing: { [Function: get] request:
[Function: bound ] },
     gasPrice: [Getter],
     getGasPrice: { [Function: get] request:
[Function: bound ] },
     accounts: [Getter],
     getAccounts: { [Function: get] request:
[Function: bound ] },
     blockNumber: [Getter],
     getBlockNumber: { [Function: get] request:
[Function: bound ] },
     protocolVersion: [Getter],
     getProtocolVersion: { [Function: get] request:
[Function: bound ] },
     iban:
      { [Function: Iban]
        fromAddress: [Function],
        fromBban: [Function],
        createIndirect: [Function],
        isValid: [Function] },
     sendIBANTransaction: [Function: bound
transfer] },
  db:
   DB {
     _requestManager: RequestManager { provider:
[Object], polls: {}, timeout: null },
     putString: { [Function: send] request:
[Function: bound ], call: 'db_putString' },
     getString: { [Function: send] request:
[Function: bound ], call: 'db_getString' },
     putHex: { [Function: send] request: [Function:
bound ], call: 'db_putHex' },
     getHex: { [Function: send] request: [Function:
bound ], call: 'db_getHex' } },
  shh:
   Shh {
     _requestManager: RequestManager { provider:
[Object], polls: {}, timeout: null },
     version: { [Function: send] request: [Function:
bound ], call: 'shh_version' },
     info: { [Function: send] request: [Function:
bound ], call: 'shh_info' },
     setMaxMessageSize: { [Function: send] request:
[Function: bound ], call: 'shh_setMaxMessageSize' },
     setMinPoW: { [Function: send] request:
[Function: bound ], call: 'shh_setMinPoW' },
     markTrustedPeer: { [Function: send] request:
[Function: bound ], call: 'shh_markTrustedPeer' },
     newKeyPair: { [Function: send] request:
[Function: bound ], call: 'shh_newKeyPair' },
     addPrivateKey: { [Function: send] request:
[Function: bound ], call: 'shh_addPrivateKey' },
     deleteKeyPair: { [Function: send] request:
[Function: bound ], call: 'shh_deleteKeyPair' },
     hasKeyPair: { [Function: send] request:
```

```
[Function: bound ], call: 'shh_hasKeyPair' },
     getPublicKey: { [Function: send] request:
[Function: bound ], call: 'shh_getPublicKey' },
     getPrivateKey: { [Function: send] request:
[Function: bound ], call: 'shh_getPrivateKey' },
     newSymKey: { [Function: send] request:
[Function: bound ], call: 'shh_newSymKey' },
     addSymKey: { [Function: send] request:
[Function: bound ], call: 'shh_addSymKey' },
     generateSymKeyFromPassword:
      { [Function: send]
        request: [Function: bound ],
        call: 'shh_generateSymKeyFromPassword' },
     hasSymKey: { [Function: send] request:
[Function: bound ], call: 'shh_hasSymKey' },
     getSymKey: { [Function: send] request:
[Function: bound ], call: 'shh_getSymKey' },
     deleteSymKey: { [Function: send] request:
[Function: bound ], call: 'shh_deleteSymKey' },
     post: { [Function: send] request: [Function:
bound ], call: 'shh_post' } },
  net:
   Net {
     _requestManager: RequestManager { provider:
[Object], polls: {}, timeout: null },
     listening: [Getter],
     getListening: { [Function: get] request:
[Function: bound ] },
     peerCount: [Getter],
     getPeerCount: { [Function: get] request:
[Function: bound ] } },
  personal:
   Personal {
     _requestManager: RequestManager { provider:
[Object], polls: {}, timeout: null },
     newAccount: { [Function: send] request:
[Function: bound ], call: 'personal_newAccount' },
     importRawKey: { [Function: send] request:
[Function: bound ], call: 'personal_importRawKey' },
     unlockAccount: { [Function: send] request:
[Function: bound ], call: 'personal_unlockAccount' },
```

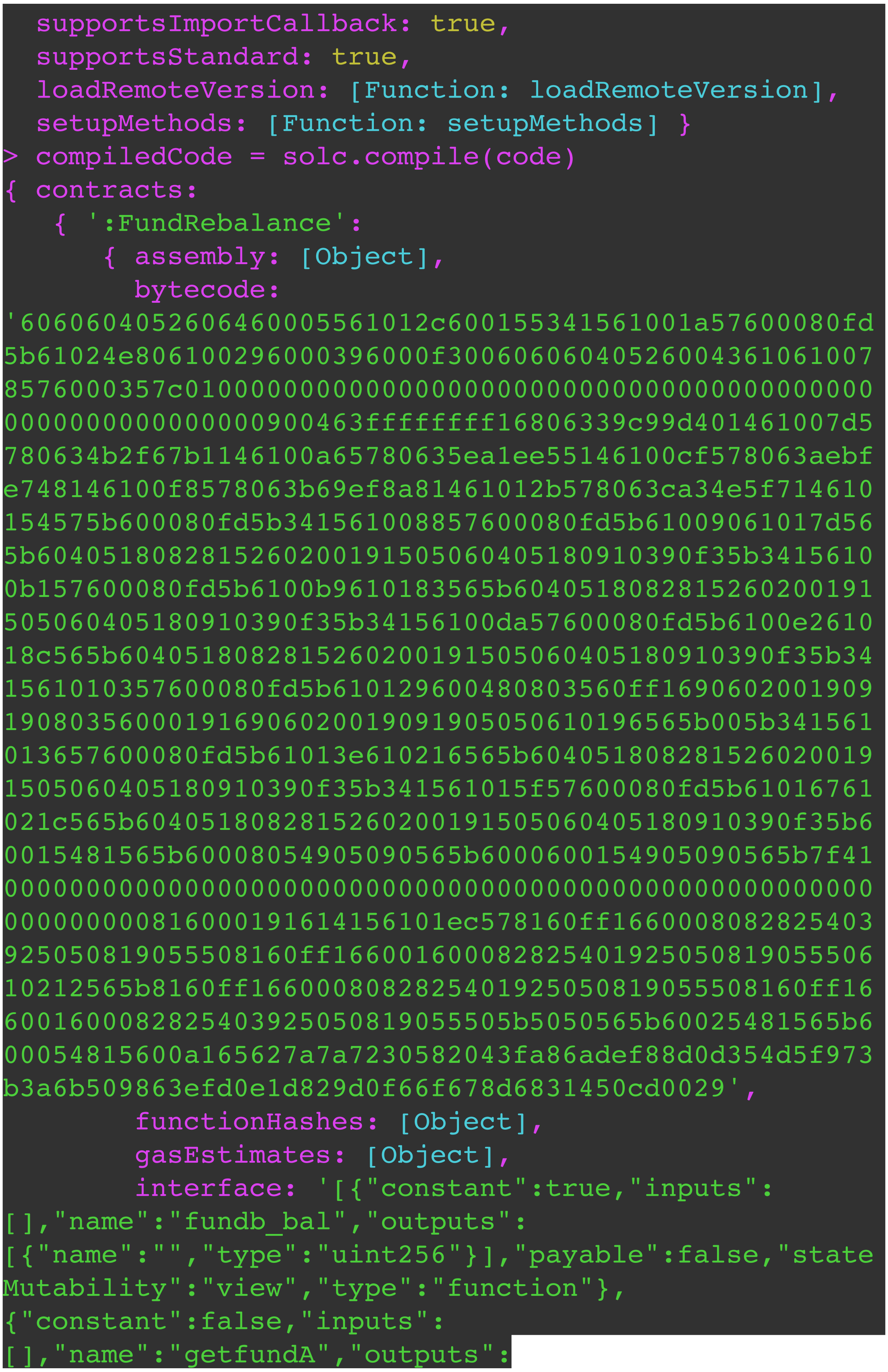


```
     ecRecover: { [Function: send] request:
[Function: bound ], call: 'personal_ecRecover' },
     sign: { [Function: send] request: [Function:
bound ], call: 'personal_sign' },
     sendTransaction:
      { [Function: send]
        request: [Function: bound ],
        call: 'personal_sendTransaction' },
     lockAccount: { [Function: send] request:
[Function: bound ], call: 'personal_lockAccount' },
     listAccounts: [Getter],
     getListAccounts: { [Function: get] request:
[Function: bound ] } },
  bzz:
   Swarm {
     _requestManager: RequestManager { provider:
[Object], polls: {}, timeout: null },
     blockNetworkRead: { [Function: send] request:
[Function: bound ], call: 'bzz_blockNetworkRead' },
     syncEnabled: { [Function: send] request:
[Function: bound ], call: 'bzz_syncEnabled' },
     swapEnabled: { [Function: send] request:
[Function: bound ], call: 'bzz_swapEnabled' },
     download: { [Function: send] request: [Function:
bound ], call: 'bzz_download' },
     upload: { [Function: send] request: [Function:
bound ], call: 'bzz_upload' },
     retrieve: { [Function: send] request: [Function:
bound ], call: 'bzz_retrieve' },
     store: { [Function: send] request: [Function:
bound ], call: 'bzz_store' },
     get: { [Function: send] request: [Function:
bound ], call: 'bzz_get' },
     put: { [Function: send] request: [Function:
bound ], call: 'bzz_put' },
     modify: { [Function: send] request: [Function:
bound ], call: 'bzz_modify' },
     hive: [Getter],
     getHive: { [Function: get] request: [Function:
bound ] },
     info: [Getter],
     getInfo: { [Function: get] request: [Function:
bound ] } },
  settings: Settings { defaultBlock: 'latest',
defaultAccount: undefined },
  version:
   { api: '0.20.1',
     node: [Getter],
     getNode: { [Function: get] request: [Function:
bound ] },
     network: [Getter],
     getNetwork: { [Function: get] request:
[Function: bound ] },
     ethereum: [Getter],
     getEthereum: { [Function: get] request:
[Function: bound ] },
     whisper: [Getter],
     getWhisper: { [Function: get] request:
[Function: bound ] } },
  providers:
   { HttpProvider: [Function: HttpProvider],
     IpcProvider: [Function: IpcProvider] },
  _extend:
   { [Function: ex]
     formatters:
      { inputDefaultBlockNumberFormatter: [Function:
inputDefaultBlockNumberFormatter],
        inputBlockNumberFormatter: [Function:
inputBlockNumberFormatter],
        inputCallFormatter: [Function:
inputCallFormatter],
        inputTransactionFormatter: [Function:
inputTransactionFormatter],
        inputAddressFormatter: [Function:
inputAddressFormatter],
        inputPostFormatter: [Function:
inputPostFormatter],
        outputBigNumberFormatter: [Function:
outputBigNumberFormatter],
        outputTransactionFormatter: [Function:
outputTransactionFormatter],
        outputTransactionReceiptFormatter: [Function:
```

```
outputTransactionReceiptFormatter],
        outputBlockFormatter: [Function:
outputBlockFormatter],
        outputLogFormatter: [Function:
outputLogFormatter],
        outputPostFormatter: [Function:
outputPostFormatter],
        outputSyncingFormatter: [Function:
outputSyncingFormatter] },
     utils:
      { padLeft: [Function: padLeft],
        padRight: [Function: padRight],
        toHex: [Function: toHex],
        toDecimal: [Function: toDecimal],
        fromDecimal: [Function: fromDecimal],
        toUtf8: [Function: toUtf8],
        toAscii: [Function: toAscii],
        fromUtf8: [Function: fromUtf8],
        fromAscii: [Function: fromAscii],
        transformToFullName: [Function:
transformToFullName],
        extractDisplayName: [Function:
extractDisplayName],
        extractTypeName: [Function: extractTypeName],
        toWei: [Function: toWei],
        fromWei: [Function: fromWei],
        toBigNumber: [Function: toBigNumber],
        toTwosComplement: [Function:
toTwosComplement],
        toAddress: [Function: toAddress],
        isBigNumber: [Function: isBigNumber],
        isStrictAddress: [Function: isStrictAddress],
        isAddress: [Function: isAddress],
        isChecksumAddress: [Function:
isChecksumAddress],
        toChecksumAddress: [Function:
toChecksumAddress],
        isFunction: [Function: isFunction],
        isString: [Function: isString],
        isObject: [Function: isObject],
        isBoolean: [Function: isBoolean],
```



