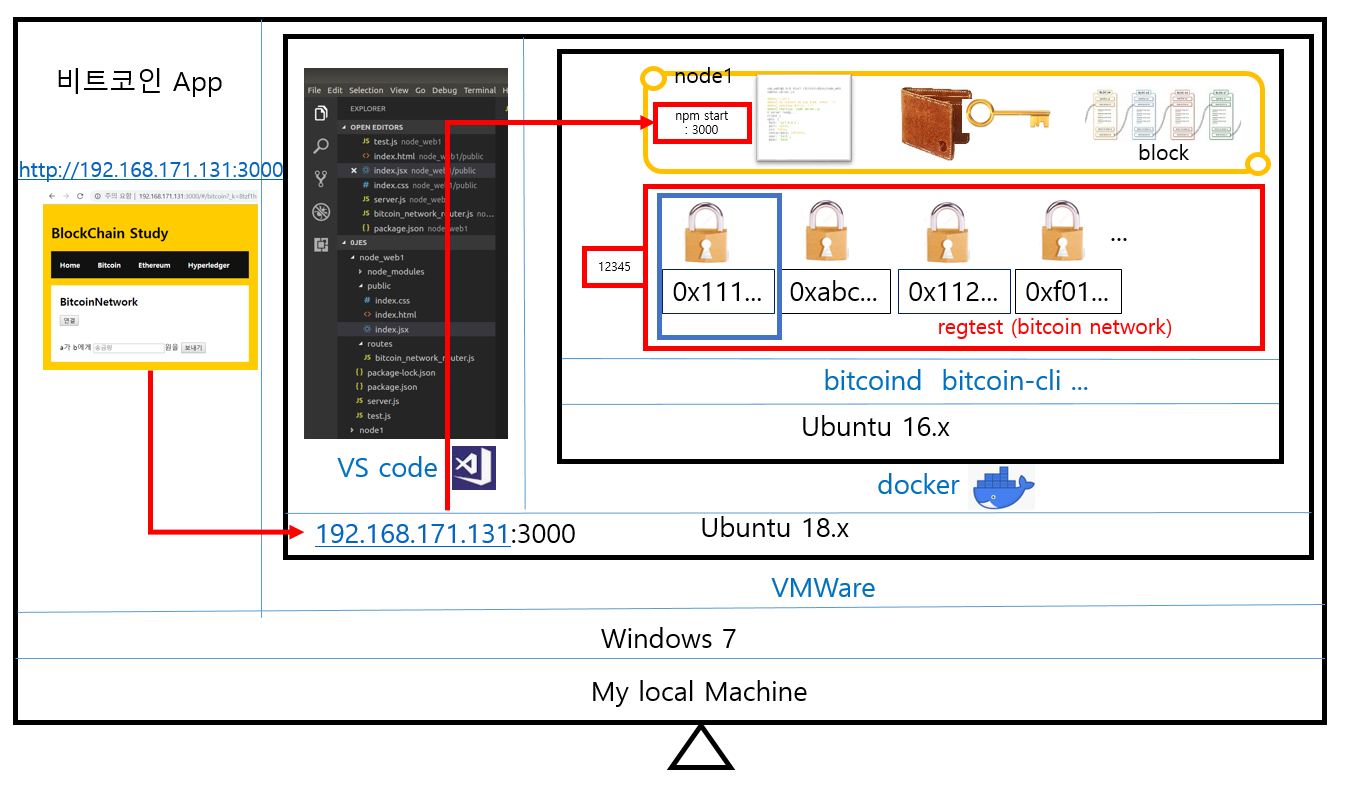
**dock React + Node + Bitcoin**



1. Docker 설치

|  |
| --- |
| 내컴>vmware>ubuntu18> sudo apt update (🡺 패키지 리스트가 최신인지 확인하는 것)  내컴>vmware>ubuntu18> sudo apt upgrade (🡺 최신 패키지로 다운 받는 것)  내컴>vmware>ubuntu18> sudo apt install docker.io (🡺 도커 설치)  내컴>vmware>ubuntu18> docker –v (🡺 18.09.7 ) |