```
        isArray: [Function: isArray],
        isJson: [Function: isJson],
        isBloom: [Function: isBloom],
        isTopic: [Function: isTopic] },
     Method: [Function: Method],
     Property: [Function: Property] } }
> web3.eth.accounts
[ '0x05f28cfed632f712dda27f558adda698365619e9',
  '0x131d9673e0e5950e984ea53e4e3516649692b4a3',
  '0x9a4056f6ba4430bb6c6f5934d212a27c3bae8de3',
  '0x493c6efeef3fc410608b74c8cb8ba562a2f7a535',
  '0x02b8e186c41031bdd0beb4e2149246cda1b07d3a',
  '0x89748790aafc18cc4770748446c6fc08407b1009',
  '0xe0b294ad9d12254084f89fc87bd4f0579c71d694',
  '0x10317ff38d8d3663fab0ff94b54962a04d7d1186',
  '0x931922b0919d23b1b59c1df3cedccd165df7db7f',
  '0xe400cf5f14b5a2b2991878fbe7b26913c25470f3' ]
>
> code = fs.readFileSync('fundrebal.sol').toString()
'pragma solidity ^0.4.18; //We have to specify what
version of compiler this code will use\n\ncontract
FundRebalance {\n\tuint public funda_bal =
100;\n\tuint public fundb_bal = 300;\n\tuint public
balance;\n\n\tfunction rebalancefunds(uint8 amount,
bytes32 fund) {\n\t\tif(fund == "A")
{\n\t\t\tfunda_bal -= amount;\n\t\t\tfundb_bal +=
amount;\n\t\t} else {\n\t\t\tfunda_bal += amount;
\n\t\t\tfundb_bal -= amount;\n\t\t}\n\t}
\n\n\tfunction getfundA() returns (uint)
{\n\t\treturn funda_bal;\n\t}\n\n\tfunction
getfundB() returns (uint) {\n\t\treturn fundb_bal;
\n\t}\n}'
> solc = require('solc')
{ version: [Function],
  license: [Function],
  compile: [Function: compile],
  compileStandard: [Function: compileStandard],
  compileStandardWrapper: [Function:
compileStandardWrapper],
  linkBytecode: [Function: linkBytecode],
  supportsMulti: true,
```

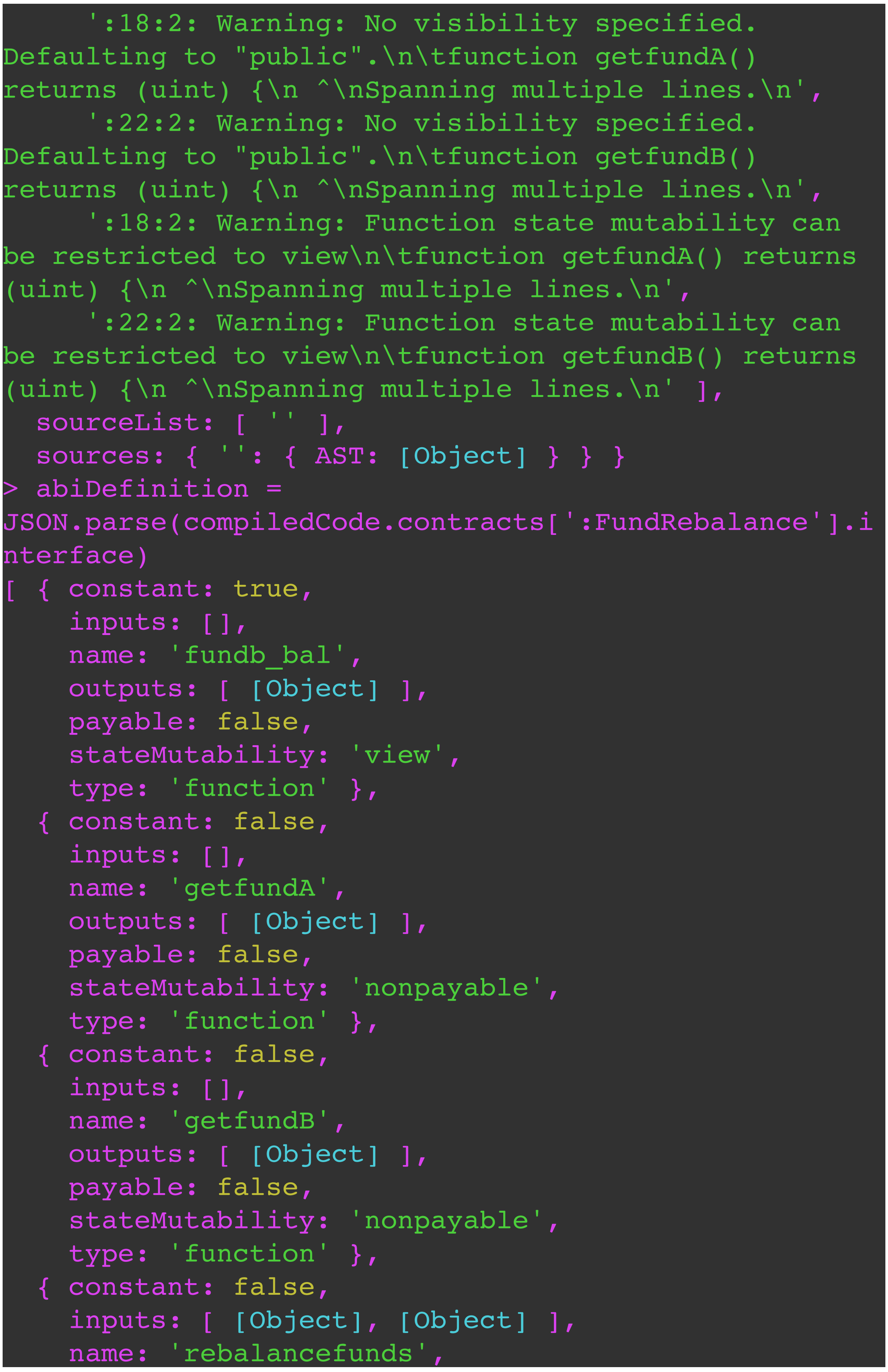


```
  supportsImportCallback: true,
  supportsStandard: true,
  loadRemoteVersion: [Function: loadRemoteVersion],
  setupMethods: [Function: setupMethods] }
> compiledCode = solc.compile(code)
{ contracts:
   { ':FundRebalance':
      { assembly: [Object],
        bytecode:
'6060604052606460005561012c600155341561001a57600080fd
5b61024e806100296000396000f300606060405260043610610 07
8576000357c0100000000000000000000000000000000000000000
0000000000000000900463ffffffff16806339c99d401461007d5
780634b2f67b1146100a65780635ea1ee55146100cf578063aebf
e748146100f8578063b69ef8a81461012b578063ca34e5f714610
154575b600080fd5b341561008857600080fd5b61009061017d56
5b6040518082815260200191505060405180910390f35b3415610
0b157600080fd5b6100b961018356 5b60405180828152602001 91
5050604051809103 90f35b34156100da57600080fd5b6100e2610
18c565b6040518082815260200191505060405180910390f35b34
1561010357600080fd5b6101296004808035 60ff1690602001909
190803560001916906020019091905050610196565b005b341561
013657600080fd5b61013e610216565b60405180828152602001 9
1505060405180910390f35b341561015f57600080fd5b61016761
021c565b6040518082815260200191505060405180910390f35b6
0015481565b6000805490509056 5b6000600154905090565b7f41
00000000000000000000000000000000000000000000000000000
0000000008160001916141561 01ec578160ff1660008082825403
9250508190555081 60ff16600160008282540192505081905550 6
10212565b8160ff1660008082825401925050819055508160ff16
6001600082825403925050819055505b5050565b60025481565b6
00054815600a165627a7a72305820 43fa86adef88d0d354d5f973
b3a6b509863efd0e1d829d0f66f678d6831450cd0029',
        functionHashes: [Object],
        gasEstimates: [Object],
        interface: '[{"constant":true,"inputs":
[],"name":"fundb_bal","outputs":
[{"name":"","type":"uint256"}],"payable":false,"state
Mutability":"view","type":"function"},
{"constant":false,"inputs":
[],"name":"getfundA","outputs":
```

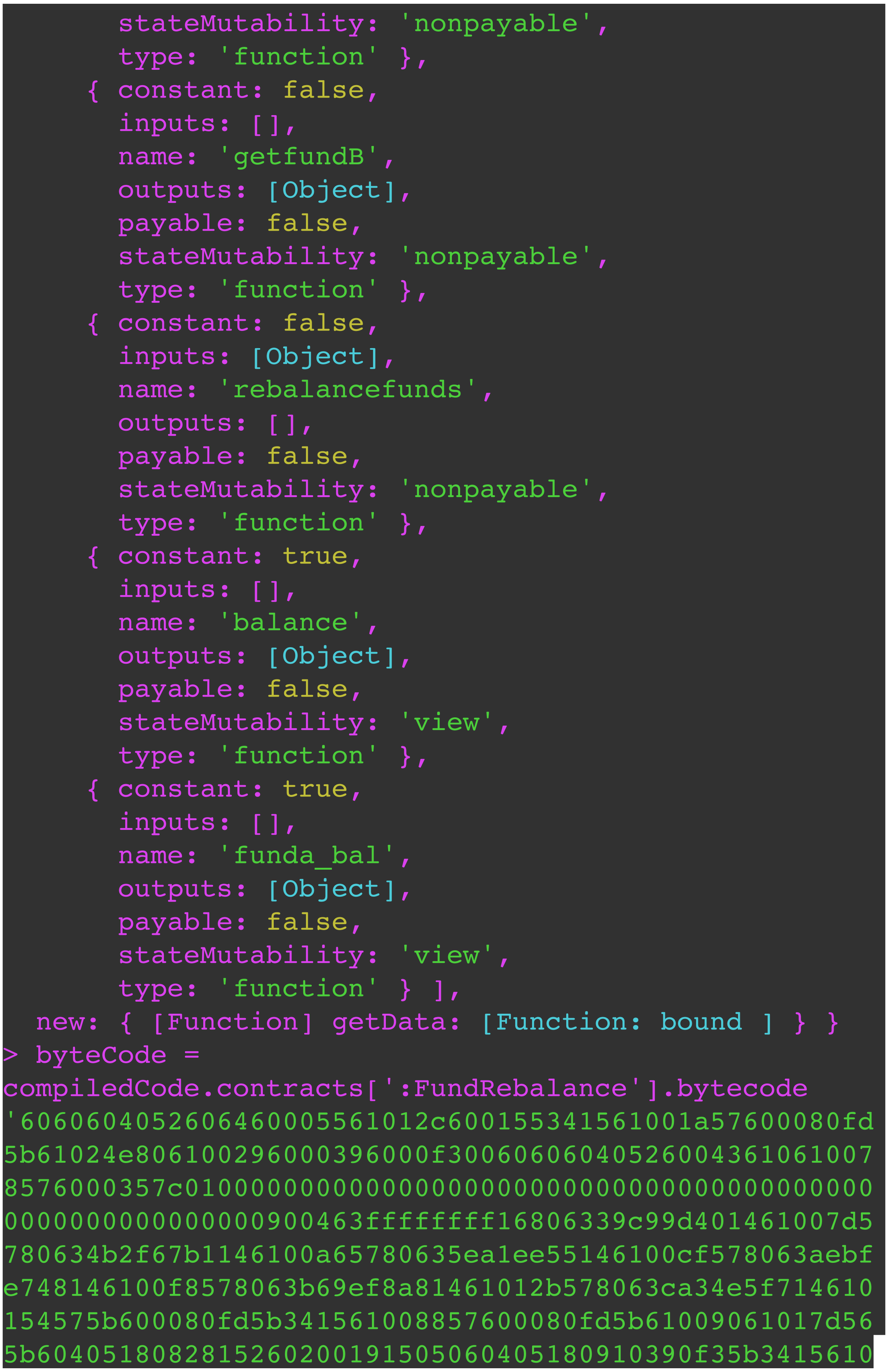
```
[{"name":"","type":"uint256"}],"payable":false,"state
Mutability":"nonpayable","type":"function"},
{"constant":false,"inputs":
[],"name":"getfundB","outputs":
[{"name":"","type":"uint256"}],"payable":false,"state
Mutability":"nonpayable","type":"function"},
{"constant":false,"inputs":
[{"name":"amount","type":"uint8"},
{"name":"fund","type":"bytes32"}],"name":"rebalancefu
nds","outputs":
[],"payable":false,"stateMutability":"nonpayable","ty
pe":"function"},{"constant":true,"inputs":
[],"name":"balance","outputs":
[{"name":"","type":"uint256"}],"payable":false,"state
Mutability":"view","type":"function"},
{"constant":true,"inputs":
[],"name":"funda_bal","outputs":
[{"name":"","type":"uint256"}],"payable":false,"state
Mutability":"view","type":"function"}]',
        metadata: '{"compiler":
{"version":"0.4.19+commit.c4cbbb05"},"language":"Soli
dity","output":{"abi":[{"constant":true,"inputs":
[],"name":"fundb_bal","outputs":
[{"name":"","type":"uint256"}],"payable":false,"state
Mutability":"view","type":"function"},
{"constant":false,"inputs":
[],"name":"getfundA","outputs":
[{"name":"","type":"uint256"}],"payable":false,"state
Mutability":"nonpayable","type":"function"},
{"constant":false,"inputs":
[],"name":"getfundB","outputs":
[{"name":"","type":"uint256"}],"payable":false,"state
Mutability":"nonpayable","type":"function"},
{"constant":false,"inputs":
[{"name":"amount","type":"uint8"},
{"name":"fund","type":"bytes32"}],"name":"rebalancefu
nds","outputs":
[],"payable":false,"stateMutability":"nonpayable","ty
pe":"function"},{"constant":true,"inputs":
[],"name":"balance","outputs":
[{"name":"","type":"uint256"}],"payable":false,"state
Mutability":"view","type":"function"},
{"constant":true,"inputs":
[],"name":"funda_bal","outputs":
[{"name":"","type":"uint256"}],"payable":false,"state
Mutability":"view","type":"function"}],"devdoc":
{"methods":{}},"userdoc":{"methods":{}}},"settings":
{"compilationTarget":
{"":"FundRebalance"},"libraries":{},"optimizer":
{"enabled":false,"runs":200},"remappings":
[]},"sources":{"":
{"keccak256":"0x65e32bc05f1ab59fb441bfa3cc3a934ddcbff
fa1f18535819edf33ff8643d59e","urls":["bzzr://
6dc5d36b427dfbb434b67b6d73e7d33dad6fc0060a5aecf69fe4a
fadf50ee094"]}},"version":1}',
        opcodes: 'PUSH1 0x60 PUSH1 0x40 MSTORE PUSH1
0x64 PUSH1 0x0 SSTORE PUSH2 0x12C PUSH1 0x1 SSTORE
CALLVALUE ISZERO PUSH2 0x1A JUMPI PUSH1 0x0 DUP1
REVERT JUMPDEST PUSH2 0x24E DUP1 PUSH2 0x29 PUSH1 0x0
CODECOPY PUSH1 0x0 RETURN STOP PUSH1 0x60 PUSH1 0x40
MSTORE PUSH1 0x4 CALLDATASIZE LT PUSH2 0x78 JUMPI
PUSH1 0x0 CALLDATALOAD PUSH29
0x100000000000000000000000000000000000000000000000000
000000 SWAP1 DIV PUSH4 0xFFFFFFFF AND DUP1 PUSH4
0x39C99D40 EQ PUSH2 0x7D JUMPI DUP1 PUSH4 0x4B2F67B1
EQ PUSH2 0xA6 JUMPI DUP1 PUSH4 0x5EA1EE55 EQ PUSH2
0xCF JUMPI DUP1 PUSH4 0xAEBFE748 EQ PUSH2 0xF8 JUMPI
DUP1 PUSH4 0xB69EF8A8 EQ PUSH2 0x12B JUMPI DUP1 PUSH4
0xCA34E5F7 EQ PUSH2 0x154 JUMPI JUMPDEST PUSH1 0x0
DUP1 REVERT JUMPDEST CALLVALUE ISZERO PUSH2 0x88
JUMPI PUSH1 0x0 DUP1 REVERT JUMPDEST PUSH2 0x90 PUSH2
0x17D JUMP JUMPDEST PUSH1 0x40 MLOAD DUP1 DUP3 DUP2
MSTORE PUSH1 0x20 ADD SWAP2 POP POP PUSH1 0x40 MLOAD
DUP1 SWAP2 SUB SWAP1 RETURN JUMPDEST CALLVALUE ISZERO
PUSH2 0xB1 JUMPI PUSH1 0x0 DUP1 REVERT JUMPDEST PUSH2
0xB9 PUSH2 0x183 JUMP JUMPDEST PUSH1 0x40 MLOAD DUP1
DUP3 DUP2 MSTORE PUSH1 0x20 ADD SWAP2 POP POP PUSH1
0x40 MLOAD DUP1 SWAP2 SUB SWAP1 RETURN JUMPDEST
CALLVALUE ISZERO PUSH2 0xDA JUMPI PUSH1 0x0 DUP1
REVERT JUMPDEST PUSH2 0xE2 PUSH2 0x18C JUMP JUMPDEST
PUSH1 0x40 MLOAD DUP1 DUP3 DUP2 MSTORE PUSH1 0x20 ADD
SWAP2 POP POP PUSH1 0x40 MLOAD DUP1 SWAP2 SUB SWAP1
```

```
RETURN JUMPDEST CALLVALUE ISZERO PUSH2 0x103 JUMPI
PUSH1 0x0 DUP1 REVERT JUMPDEST PUSH2 0x129 PUSH1 0x4
DUP1 DUP1 CALLDATALOAD PUSH1 0xFF AND SWAP1 PUSH1
0x20 ADD SWAP1 SWAP2 SWAP1 DUP1 CALLDATALOAD PUSH1
0x0 NOT AND SWAP1 PUSH1 0x20 ADD SWAP1 SWAP2 SWAP1
POP POP PUSH2 0x196 JUMP JUMPDEST STOP JUMPDEST
CALLVALUE ISZERO PUSH2 0x136 JUMPI PUSH1 0x0 DUP1
REVERT JUMPDEST PUSH2 0x13E PUSH2 0x216 JUMP JUMPDEST
PUSH1 0x40 MLOAD DUP1 DUP3 DUP2 MSTORE PUSH1 0x20 ADD
SWAP2 POP POP PUSH1 0x40 MLOAD DUP1 SWAP2 SUB SWAP1
RETURN JUMPDEST CALLVALUE ISZERO PUSH2 0x15F JUMPI
PUSH1 0x0 DUP1 REVERT JUMPDEST PUSH2 0x167 PUSH2
0x21C JUMP JUMPDEST PUSH1 0x40 MLOAD DUP1 DUP3 DUP2
MSTORE PUSH1 0x20 ADD SWAP2 POP POP PUSH1 0x40 MLOAD
DUP1 SWAP2 SUB SWAP1 RETURN JUMPDEST PUSH1 0x1 SLOAD
DUP2 JUMP JUMPDEST PUSH1 0x0 DUP1 SLOAD SWAP1 POP
SWAP1 JUMP JUMPDEST PUSH1 0x0 PUSH1 0x1 SLOAD SWAP1
POP SWAP1 JUMP JUMPDEST PUSH32
0x410000000000000000000000000000000000000000000000000
0000000000000 DUP2 PUSH1 0x0 NOT AND EQ ISZERO PUSH2
0x1EC JUMPI DUP2 PUSH1 0xFF AND PUSH1 0x0 DUP1 DUP3
DUP3 SLOAD SUB SWAP3 POP POP DUP2 SWAP1 SSTORE POP
DUP2 PUSH1 0xFF AND PUSH1 0x1 PUSH1 0x0 DUP3 DUP3
SLOAD ADD SWAP3 POP POP DUP2 SWAP1 SSTORE POP PUSH2
0x212 JUMP JUMPDEST DUP2 PUSH1 0xFF AND PUSH1 0x0
DUP1 DUP3 DUP3 SLOAD ADD SWAP3 POP POP DUP2 SWAP1
SSTORE POP DUP2 PUSH1 0xFF AND PUSH1 0x1 PUSH1 0x0
DUP3 DUP3 SLOAD SUB SWAP3 POP POP DUP2 SWAP1 SSTORE
POP JUMPDEST POP POP JUMP JUMPDEST PUSH1 0x2 SLOAD
DUP2 JUMP JUMPDEST PUSH1 0x0 SLOAD DUP2 JUMP STOP
LOG1 PUSH6 0x627A7A723058 KECCAK256 NUMBER STATICCALL
DUP7 0xad 0xef DUP9 0xd0 0xd3 SLOAD 0xd5 0xf9 PUSH20
0xB3A6B509863EFD0E1D829D0F66F678D6831450CD STOP 0x29
',
        runtimeBytecode:
'606060405260043610610078576000357c0100000000000000000
00000000000000000000000000000000000000000900463ffffff
f16806339c99d401461007d5780634b2f67b1146100a65780635e
a1ee55146100cf578063aebfe748146100f8578063b69ef8a8146
1012b578063ca34e5f714610154575b600080fd5b341561008857
600080fd5b61009061017d565b60405180828152602001915050 6
```

```
0405180910390f35b34156100b157600080fd5b6100b961018356
5b6040518082815260200191505060405180910390f35b3415610
0da57600080fd5b6100e261018c565b604051808281526020019l
505060405180910390f35b3415610103576000080fd5b610129600
480803560ff169060200190919080356000191690602001909190
5050610196565b005b341561013657600080fd5b61013e6102165
65b6040518082815260200191505060405180910390f35b341561
015f57600080fd5b61016761021c565b60405180828152602001 9
150506040518091039 0f35b60015481565b6000805490509056 5b
600060015490509056 5b7f41000000000000000000000000000000
0000000000000000000000000000008160001916141561 01ec
578160ff16600080828254039250508190555081 60ff166001600
0828254019250508190555061021256 5b8160ff166000808282 54
0192505081905550 8160ff1660016000828254039250508190555
05b5050565b60025481565b600054815600a165627a7a72305820
43fa86adef88d0d354d5f973b3a6b509863efd0e1d829d0f66f67
8d6831450cd0029',
        srcmap: '91:422:0:-;;;
141:3;117:27;;171:3;147:27;;91:422;;;;;;;;;;;;;;',
        srcmapRuntime:
'91:422:0:-;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;
147:27;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;390:59;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;
452;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;200:187;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;
177:19;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;117:27;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;147;;;;:::o;390:59::-;
419:4;436:9;;429:16;;390:59;:::o;452:::-;
481:4;498:9;;491:16;;452:59;:::o;200:187::-;
259:11;:4;:11;;;;256:128;;;290:6;277:19;;:9;;:19;;;;;
;;;;;;
314:6;301:19;;:9;;:19;;;;;;;;;;;256:128;;;349:6;336:1
9;;:9;;:19;;;;;;;;;;;373:6;360:19;;:9;;:19;;;;;;;;;;;
256:128;200:187;;:::o;177:19::-;;;;:::o;
117:27::-;;;;:::o' } },
  errors:
   [ ':8:2: Warning: No visibility specified.
Defaulting to "public".\n\tfunction
rebalancefunds(uint8 amount, bytes32 fund) {\n
^\nSpanning multiple lines.\n',
```

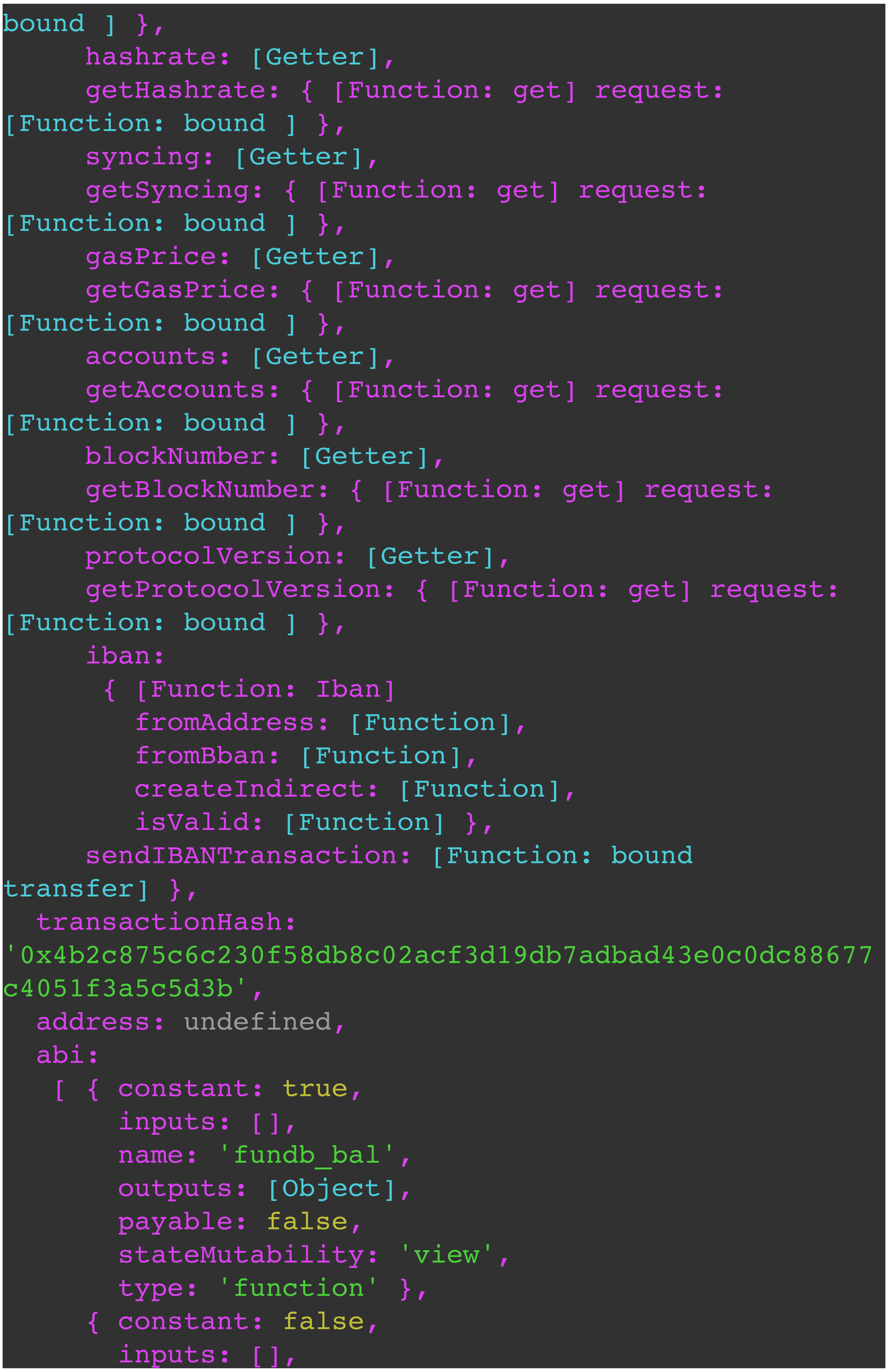