1. 블록체인 네트웍이미지 받아 실행하기

|  |
| --- |
| 내컴>vmware>ubuntu18> sudo docker pull pjt3591oo/bitcoin:0.17.01 (🡺 블록체인에 연결하기 위한 클라이언트 프로그램이 설치된 도커 컨테이너를 다운 받는 것)  내컴>vmware>ubuntu18> sudo docker images  내컴>vmware>ubuntu18> mkdir 0jes  내컴>vmware>ubuntu18> cd 0jes  내컴>vmware>ubuntu18> sudo docker run -it --name jes\_bit -p 3000:3000 -v $(pwd):/bitcoin/0jes pjt3591oo/bitcoin:0.17.01 bash (🡺 컨테이너 가동/접속)  root@1060e4b3d645:/#  root@1060e4b3d645:/# cd ~  root@1060e4b3d645:~# ls (🡺start.sh보임)  root@1060e4b3d645:~# ~/start.sh |

1. 다른 터미널을 열어 jes\_bit 컨테이너에 접속한다

|  |
| --- |
| 내컴>vmware>ubuntu18> sudo docker exec -it jes\_bit bash  root@1060e4b3d645:/#  root@1060e4b3d645:/# cd ~  root@1060e4b3d645:~# cd .bitcoin  root@1060e4b3d645:~# ll (🡺 blocks 와 regtest 가 보임) |

1. 실행중인 bitcoind를 종료하고 새로운 노드를 구성해 본다

|  |
| --- |
| root@1060e4b3d645:~# mkdir node1  root@1060e4b3d645:~# **bitcoind** -regtest -rpcuser=test -rpcpassword=test -server -rpcport=12345 -datadir="$PWD/node1" -deprecatedrpc=accounts (🡺 개인용 비트코인 네트워크가 가동됨) |