```
     ':18:2: Warning: No visibility specified.
Defaulting to "public".\n\tfunction getfundA()
returns (uint) {\n ^\nSpanning multiple lines.\n',
     ':22:2: Warning: No visibility specified.
Defaulting to "public".\n\tfunction getfundB()
returns (uint) {\n ^\nSpanning multiple lines.\n',
     ':18:2: Warning: Function state mutability can
be restricted to view\n\tfunction getfundA() returns
(uint) {\n ^\nSpanning multiple lines.\n',
     ':22:2: Warning: Function state mutability can
be restricted to view\n\tfunction getfundB() returns
(uint) {\n ^\nSpanning multiple lines.\n' ],
  sourceList: [ '' ],
  sources: { '': { AST: [Object] } } }
> abiDefinition =
JSON.parse(compiledCode.contracts[':FundRebalance'].i
nterface)
[ { constant: true,
    inputs: [],
    name: 'fundb_bal',
    outputs: [ [Object] ],
    payable: false,
    stateMutability: 'view',
    type: 'function' },
  { constant: false,
    inputs: [],
    name: 'getfundA',
    outputs: [ [Object] ],
    payable: false,
    stateMutability: 'nonpayable',
    type: 'function' },
  { constant: false,
    inputs: [],
    name: 'getfundB',
    outputs: [ [Object] ],
    payable: false,
    stateMutability: 'nonpayable',
    type: 'function' },
  { constant: false,
    inputs: [ [Object], [Object] ],
    name: 'rebalancefunds',
    outputs: [],
    payable: false,
    stateMutability: 'nonpayable',
    type: 'function' },
  { constant: true,
    inputs: [],
    name: 'balance',
    outputs: [ [Object] ],
    payable: false,
    stateMutability: 'view',
    type: 'function' },
  { constant: true,
    inputs: [],
    name: 'funda_bal',
    outputs: [ [Object] ],
    payable: false,
    stateMutability: 'view',
    type: 'function' } ]
> fundContract = web3.eth.contract(abiDefinition)
ContractFactory {
  eth:
   Eth {
     _requestManager: RequestManager { provider:
[Object], polls: {}, timeout: null },
     getBalance: { [Function: send] request:
[Function: bound ], call: 'eth_getBalance' },
     getStorageAt: { [Function: send] request:
[Function: bound ], call: 'eth_getStorageAt' },
     getCode: { [Function: send] request: [Function:
bound ], call: 'eth_getCode' },
     getBlock: { [Function: send] request: [Function:
bound ], call: [Function: blockCall] },
     getUncle: { [Function: send] request: [Function:
bound ], call: [Function: uncleCall] },
     getCompilers: { [Function: send] request:
[Function: bound ], call: 'eth_getCompilers' },
     getBlockTransactionCount:
      { [Function: send]
        request: [Function: bound ],
        call: [Function:
getBlockTransactionCountCall] },
     getBlockUncleCount:
      { [Function: send]
        request: [Function: bound ],
        call: [Function: uncleCountCall] },
     getTransaction:
      { [Function: send]
        request: [Function: bound ],
        call: 'eth_getTransactionByHash' },
     getTransactionFromBlock:
      { [Function: send]
        request: [Function: bound ],
        call: [Function: transactionFromBlockCall] },
     getTransactionReceipt:
      { [Function: send]
        request: [Function: bound ],
        call: 'eth_getTransactionReceipt' },
     getTransactionCount: { [Function: send] request:
[Function: bound ], call:
'eth_getTransactionCount' },
     call: { [Function: send] request: [Function:
bound ], call: 'eth_call' },
     estimateGas: { [Function: send] request:
[Function: bound ], call: 'eth_estimateGas' },
     sendRawTransaction: { [Function: send] request:
[Function: bound ], call: 'eth_sendRawTransaction' },
     signTransaction: { [Function: send] request:
[Function: bound ], call: 'eth_signTransaction' },
     sendTransaction: { [Function: send] request:
[Function: bound ], call: 'eth_sendTransaction' },
     sign: { [Function: send] request: [Function:
bound ], call: 'eth_sign' },
     compile: { solidity: [Object], lll: [Object],
serpent: [Object] },
     submitWork: { [Function: send] request:
[Function: bound ], call: 'eth_submitWork' },
     getWork: { [Function: send] request: [Function:
bound ], call: 'eth_getWork' },
     coinbase: [Getter],
     getCoinbase: { [Function: get] request:
[Function: bound ] },
     mining: [Getter],
     getMining: { [Function: get] request: [Function:
bound ] },
     hashrate: [Getter],
     getHashrate: { [Function: get] request:
[Function: bound ] },
     syncing: [Getter],
     getSyncing: { [Function: get] request:
[Function: bound ] },
     gasPrice: [Getter],
     getGasPrice: { [Function: get] request:
[Function: bound ] },
     accounts: [Getter],
     getAccounts: { [Function: get] request:
[Function: bound ] },
     blockNumber: [Getter],
     getBlockNumber: { [Function: get] request:
[Function: bound ] },
     protocolVersion: [Getter],
     getProtocolVersion: { [Function: get] request:
[Function: bound ] },
     iban:
      { [Function: Iban]
        fromAddress: [Function],
        fromBban: [Function],
        createIndirect: [Function],
        isValid: [Function] },
     sendIBANTransaction: [Function: bound
transfer] },
  abi:
   [ { constant: true,
       inputs: [],
       name: 'fundb_bal',
       outputs: [Object],
       payable: false,
       stateMutability: 'view',
       type: 'function' },
     { constant: false,
       inputs: [],
       name: 'getfundA',
       outputs: [Object],
       payable: false,
```



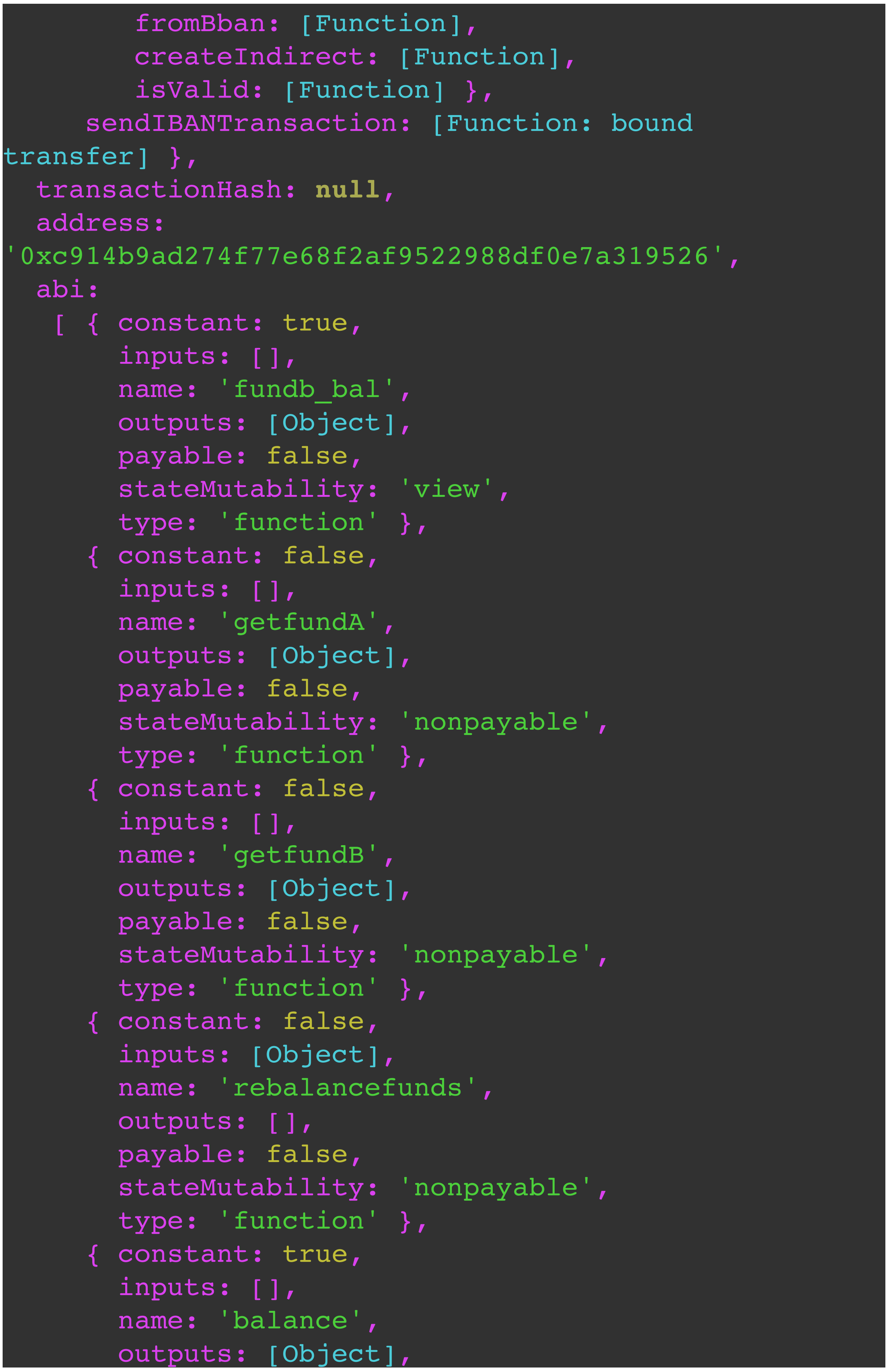
```
        stateMutability: 'nonpayable',
        type: 'function' },
      { constant: false,
        inputs: [],
        name: 'getfundB',
        outputs: [Object],
        payable: false,
        stateMutability: 'nonpayable',
        type: 'function' },
      { constant: false,
        inputs: [Object],
        name: 'rebalancefunds',
        outputs: [],
        payable: false,
        stateMutability: 'nonpayable',
        type: 'function' },
      { constant: true,
        inputs: [],
        name: 'balance',
        outputs: [Object],
        payable: false,
        stateMutability: 'view',
        type: 'function' },
      { constant: true,
        inputs: [],
        name: 'funda_bal',
        outputs: [Object],
        payable: false,
        stateMutability: 'view',
        type: 'function' } ],
  new: { [Function] getData: [Function: bound ] } }
> byteCode =
compiledCode.contracts[':FundRebalance'].bytecode
'6060604052606460005561012c600155341561001a57600080fd
5b61024e806100296000396000f300606060405260043610610007
8576000357c0100000000000000000000000000000000000000000
0000000000000000900463ffffffff16806339c99d401461007d5
780634b2f67b1146100a65780635ea1ee55146100cf578063aebf
e748146100f8578063b69ef8a81461012b578063ca34e5f714610
154575b600080fd5b341561008857600080fd5b61009061017d56
5b6040518082815260200191505060405180910390f35b3415610
```

```
0b157600080fd5b6100b9610183565b6040518082815260200191
5050604051809103 90f35b34156100da57600080fd5b6100e2610
18c565b6040518082815260200191505060405180910390f35b34
156101035760008 0fd5b61012960048080356 0ff1690602001909
190803560001916906020019091905050610196565b005b341561
013657600080fd5b61013e610216565b6040518082815260200 19
1505060405180910390f35b341561015f57600080fd5b61016761
021c565b6040518082815260200191505060405180910390f35b6
0015481565b600080549050905 65b600060015490509056 5b7f41
00000000000000000000000000000000000000000000000000000
0000000008160001916141561 01ec578160ff1660008082825403
925050819055508160ff166001600082825401925050819055506
10212565b8160ff16600080828254019250508190555081 60ff16
600160008282540392505081905550 5b5050565b60025481565b6
00054815600a165627a7a72305820 43fa86adef88d0d354d5f973
b3a6b509863efd0e1d829d0f66f678d6831450cd0029'
> deployedContract = fundContract.new(100,300,{data:
byteCode, from: web3.eth.accounts[0], gas: 4700000})
Contract {
  _eth:
   Eth {
     _requestManager: RequestManager { provider:
[Object], polls: {}, timeout: null },
     getBalance: { [Function: send] request:
[Function: bound ], call: 'eth_getBalance' },
     getStorageAt: { [Function: send] request:
[Function: bound ], call: 'eth_getStorageAt' },
     getCode: { [Function: send] request: [Function:
bound ], call: 'eth_getCode' },
     getBlock: { [Function: send] request: [Function:
bound ], call: [Function: blockCall] },
     getUncle: { [Function: send] request: [Function:
bound ], call: [Function: uncleCall] },
     getCompilers: { [Function: send] request:
[Function: bound ], call: 'eth_getCompilers' },
     getBlockTransactionCount:
      { [Function: send]
        request: [Function: bound ],
        call: [Function:
getBlockTransactionCountCall] },
     getBlockUncleCount:
      { [Function: send]
        request: [Function: bound ],
        call: [Function: uncleCountCall] },
     getTransaction:
      { [Function: send]
        request: [Function: bound ],
        call: 'eth_getTransactionByHash' },
     getTransactionFromBlock:
      { [Function: send]
        request: [Function: bound ],
        call: [Function: transactionFromBlockCall] },
     getTransactionReceipt:
      { [Function: send]
        request: [Function: bound ],
        call: 'eth_getTransactionReceipt' },
     getTransactionCount: { [Function: send] request:
[Function: bound ], call:
'eth_getTransactionCount' },
     call: { [Function: send] request: [Function:
bound ], call: 'eth_call' },
     estimateGas: { [Function: send] request:
[Function: bound ], call: 'eth_estimateGas' },
     sendRawTransaction: { [Function: send] request:
[Function: bound ], call: 'eth_sendRawTransaction' },
     signTransaction: { [Function: send] request:
[Function: bound ], call: 'eth_signTransaction' },
     sendTransaction: { [Function: send] request:
[Function: bound ], call: 'eth_sendTransaction' },
     sign: { [Function: send] request: [Function:
bound ], call: 'eth_sign' },
     compile: { solidity: [Object], lll: [Object],
serpent: [Object] },
     submitWork: { [Function: send] request:
[Function: bound ], call: 'eth_submitWork' },
     getWork: { [Function: send] request: [Function:
bound ], call: 'eth_getWork' },
     coinbase: [Getter],
     getCoinbase: { [Function: get] request:
[Function: bound ] },
     mining: [Getter],
     getMining: { [Function: get] request: [Function:
```

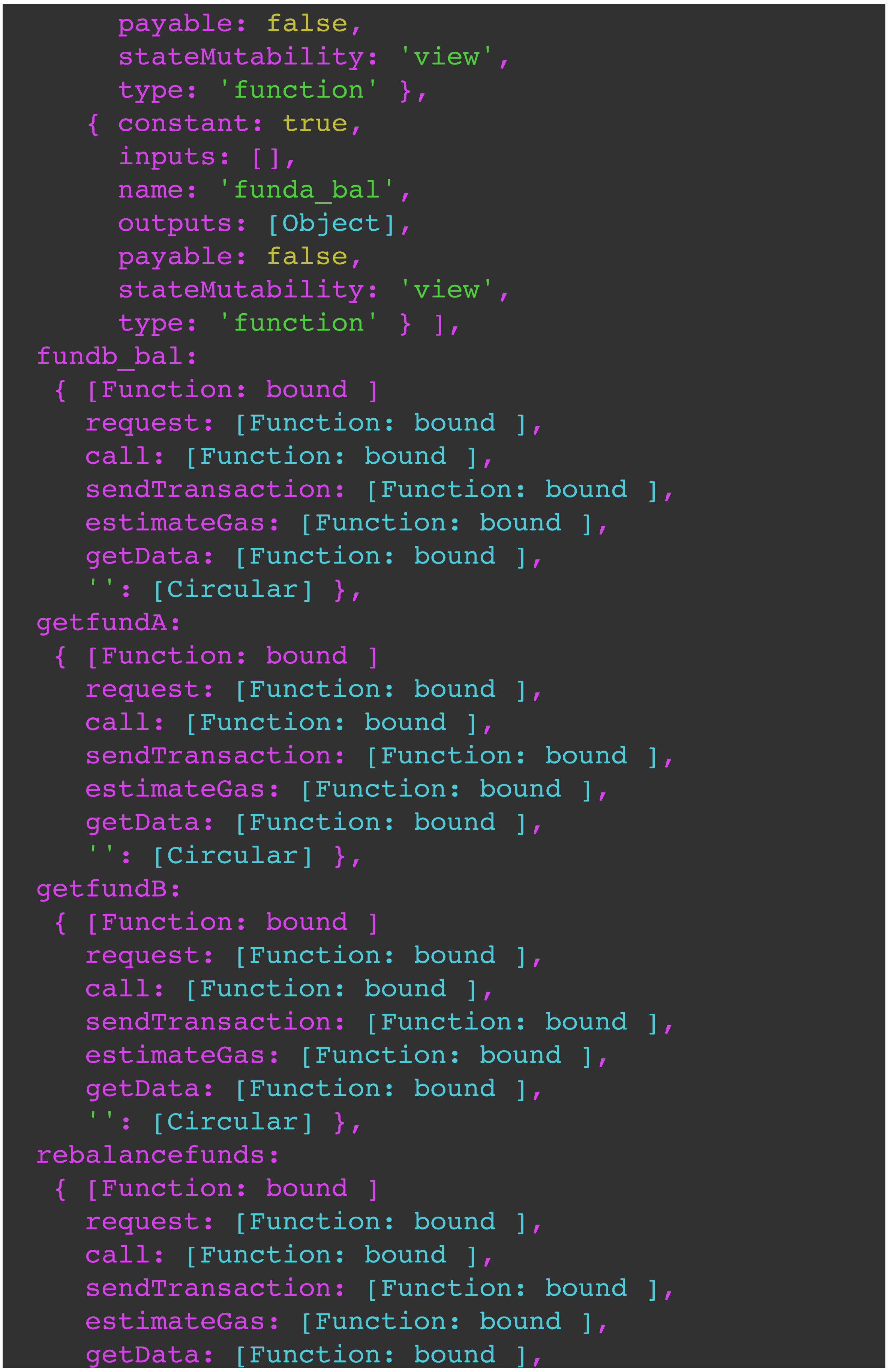