1. 다른 터미널에서 bitcoin-cli를 실행한다

|  |
| --- |
| root@1060e4b3d645:~# **bitcoin-cli** -regtest -rpcuser=test -rpcpassword=test -rpcconnect="127.0.0.1" -rpcport=12345 -datadir="$PWD/node1" getnewaddress acc1 (🡺 4번 네트워크에 접속하여 새 acc1이라는 이름의 새 계정을 만듦)  root@1060e4b3d645:~# **bitcoin-cli** -regtest -rpcuser=test -rpcpassword=test -rpcconnect="127.0.0.1" -rpcport=12345 -datadir="$PWD/node1" getnewaddress acc2 (🡺 4번 네트워크에 접속하여 새 acc2이라는 이름의 새 계정을 만듦)  //계정 확인  root@1060e4b3d645:~# **bitcoin-cli** -regtest -rpcuser=test -rpcpassword=test -rpcconnect="127.0.0.1" -rpcport=12345 -datadir="$PWD/node1" listaccounts  //101개 블록 생성  root@1060e4b3d645:~# **bitcoin-cli** -regtest -rpcuser=test -rpcpassword=test -rpcconnect="127.0.0.1" -rpcport=12345 -datadir="$PWD/node1" generate 101  //계정 확인  root@1060e4b3d645:~# **bitcoin-cli** -regtest -rpcuser=test -rpcpassword=test -rpcconnect="127.0.0.1" -rpcport=12345 -datadir="$PWD/node1" listaccounts  (🡺이름없는 마스터 계정에 50BTC이 지급된 것을 볼 수 있다, 얘가 채굴 노드)  //블록 개수 확인  root@1060e4b3d645:~# **bitcoin-cli** -regtest -rpcuser=test -rpcpassword=test -rpcconnect="127.0.0.1" -rpcport=12345 -datadir="$PWD/node1" getblockcount  //블록 내용 보기  root@1060e4b3d645:~# **bitcoin-cli** -regtest -rpcuser=test -rpcpassword=test -rpcconnect="127.0.0.1" -rpcport=12345 -datadir="$PWD/node1" getblock 223537018d9fcc16fb956ad2a9dc7fbc45e047727515259c972b2f976d9695a  (🡺 맨 마지막 파라미터는 101번째 블록의 id임)  //acc1이라는 라벨이 붙은 계정의 주소 찾기  root@1060e4b3d645:~# **bitcoin-cli** -regtest -rpcuser=test -rpcpassword=test -rpcconnect="127.0.0.1" -rpcport=12345 -datadir="$PWD/node1" getaccountaddress acc1  //위 주소에 10BTC 보내기  root@1060e4b3d645:~# **bitcoin-cli** -regtest -rpcuser=test -rpcpassword=test -rpcconnect="127.0.0.1" -rpcport=12345 -datadir="$PWD/node1" sendtoaddress 2MvZqjR7WkbwLQtgDsQp4sD52GuPg5hvR7X 10  (🡺마스터 계정에서 지정 계정으로 10BTC 전송)  //블록 개수 확인  root@1060e4b3d645:~# **bitcoin-cli** -regtest -rpcuser=test -rpcpassword=test -rpcconnect="127.0.0.1" -rpcport=12345 -datadir="$PWD/node1" getblockcount  (🡺여전히 101개의 블록만 보임)  //송금 트랜잭션을 블록에 넣기  root@1060e4b3d645:~# **bitcoin-cli** -regtest -rpcuser=test -rpcpassword=test -rpcconnect="127.0.0.1" -rpcport=12345 -datadir="$PWD/node1" generate 1  (🡺강제로 블록 생성, 마스터 계정은 채굴 노드이므로 보상 50BTC를 받는다)  //블록 개수 확인  root@1060e4b3d645:~# **bitcoin-cli** -regtest -rpcuser=test -rpcpassword=test -rpcconnect="127.0.0.1" -rpcport=12345 -datadir="$PWD/node1" getblockcount  (🡺102개의 블록 보임)  //계정 확인  root@1060e4b3d645:~# **bitcoin-cli** -regtest -rpcuser=test -rpcpassword=test -rpcconnect="127.0.0.1" -rpcport=12345 -datadir="$PWD/node1" listaccounts  (마스터 계정이 50BTC 이상인 것이 보임)  //UTXO 확인  root@1060e4b3d645:~# **bitcoin-cli** -regtest -rpcuser=test -rpcpassword=test -rpcconnect="127.0.0.1" -rpcport=12345 -datadir="$PWD/node1" listunspent  //전체 금액 조회  root@1060e4b3d645:~# **bitcoin-cli** -regtest -rpcuser=test -rpcpassword=test -rpcconnect="127.0.0.1" -rpcport=12345 -datadir="$PWD/node1" getbalance  //계정별 잔액 조회  root@1060e4b3d645:~# **bitcoin-cli** -regtest -rpcuser=test -rpcpassword=test -rpcconnect="127.0.0.1" -rpcport=12345 -datadir="$PWD/node1" getbalance acc1 |