```
bound ] },
     hashrate: [Getter],
     getHashrate: { [Function: get] request:
[Function: bound ] },
     syncing: [Getter],
     getSyncing: { [Function: get] request:
[Function: bound ] },
     gasPrice: [Getter],
     getGasPrice: { [Function: get] request:
[Function: bound ] },
     accounts: [Getter],
     getAccounts: { [Function: get] request:
[Function: bound ] },
     blockNumber: [Getter],
     getBlockNumber: { [Function: get] request:
[Function: bound ] },
     protocolVersion: [Getter],
     getProtocolVersion: { [Function: get] request:
[Function: bound ] },
     iban:
      { [Function: Iban]
        fromAddress: [Function],
        fromBban: [Function],
        createIndirect: [Function],
        isValid: [Function] },
     sendIBANTransaction: [Function: bound
transfer] },
  transactionHash:
'0x4b2c875c6c230f58db8c02acf3d19db7adbad43e0c0dc88677
c4051f3a5c5d3b',
  address: undefined,
  abi:
   [ { constant: true,
       inputs: [],
       name: 'fundb_bal',
       outputs: [Object],
       payable: false,
       stateMutability: 'view',
       type: 'function' },
     { constant: false,
       inputs: [],
       name: 'getfundA',
       outputs: [Object],
       payable: false,
       stateMutability: 'nonpayable',
       type: 'function' },
     { constant: false,
       inputs: [],
       name: 'getfundB',
       outputs: [Object],
       payable: false,
       stateMutability: 'nonpayable',
       type: 'function' },
     { constant: false,
       inputs: [Object],
       name: 'rebalancefunds',
       outputs: [],
       payable: false,
       stateMutability: 'nonpayable',
       type: 'function' },
     { constant: true,
       inputs: [],
       name: 'balance',
       outputs: [Object],
       payable: false,
       stateMutability: 'view',
       type: 'function' },
     { constant: true,
       inputs: [],
       name: 'funda_bal',
       outputs: [Object],
       payable: false,
       stateMutability: 'view',
       type: 'function' } ] }
> deployedContract.address
'0xc914b9ad274f77e68f2af9522988df0e7a319526'
> contractInstance =
fundContract.at(deployedContract.address)
Contract {
  _eth:
   Eth {
     _requestManager: RequestManager { provider:
```

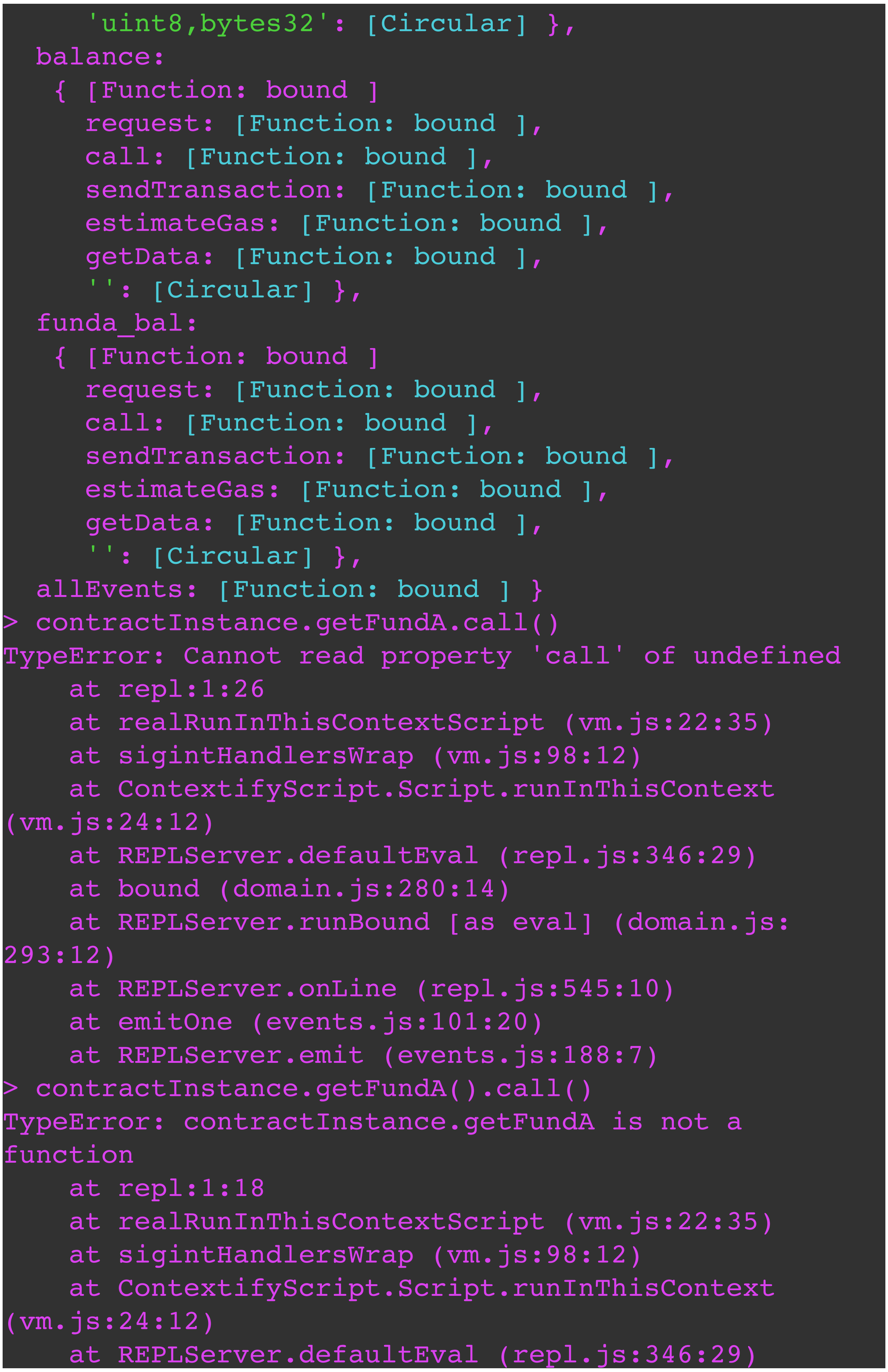
```
[Object], polls: {}, timeout: null },
     getBalance: { [Function: send] request:
[Function: bound ], call: 'eth_getBalance' },
     getStorageAt: { [Function: send] request:
[Function: bound ], call: 'eth_getStorageAt' },
     getCode: { [Function: send] request: [Function:
bound ], call: 'eth_getCode' },
     getBlock: { [Function: send] request: [Function:
bound ], call: [Function: blockCall] },
     getUncle: { [Function: send] request: [Function:
bound ], call: [Function: uncleCall] },
     getCompilers: { [Function: send] request:
[Function: bound ], call: 'eth_getCompilers' },
     getBlockTransactionCount:
      { [Function: send]
        request: [Function: bound ],
        call: [Function:
getBlockTransactionCountCall] },
     getBlockUncleCount:
      { [Function: send]
        request: [Function: bound ],
        call: [Function: uncleCountCall] },
     getTransaction:
      { [Function: send]
        request: [Function: bound ],
        call: 'eth_getTransactionByHash' },
     getTransactionFromBlock:
      { [Function: send]
        request: [Function: bound ],
        call: [Function: transactionFromBlockCall] },
     getTransactionReceipt:
      { [Function: send]
        request: [Function: bound ],
        call: 'eth_getTransactionReceipt' },
     getTransactionCount: { [Function: send] request:
[Function: bound ], call:
'eth_getTransactionCount' },
     call: { [Function: send] request: [Function:
bound ], call: 'eth_call' },
     estimateGas: { [Function: send] request:
[Function: bound ], call: 'eth_estimateGas' },
     sendRawTransaction: { [Function: send] request:
[Function: bound ], call: 'eth_sendRawTransaction' },
     signTransaction: { [Function: send] request:
[Function: bound ], call: 'eth_signTransaction' },
     sendTransaction: { [Function: send] request:
[Function: bound ], call: 'eth_sendTransaction' },
     sign: { [Function: send] request: [Function:
bound ], call: 'eth_sign' },
     compile: { solidity: [Object], lll: [Object],
serpent: [Object] },
     submitWork: { [Function: send] request:
[Function: bound ], call: 'eth_submitWork' },
     getWork: { [Function: send] request: [Function:
bound ], call: 'eth_getWork' },
     coinbase: [Getter],
     getCoinbase: { [Function: get] request:
[Function: bound ] },
     mining: [Getter],
     getMining: { [Function: get] request: [Function:
bound ] },
     hashrate: [Getter],
     getHashrate: { [Function: get] request:
[Function: bound ] },
     syncing: [Getter],
     getSyncing: { [Function: get] request:
[Function: bound ] },
     gasPrice: [Getter],
     getGasPrice: { [Function: get] request:
[Function: bound ] },
     accounts: [Getter],
     getAccounts: { [Function: get] request:
[Function: bound ] },
     blockNumber: [Getter],
     getBlockNumber: { [Function: get] request:
[Function: bound ] },
     protocolVersion: [Getter],
     getProtocolVersion: { [Function: get] request:
[Function: bound ] },
     iban:
      { [Function: Iban]
        fromAddress: [Function],
```



```
        fromBban: [Function],
        createIndirect: [Function],
        isValid: [Function] },
     sendIBANTransaction: [Function: bound
transfer] },
  transactionHash: null,
  address:
'0xc914b9ad274f77e68f2af9522988df0e7a319526',
  abi:
   [ { constant: true,
       inputs: [],
       name: 'fundb_bal',
       outputs: [Object],
       payable: false,
       stateMutability: 'view',
       type: 'function' },
     { constant: false,
       inputs: [],
       name: 'getfundA',
       outputs: [Object],
       payable: false,
       stateMutability: 'nonpayable',
       type: 'function' },
     { constant: false,
       inputs: [],
       name: 'getfundB',
       outputs: [Object],
       payable: false,
       stateMutability: 'nonpayable',
       type: 'function' },
     { constant: false,
       inputs: [Object],
       name: 'rebalancefunds',
       outputs: [],
       payable: false,
       stateMutability: 'nonpayable',
       type: 'function' },
     { constant: true,
       inputs: [],
       name: 'balance',
       outputs: [Object],
```



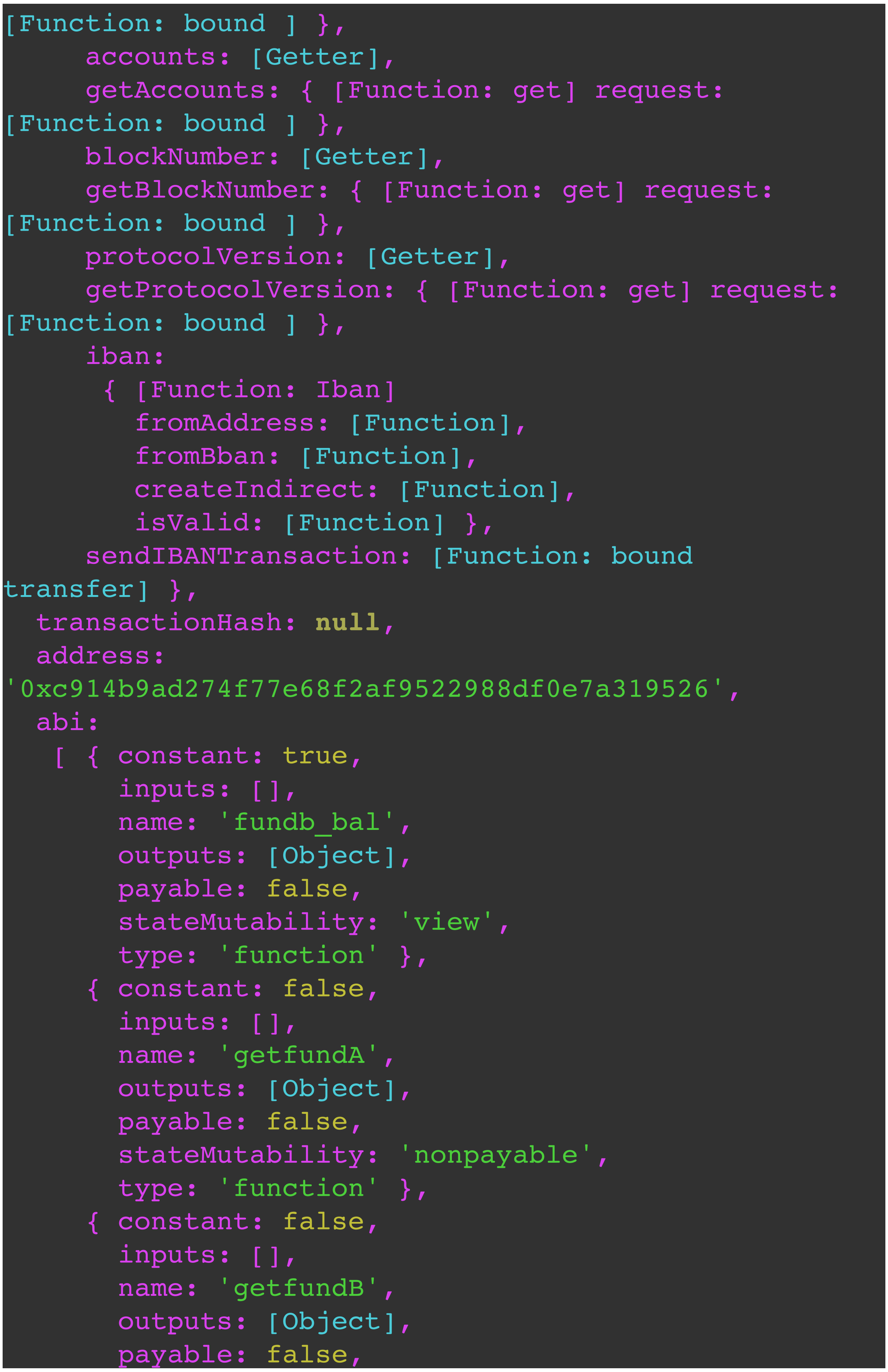
```
       payable: false,
       stateMutability: 'view',
       type: 'function' },
     { constant: true,
       inputs: [],
       name: 'funda_bal',
       outputs: [Object],
       payable: false,
       stateMutability: 'view',
       type: 'function' } ],
  fundb_bal:
   { [Function: bound ]
     request: [Function: bound ],
     call: [Function: bound ],
     sendTransaction: [Function: bound ],
     estimateGas: [Function: bound ],
     getData: [Function: bound ],
     '': [Circular] },
  getfundA:
   { [Function: bound ]
     request: [Function: bound ],
     call: [Function: bound ],
     sendTransaction: [Function: bound ],
     estimateGas: [Function: bound ],
     getData: [Function: bound ],
     '': [Circular] },
  getfundB:
   { [Function: bound ]
     request: [Function: bound ],
     call: [Function: bound ],
     sendTransaction: [Function: bound ],
     estimateGas: [Function: bound ],
     getData: [Function: bound ],
     '': [Circular] },
  rebalancefunds:
   { [Function: bound ]
     request: [Function: bound ],
     call: [Function: bound ],
     sendTransaction: [Function: bound ],
     estimateGas: [Function: bound ],
     getData: [Function: bound ],
```



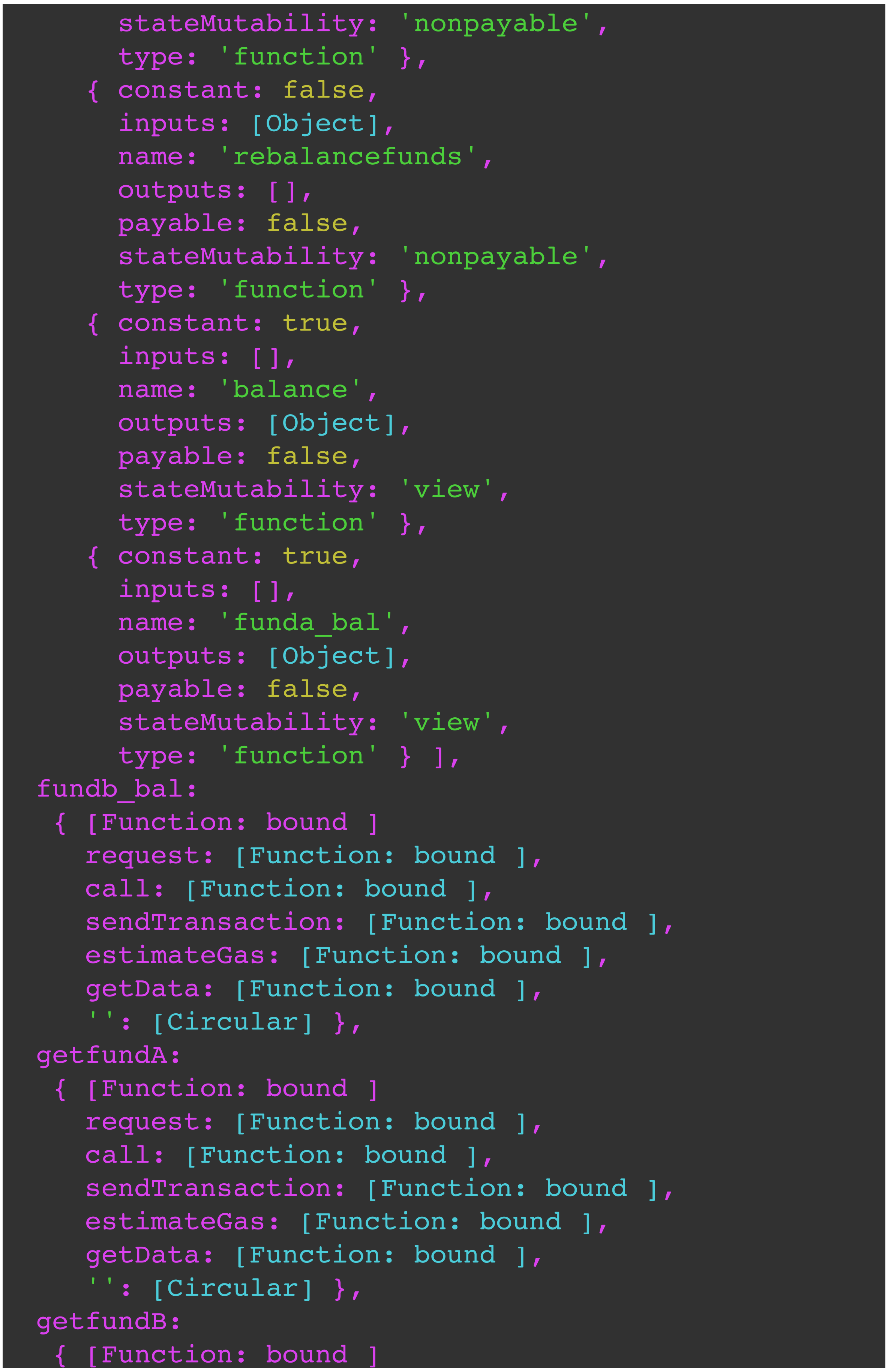
```
     'uint8,bytes32': [Circular] },
  balance:
   { [Function: bound ]
     request: [Function: bound ],
     call: [Function: bound ],
     sendTransaction: [Function: bound ],
     estimateGas: [Function: bound ],
     getData: [Function: bound ],
     '': [Circular] },
  funda_bal:
   { [Function: bound ]
     request: [Function: bound ],
     call: [Function: bound ],
     sendTransaction: [Function: bound ],
     estimateGas: [Function: bound ],
     getData: [Function: bound ],
     '': [Circular] },
  allEvents: [Function: bound ] }
> contractInstance.getFundA.call()
TypeError: Cannot read property 'call' of undefined
    at repl:1:26
    at realRunInThisContextScript (vm.js:22:35)
    at sigintHandlersWrap (vm.js:98:12)
    at ContextifyScript.Script.runInThisContext
(vm.js:24:12)
    at REPLServer.defaultEval (repl.js:346:29)
    at bound (domain.js:280:14)
    at REPLServer.runBound [as eval] (domain.js:
293:12)
    at REPLServer.onLine (repl.js:545:10)
    at emitOne (events.js:101:20)
    at REPLServer.emit (events.js:188:7)
> contractInstance.getFundA().call()
TypeError: contractInstance.getFundA is not a
function
    at repl:1:18
    at realRunInThisContextScript (vm.js:22:35)
    at sigintHandlersWrap (vm.js:98:12)
    at ContextifyScript.Script.runInThisContext
(vm.js:24:12)
    at REPLServer.defaultEval (repl.js:346:29)
    at bound (domain.js:280:14)
    at REPLServer.runBound [as eval] (domain.js:
293:12)
    at REPLServer.onLine (repl.js:545:10)
    at emitOne (events.js:101:20)
    at REPLServer.emit (events.js:188:7)
> contractInstance.getfundA().call()
Error: invalid address
    at inputAddressFormatter (/Users/srishti/
node_modules/web3/lib/web3/formatters.js:273:11)
    at inputTransactionFormatter (/Users/srishti/
node_modules/web3/lib/web3/formatters.js:99:20)
    at /Users/srishti/node_modules/web3/lib/web3/
method.js:89:28
    at Array.map (native)
    at Method.formatInput (/Users/srishti/
node_modules/web3/lib/web3/method.js:88:32)
    at Method.toPayload (/Users/srishti/node_modules/
web3/lib/web3/method.js:114:23)
    at Eth.send [as sendTransaction] (/Users/srishti/
node_modules/web3/lib/web3/method.js:139:30)
    at SolidityFunction.sendTransaction (/Users/
srishti/node_modules/web3/lib/web3/function.js:
170:26)
    at SolidityFunction.execute (/Users/srishti/
node_modules/web3/lib/web3/function.js:256:37)
    at repl:1:18
> contractInstance.getfundA([],{from:
web3.eth.accounts[0]})
Error: Invalid number of arguments to Solidity
function
    at Object.InvalidNumberOfSolidityArgs (/Users/
srishti/node_modules/web3/lib/web3/errors.js:25:16)
    at SolidityFunction.validateArgs (/Users/srishti/
node_modules/web3/lib/web3/function.js:74:22)
    at SolidityFunction.toPayload (/Users/srishti/
node_modules/web3/lib/web3/function.js:90:10)
    at SolidityFunction.sendTransaction (/Users/
srishti/node_modules/web3/lib/web3/function.js:
163:24)
    at SolidityFunction.execute (/Users/srishti/
node_modules/web3/lib/web3/function.js:256:37)
    at repl:1:18
    at realRunInThisContextScript (vm.js:22:35)
    at sigintHandlersWrap (vm.js:98:12)
    at ContextifyScript.Script.runInThisContext
(vm.js:24:12)
    at REPLServer.defaultEval (repl.js:346:29)
> contractInstance.getfundA({from:
web3.eth.accounts[0]})
'0xbe925b63632ebabd5b4ac7bb59153583e73b1c3515eff17756
567af278e843a9'
> a = contractInstance.getfundA({from:
web3.eth.accounts[0]})
'0xd7ac6f00b29b89afc57f8c302f8de92780704b2f54b6abf85a
8e82879ce8f880'
> a
'0xd7ac6f00b29b89afc57f8c302f8de92780704b2f54b6abf85a
8e82879ce8f880'
> contractInstance.rebalanceFunds('A', {from:
web3.eth.accounts[0]})
TypeError: contractInstance.rebalanceFunds is not a
function
    at repl:1:18
    at realRunInThisContextScript (vm.js:22:35)
    at sigintHandlersWrap (vm.js:98:12)
    at ContextifyScript.Script.runInThisContext
(vm.js:24:12)
    at REPLServer.defaultEval (repl.js:346:29)
    at bound (domain.js:280:14)
    at REPLServer.runBound [as eval] (domain.js:
293:12)
    at REPLServer.onLine (repl.js:545:10)
    at emitOne (events.js:101:20)
    at REPLServer.emit (events.js:188:7)
> contractInstamce
ReferenceError: contractInstamce is not defined
    at repl:1:1
    at realRunInThisContextScript (vm.js:22:35)
    at sigintHandlersWrap (vm.js:98:12)
    at ContextifyScript.Script.runInThisContext
(vm.js:24:12)
```