# Bitcoin-cli 명령어

|  |  |  |  |
| --- | --- | --- | --- |
| **명령** | **매개 변수** | **기술** | **잠금 해제 된 지갑이 필요하십니까?(v0.4.0 +)** |
| addmultisigaddress | <nrequired> < ‘[ “key”, “key”]’> [account] | nrequired-to-sign 다중 서명 주소를 지갑에 추가하십시오. 각 키는 비트 코인 주소 또는 16 진수로 인코딩 된 공개 키입니다.[account]가 지정되면 [account]에 주소를 할당하십시오. 주소가 포함 된 문자열을 반환합니다. | 엔 |
| addnode | <node> <add / remove / onetry> | **version 0.8** addnode 목록에서 <node>를 추가하거나 제거하려고 시도하거나 <node>에 대한 연결을 한 번 시도합니다. | 엔 |
| backupwallet | <대상> | 안전하게 wallet.dat를 destination에 복사합니다. 대상은 filename 또는 filename이있는 디렉토리 또는 경로 일 수 있습니다. | 엔 |
| createmultisig | <nrequired> < ‘[ “key,”key “]’> | 다중 서명 주소를 생성하고 json 객체를 반환합니다. |  |
| createrawtransaction | [{ “”txid “: txid,”vout “: n}, …] {주소 : 금액, …} | **버전 0.7은** 작성 [**~~원시 거래~~**](https://en.bitcoin.it/wiki/Raw_Transactions) 주어진 입력을 지출합니다. | 엔 |
| decoderawtransaction | <16 진수 문자열> | **버전 0.7**[**~~원시 트랜잭션에~~**](https://en.bitcoin.it/wiki/Raw_Transactions) 대해 사람이 읽을 수있는 JSON 객체를 생성 [**~~합니다~~**](https://en.bitcoin.it/wiki/Raw_Transactions) . | 엔 |
| dumpprivkey | <bitcoinaddress> | <bitcoinaddress>에 해당하는 개인 키를 표시합니다. | 와이 |
| dumpwallet | <파일 이름> | **버전 0.13.0** 모든 지갑 개인 키를 파일로 내 **보냅니다.** | 와이 |
| encryptwallet | <passphrase> | <passphrase>를 사용하여 지갑을 암호화합니다. | 엔 |
| getaccount | <bitcoinaddress> | 주어진 주소와 관련된 계정을 반환합니다. | 엔 |
| getaccountaddress | <account> | 이 계정으로 지불을 받기위한 현재 비트 코인 주소를 반환합니다. <account>가 없으면 반환 될 관련 새 주소와 함께 만들어집니다. | 엔 |
| getaddednodeinfo | <dns> [노드] | **version 0.8** 제공된 추가 노드 또는 추가 된 모든 노드에 대한 정보를 반환합니다.(onetry addnodes는 여기에 나열되지 않습니다.) dns가 false이면 추가 된 노드 목록 만 제공되며, 그렇지 않으면 연결된 정보도 사용할 수 있습니다. |  |
| getaddressesbyaccount | <account> | 지정된 계정의 주소 목록을 반환합니다. | 엔 |
| getbalance | [계정] [minconf = 1] | [account]가 지정되지 않으면 서버의 사용 가능한 총 잔액을 반환합니다. [account]가 지정되면 계정의 잔액을 반환합니다. | 엔 |
| getbestblockhash |  | **버전 0.9** 가장 긴 블록 체인에서 가장 좋은 (팁) 블록의 해시를 반환합니다. | 엔 |
| getblock | <해시> | 지정된 해시가있는 블록에 대한 정보를 반환합니다. | 엔 |
| getblockcount |  | 가장 긴 블록 체인의 블록 수를 반환합니다. | 엔 |
| getblockhash | <색인> | <index>에서 best-block-chain의 블록 해시를 반환합니다. 인덱스 0은 [**~~기원 블록입니다.~~**](https://en.bitcoin.it/wiki/Genesis_block) | 엔 |
| getblocknumber |  | **지원되지 않습니다** . **버전 0.7에서 제거되었습니다** . getblockcount를 사용하십시오. | 엔 |
| getblocktemplate | [params] | 작업 할 블록을 만드는 데 필요한 데이터를 반환합니다. 매개 변수에 대한 자세한 내용은 [**~~BIP\_0022~~**](https://en.bitcoin.it/wiki/BIP_0022) 를 참조하십시오 . | 엔 |
| getconnectioncount |  | 다른 노드에 대한 연결 수를 반환합니다. | 엔 |
| getdifficulty |  | 최소 작업 난이도의 배수로 작업 증명 난이도를 반환합니다. | 엔 |
| getgenerate |  | 비트 코인이 현재 해시를 생성하는지 여부를 true 또는 false로 반환합니다. | 엔 |
| gethashespersec |  | 생성하는 동안 최근 초당 성능 측정 값을 반환합니다. | 엔 |
| getinfo |  | 다양한 상태 정보를 포함하는 객체를 반환합니다. | 엔 |
| getmemorypool | [데이터] | **v0.7.0에서 getblocktemplate, submitblock, getrawmempool로 대체되었습니다.** | 엔 |
| getmininginfo |  | 마이닝 관련 정보를 포함하는 객체를 반환합니다.   * 블록들 * 현재 차단하다 * currentblocktx * 어려움 * 오류 * 일으키다 * 유전자 명세 * 해시 퍼슨 * pooledtx * 시험 네트 | 엔 |
| getnewaddress | [계정] | 지불을 받기위한 새로운 비트 코인 주소를 반환합니다. [계좌]가 명시되어 있으면 해당 주소로받은 지불금은 [계좌]로 입금됩니다. | 엔 |
| getpeerinfo |  | **버전 0.7** 연결된 각 노드에 대한 데이터를 반환합니다. | 엔 |
| getrawchangeaddress | [계정] | **버전 0.9** 변경 사항을 수신 할 새 Bitcoin 주소를 반환합니다. 이것은 원시 트랜잭션과 함께 사용하기위한 것이며 정상적인 사용은 아닙니다. | 엔 |
| getrawmempool |  | **버전 0.7** 메모리 풀의 모든 트랜잭션 ID를 반환합니다. | 엔 |
| getrawtransaction | <txid> [verbose = 0] | **버전 0.7** 주어진 트랜잭션 ID에 대한 [**~~원시 트랜잭션~~**](https://en.bitcoin.it/wiki/Raw_Transactions) 표현을 반환 합니다. | 엔 |
| getreceivedbyaccount | [계정] [minconf = 1] | 적어도 [minconf] 확인이있는 거래에서 [account]가있는 주소로받은 총 금액을 반환합니다. [account]가 제공되지 않으면 모든 계정에 대한 모든 거래가 포함됩니다. (버전 0.3.24) | 엔 |
| getreceivedbyaddress | <bitcoinaddress> [minconf = 1] | 적어도 [minconf] 확인이있는 트랜잭션에서 <bitcoinaddress>가받은 금액을 반환합니다. 누군가가 여러 트랜잭션에서 주소를 보낸 경우를 올바르게 처리합니다. 주소는 거래 수신에만 사용됩니다. 로컬 월렛의 주소에만 작동하며 외부 주소는 항상 0으로 표시됩니다. | 엔 |
| gettransaction | <txid> | 지정된 트랜잭션에 대한 다음을 포함하는 객체를 반환합니다.   * “금액”: 거래 총액 * “confirmations”: 거래 확인 건수 * “txid”: 거래 ID * “시간”: 거래와 관련된 시간 [**~~[1]~~**](https://en.bitcoin.it/wiki/Original_Bitcoin_client/API_calls_list#cite_note-1) . * “details”- 다음을 포함하는 객체의 배열입니다.   + “계정”   + “주소”   + “범주”   + “양”   + “보수” | 엔 |
| gettxout | <txid> <n> [includemempool = true] | 사용되지 않은 트랜잭션 출력 (UTXO)에 대한 세부 정보를 반환합니다. | 엔 |
| gettxoutsetinfo |  | 미사용 트랜잭션 출력 (UTXO) 세트에 대한 통계를 리턴합니다. | 엔 |
| [**getwork**](https://en.bitcoin.it/wiki/Getwork) | [데이터] | [data]가 지정되어 있지 않은 경우, 서식 화 된 해시 데이터가 돌려 주어집니다.   * “중간 상태”: 데이터의 첫 번째 절반을 해싱 한 후 사전 계산 된 해시 상태 * “데이터”: 블록 데이터 * “hash1”: 두 번째 해시에 대한 형식화 된 해시 버퍼 * “target”: 리틀 엔디안 해시 타겟   [data]를 지정하면 블록을 풀려고 시도하고 성공하면 true를 반환합니다. | 엔 |