```
    at REPLServer.defaultEval (repl.js:346:29)
    at bound (domain.js:280:14)
    at REPLServer.runBound [as eval] (domain.js:
293:12)
    at REPLServer.onLine (repl.js:545:10)
    at emitOne (events.js:101:20)
    at REPLServer.emit (events.js:188:7)
> contractInstance =
fundContract.at(deployedContract.address)
Contract {
  _eth:
   Eth {
     _requestManager: RequestManager { provider:
[Object], polls: {}, timeout: null },
     getBalance: { [Function: send] request:
[Function: bound ], call: 'eth_getBalance' },
     getStorageAt: { [Function: send] request:
[Function: bound ], call: 'eth_getStorageAt' },
     getCode: { [Function: send] request: [Function:
bound ], call: 'eth_getCode' },
     getBlock: { [Function: send] request: [Function:
bound ], call: [Function: blockCall] },
     getUncle: { [Function: send] request: [Function:
bound ], call: [Function: uncleCall] },
     getCompilers: { [Function: send] request:
[Function: bound ], call: 'eth_getCompilers' },
     getBlockTransactionCount:
      { [Function: send]
        request: [Function: bound ],
        call: [Function:
getBlockTransactionCountCall] },
     getBlockUncleCount:
      { [Function: send]
        request: [Function: bound ],
        call: [Function: uncleCountCall] },
     getTransaction:
      { [Function: send]
        request: [Function: bound ],
        call: 'eth_getTransactionByHash' },
     getTransactionFromBlock:
      { [Function: send]
```

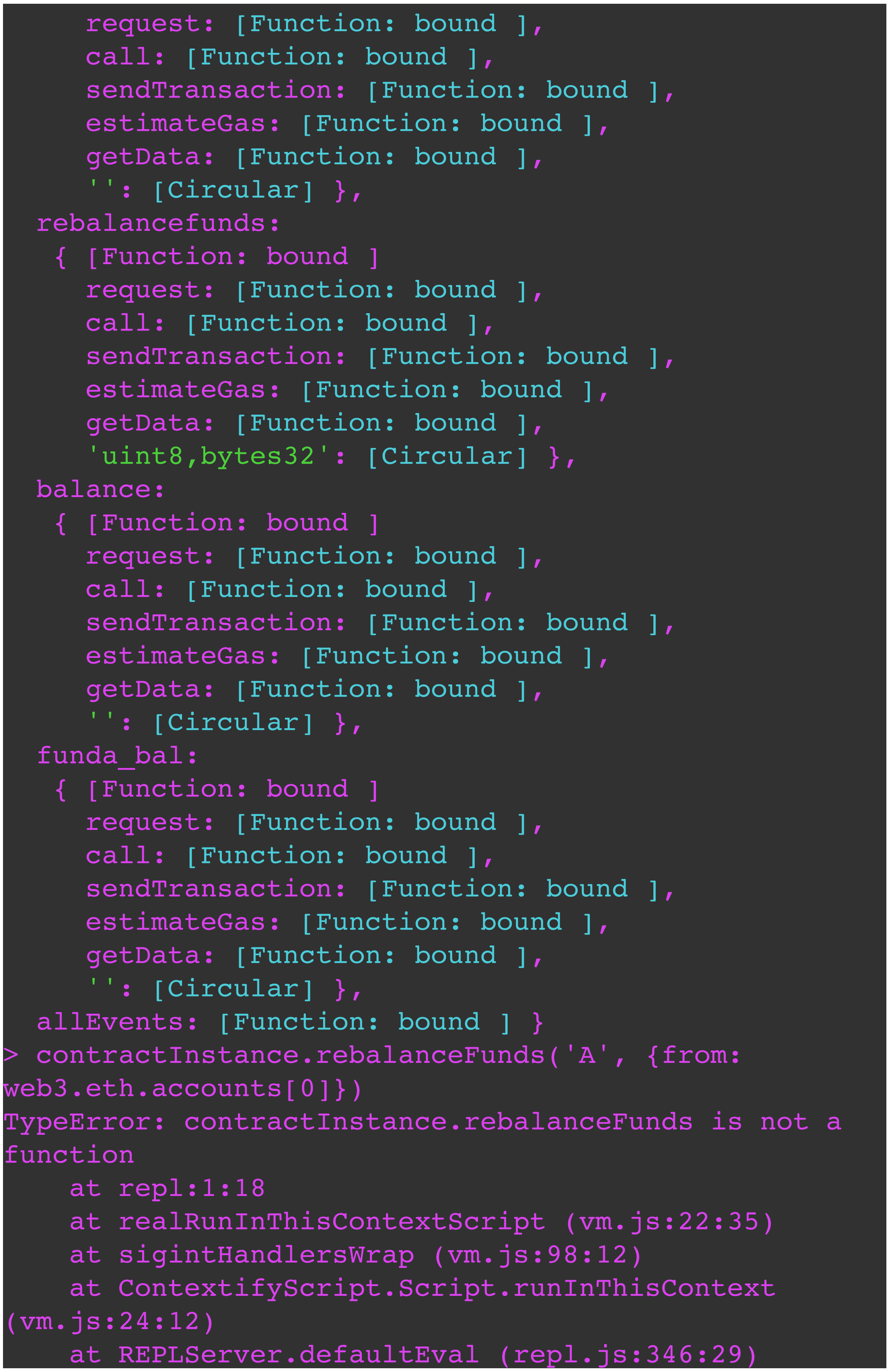
```
        request: [Function: bound ],
        call: [Function: transactionFromBlockCall] },
     getTransactionReceipt:
      { [Function: send]
        request: [Function: bound ],
        call: 'eth_getTransactionReceipt' },
     getTransactionCount: { [Function: send] request:
[Function: bound ], call:
'eth_getTransactionCount' },
     call: { [Function: send] request: [Function:
bound ], call: 'eth_call' },
     estimateGas: { [Function: send] request:
[Function: bound ], call: 'eth_estimateGas' },
     sendRawTransaction: { [Function: send] request:
[Function: bound ], call: 'eth_sendRawTransaction' },
     signTransaction: { [Function: send] request:
[Function: bound ], call: 'eth_signTransaction' },
     sendTransaction: { [Function: send] request:
[Function: bound ], call: 'eth_sendTransaction' },
     sign: { [Function: send] request: [Function:
bound ], call: 'eth_sign' },
     compile: { solidity: [Object], lll: [Object],
serpent: [Object] },
     submitWork: { [Function: send] request:
[Function: bound ], call: 'eth_submitWork' },
     getWork: { [Function: send] request: [Function:
bound ], call: 'eth_getWork' },
     coinbase: [Getter],
     getCoinbase: { [Function: get] request:
[Function: bound ] },
     mining: [Getter],
     getMining: { [Function: get] request: [Function:
bound ] },
     hashrate: [Getter],
     getHashrate: { [Function: get] request:
[Function: bound ] },
     syncing: [Getter],
     getSyncing: { [Function: get] request:
[Function: bound ] },
     gasPrice: [Getter],
     getGasPrice: { [Function: get] request:
```



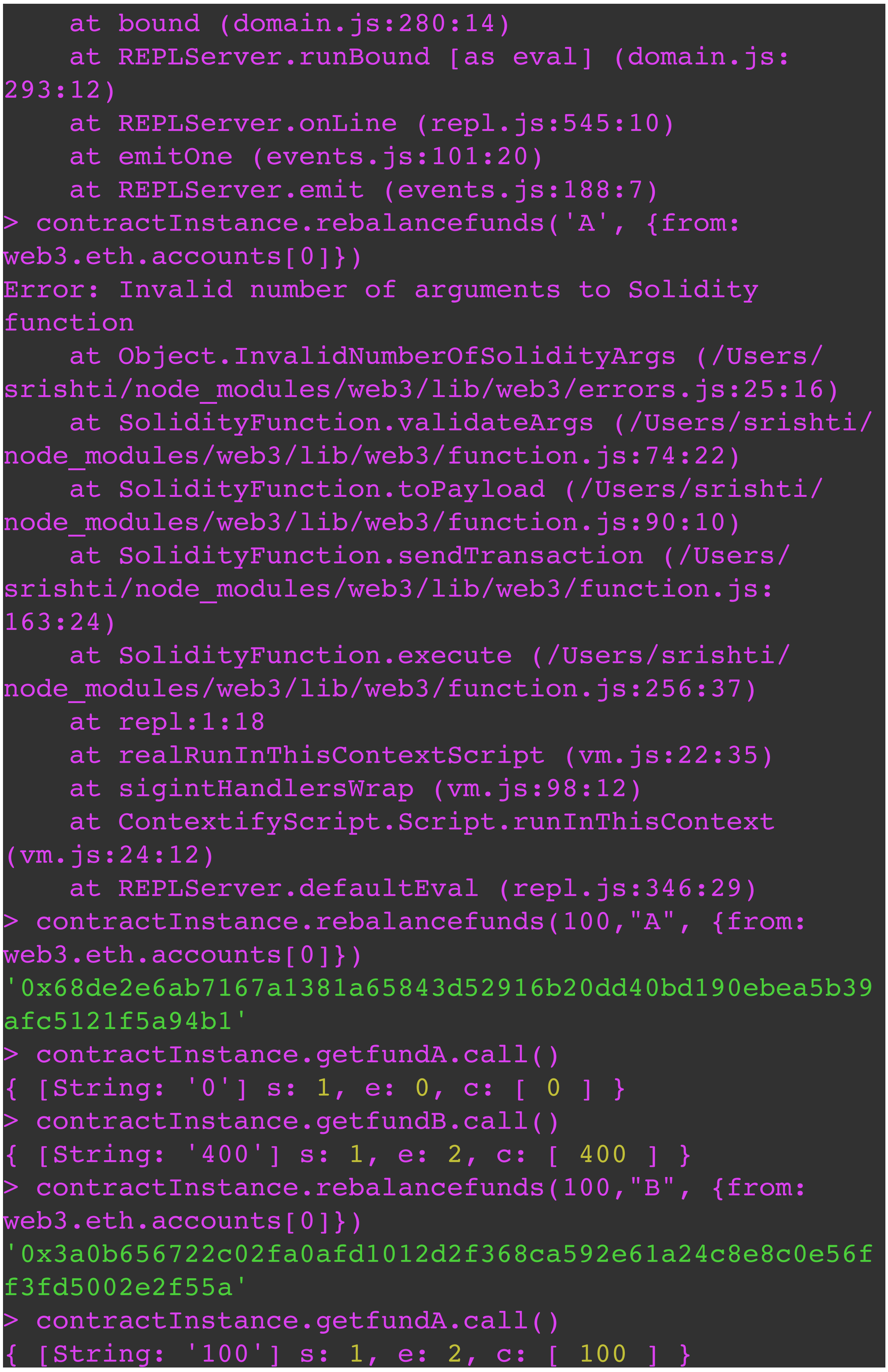
```
[Function: bound ] },
     accounts: [Getter],
     getAccounts: { [Function: get] request:
[Function: bound ] },
     blockNumber: [Getter],
     getBlockNumber: { [Function: get] request:
[Function: bound ] },
     protocolVersion: [Getter],
     getProtocolVersion: { [Function: get] request:
[Function: bound ] },
     iban:
      { [Function: Iban]
        fromAddress: [Function],
        fromBban: [Function],
        createIndirect: [Function],
        isValid: [Function] },
     sendIBANTransaction: [Function: bound
transfer] },
  transactionHash: null,
  address:
'0xc914b9ad274f77e68f2af9522988df0e7a319526',
  abi:
   [ { constant: true,
       inputs: [],
       name: 'fundb_bal',
       outputs: [Object],
       payable: false,
       stateMutability: 'view',
       type: 'function' },
     { constant: false,
       inputs: [],
       name: 'getfundA',
       outputs: [Object],
       payable: false,
       stateMutability: 'nonpayable',
       type: 'function' },
     { constant: false,
       inputs: [],
       name: 'getfundB',
       outputs: [Object],
       payable: false,
```



```
       stateMutability: 'nonpayable',
       type: 'function' },
     { constant: false,
       inputs: [Object],
       name: 'rebalancefunds',
       outputs: [],
       payable: false,
       stateMutability: 'nonpayable',
       type: 'function' },
     { constant: true,
       inputs: [],
       name: 'balance',
       outputs: [Object],
       payable: false,
       stateMutability: 'view',
       type: 'function' },
     { constant: true,
       inputs: [],
       name: 'funda_bal',
       outputs: [Object],
       payable: false,
       stateMutability: 'view',
       type: 'function' } ],
  fundb_bal:
   { [Function: bound ]
     request: [Function: bound ],
     call: [Function: bound ],
     sendTransaction: [Function: bound ],
     estimateGas: [Function: bound ],
     getData: [Function: bound ],
     '': [Circular] },
  getfundA:
   { [Function: bound ]
     request: [Function: bound ],
     call: [Function: bound ],
     sendTransaction: [Function: bound ],
     estimateGas: [Function: bound ],
     getData: [Function: bound ],
     '': [Circular] },
  getfundB:
   { [Function: bound ]
```



```
     request: [Function: bound ],
     call: [Function: bound ],
     sendTransaction: [Function: bound ],
     estimateGas: [Function: bound ],
     getData: [Function: bound ],
     '': [Circular] },
  rebalancefunds:
   { [Function: bound ]
     request: [Function: bound ],
     call: [Function: bound ],
     sendTransaction: [Function: bound ],
     estimateGas: [Function: bound ],
     getData: [Function: bound ],
     'uint8,bytes32': [Circular] },
  balance:
   { [Function: bound ]
     request: [Function: bound ],
     call: [Function: bound ],
     sendTransaction: [Function: bound ],
     estimateGas: [Function: bound ],
     getData: [Function: bound ],
     '': [Circular] },
  funda_bal:
   { [Function: bound ]
     request: [Function: bound ],
     call: [Function: bound ],
     sendTransaction: [Function: bound ],
     estimateGas: [Function: bound ],
     getData: [Function: bound ],
     '': [Circular] },
  allEvents: [Function: bound ] }
> contractInstance.rebalanceFunds('A', {from:
web3.eth.accounts[0]})
TypeError: contractInstance.rebalanceFunds is not a
function
    at repl:1:18
    at realRunInThisContextScript (vm.js:22:35)
    at sigintHandlersWrap (vm.js:98:12)
    at ContextifyScript.Script.runInThisContext
(vm.js:24:12)
    at REPLServer.defaultEval (repl.js:346:29)
```



```
    at bound (domain.js:280:14)
    at REPLServer.runBound [as eval] (domain.js:
293:12)
    at REPLServer.onLine (repl.js:545:10)
    at emitOne (events.js:101:20)
    at REPLServer.emit (events.js:188:7)
> contractInstance.rebalancefunds('A', {from:
web3.eth.accounts[0]})
Error: Invalid number of arguments to Solidity
function
    at Object.InvalidNumberOfSolidityArgs (/Users/
srishti/node_modules/web3/lib/web3/errors.js:25:16)
    at SolidityFunction.validateArgs (/Users/srishti/
node_modules/web3/lib/web3/function.js:74:22)
    at SolidityFunction.toPayload (/Users/srishti/
node_modules/web3/lib/web3/function.js:90:10)
    at SolidityFunction.sendTransaction (/Users/
srishti/node_modules/web3/lib/web3/function.js:
163:24)
    at SolidityFunction.execute (/Users/srishti/
node_modules/web3/lib/web3/function.js:256:37)
    at repl:1:18
    at realRunInThisContextScript (vm.js:22:35)
    at sigintHandlersWrap (vm.js:98:12)
    at ContextifyScript.Script.runInThisContext
(vm.js:24:12)
    at REPLServer.defaultEval (repl.js:346:29)
> contractInstance.rebalancefunds(100,"A", {from:
web3.eth.accounts[0]})
'0x68de2e6ab7167a1381a65843d52916b20dd40bd190ebea5b39
afc5121f5a94b1'
> contractInstance.getfundA.call()
{ [String: '0'] s: 1, e: 0, c: [ 0 ] }
> contractInstance.getfundB.call()
{ [String: '400'] s: 1, e: 2, c: [ 400 ] }
> contractInstance.rebalancefunds(100,"B", {from:
web3.eth.accounts[0]})
'0x3a0b656722c02fa0afd1012d2f368ca592e61a24c8e8c0e56f
f3fd5002e2f55a'
> contractInstance.getfundA.call()
{ [String: '100'] s: 1, e: 2, c: [ 100 ] }
```

```
> contractInstance.getfundB.call()
{ [String: '300'] s: 1, e: 2, c: [ 300 ] }
> contractInstance.getfundA.call().toString()
'100'
```