| help | [명령] | 명령을 나열하거나 명령에 대한 도움말을 얻으십시오. | 엔 |
| importprivkey | <bitcoinprivkey> [label] [rescan = true] | dumpprivkey가 반환 한 개인 키를 지갑에 추가합니다. **~~[재검색](https://en.bitcoin.it/wiki/How_to_import_private_keys" \l "Import_Private_key.28s.29)~~** 이 완료되면 기존 트랜잭션을 찾는 데 시간이 걸릴 수 있습니다 . **선택적 [rescan] 매개 변수가 0.8.0에 추가되었습니다.**참고 : 공개 키를 가져올 필요가 없습니다. **~~[ECDSA](https://en.bitcoin.it/wiki/Elliptic_Curve_Digital_Signature_Algorithm)~~** (RSA와 달리)는 개인 키에서 계산할 수 있습니다. | 와이 |
| invalidateblock | <해시> | 일치 규칙을 위반 한 것처럼 영구적으로 블록을 무효로 표시합니다. | 엔 |
| keypoolrefill |  | 키풀을 채우려면 지갑 암호를 설정해야합니다. | 와이 |
| listaccounts | [minconf = 1] | 계정 이름을 키로, 계정 잔액을 값으로 갖는 개체를 반환합니다. | 엔 |
| listaddressgroupings |  | **버전 0.7** 동전 지갑에있는 모든 주소와 동전 제어에 사용 된 정보를 반환합니다. | 엔 |
| listreceivedbyaccount | [minconf = 1] [includeempty = false] | 다음을 포함하는 객체의 배열을 반환합니다.   * “계정”: 수신 주소의 계정 * ‘금액’:이 계정으로 수신 한 총 금액 * “confirmations”: 가장 최근 트랜잭션의 확인 수 | 엔 |
| listreceivedbyaddress | [minconf = 1] [includeempty = false] | 다음을 포함하는 객체의 배열을 반환합니다.   * “주소”: 수신 주소 * “account”: 수신 주소의 계정 * “금액”: 주소로받은 총 금액 * “confirmations”: 가장 최근 트랜잭션의 확인 수   시스템에서 계정 목록을 얻으려면 bitcoind listreceivedbyaddress 0 true를 실행하십시오. | 엔 |
| listsinceblock | [blockhash] [target-confirmations] | [blockhash] 블록 이후의 모든 트랜잭션을 블록에서 가져 오거나, 생략 된 경우 모든 트랜잭션을 가져옵니다. [목표 확정]을 의도적으로 **하지 않습니다** 반환 트랜잭션의 목록에 영향을 미치지 만 반환 “lastblock”값에 영향을 미칩니다. **~~[[1]](https://github.com/bitcoin/bitcoin/pull/199" \l "issuecomment-1514952)~~** | 엔 |
| listtransactions | [account] [count = 10] [from = 0] | 계정 [account]의 첫 [from] 트랜잭션을 건너 뛰는 가장 최근의 트랜잭션을 [count] 개까지 반환합니다. [account]가 제공되지 않으면 모든 계정의 최근 거래가 반환됩니다. | 엔 |
| listunspent | [minconf = 1] [maxconf = 999999] | **버전 0.7** 지갑의 미사용 트랜잭션 입력 배열을 리턴합니다. | 엔 |
| listlockunspent |  | **버전 0.8** 임시로 출력 할 수없는 출력 목록을 반환합니다. |  |
| lockunspent | <unlock?> [배열 오브 개체] | **버전 0.8** 임시로 출력 할 수없는 출력의 목록을 업데이트합니다 |  |
| move | <fromaccount> <toaccount> <amount> [minconf = 1] [comment] | 지갑 한 계정에서 다른 계정으로 이동하십시오. | 엔 |
| sendfrom | <fromaccount> <tobitcoinaddress> <amount> [minconf = 1] [comment] [comment-to] | <금액>은 실수이며 소수점 이하 8 자리로 반올림됩니다. 지정된 금액을 지정된 주소로 보내서 [minconf] 확인을 사용하여 계정의 잔액이 올바른지 확인합니다. 성공하면 트랜잭션 ID를 반환합니다 (JSON 객체가 아님). | 와이 |
| sendmany | <fromaccount> {주소 : 금액, …} [minconf = 1] [의견] | 양은 배정 밀도 부동 소수점 숫자입니다. | 와이 |
| sendrawtransaction | <hexstring> | **version 0.7**[**~~raw 트랜잭션~~**](https://en.bitcoin.it/wiki/Raw_Transactions) (serialized, hex-encoded)을 로컬 노드와 네트워크에 제출 합니다. | 엔 |
| sendtoaddress | <bitcoinaddress> <amount> [comment] [comment-to] | <금액>은 실수이며 소수점 이하 8 자리로 반올림됩니다. 성공하면 트랜잭션 ID <txid>를 반환합니다. | 와이 |
| setaccount | <bitcoinaddress> <account> | 주어진 주소와 관련된 계정을 설정합니다. 동일한 계정에 이미 할당 된 주소를 할당하면 해당 계정과 관련된 새 주소가 만들어집니다. | 엔 |
| setgenerate | <생성> [genproclimit] | 생성을 켜거나 끄려면 <generate>가 true 또는 false입니다. 세대는 [genproclimit] 프로세서로 제한되며 -1은 무제한입니다. | 엔 |
| settxfee | <금액> | <amount>는 실수이고 가장 가까운 0.00000001로 반올림됩니다. | 엔 |
| signmessage | <bitcoinaddress> <message> | 주소의 개인 키로 메시지에 서명하십시오. | 와이 |
| signrawtransaction | <hexstring> [txid, txid, “vout”: n, “scriptPubKey”: 16 진수}, …] [<privatekey1>, …] | **버전 0.7**[**~~원시 트랜잭션에~~**](https://en.bitcoin.it/wiki/Raw_Transactions) 서명을 추가 하고 결과 원시 트랜잭션을 반환합니다. | 예 / 아니오 |
| stop |  | 비트 코인 서버를 중지하십시오. | 엔 |
| submitblock | <hex data> [optional-params-obj] | 네트워크에 새 블록을 제출하려고 시도합니다. | 엔 |
| validateaddress | <bitcoinaddress> | <bitcoinaddress>에 대한 정보를 반환하십시오. | 엔 |
| verifymessage | <bitcoinaddress> <signature> <message> | 서명 된 메시지를 확인하십시오. | 엔 |
| walletlock |  | 지갑 암호화 키를 메모리에서 제거하여 지갑을 잠급니다. 이 메소드를 호출 한 후에는 지갑을 잠금 해제해야하는 메소드를 호출하기 전에 walletpassphrase를 다시 호출해야합니다. | 엔 |
| walletpassphrase | <passphrase> <timeout> | <timeout> 초 동안 지갑 암호 해독 키를 메모리에 저장합니다. | 엔 |
| walletpassphrasechange | <oldpassphrase> <newpassphrase> | 지갑 패스 프레이즈를 <oldpassphrase>에서 <newpassphrase>로 변경합니다. | 엔 |

1. node 설치

|  |
| --- |
| root@1060e4b3d645:~# **apt-get update**  root@1060e4b3d645:~# **apt-get upgrade**  root@1060e4b3d645:~# **curl -sL** [**https://deb.nodesource.com/setup\_10.x**](https://deb.nodesource.com/setup_10.x) **| bash -**  root@1060e4b3d645:~# **apt-get install node**  root@1060e4b3d645:~# **node -v**  **(만약 node 가 설치되지 않는다면)**  root@1060e4b3d645:~# **apt-get install npm**  root@1060e4b3d645:~# **npm install -g n**  root@1060e4b3d645:~# **n latest**  root@1060e4b3d645:~# **node -v** |

1. node package 작성

|  |
| --- |
| mkdir node\_web1  npm init  npm install bitcoind-rpc-client express nodemon |

1. server.js