1. 下载钱包

<https://bitcoin.org/en/download>

1. bitcoin.conf配置文件
   1. 在安装根目录下新建此文件
   2. 个人当前配置

# 告知 Bitcoin-QT 接受 JSON-RPC 命令

server=1

# JSON-RPC 连接使用的用户名和密码

rpcuser=root

rpcpassword=root

rpcport=8332

rpctimeout=30

# 需要远程访问钱包的时候这个需要配置

rpcbind=0.0.0.0

# rpc访问白名单

rpcallowip=192.168.1.14

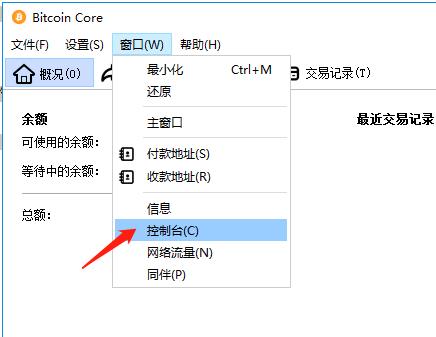
* 1. 其他配置

<https://blog.csdn.net/mongo_node/article/details/89633834>

1. cmd进入安装根目录，以下命令运行bitcoin钱包

bitcoin-qt -conf=E:\Bitcoin\bitcoin.conf

1. 开始下载区块
2. 可以在控制台中输入指令



* 1. help：获取指令

1. 所有指令及其解释

<http://cw.hubwiz.com/card/c/bitcoin-json-rpc-api/>

1. Java调用
   1. 导入json-rpc Jar

<dependency>  
 <groupId>com.github.briandilley.jsonrpc4j</groupId>  
 <artifactId>jsonrpc4j</artifactId>  
 <version>1.1</version>  
</dependency>

* 1. 小demo

import com.alibaba.fastjson.JSONObject;  
import com.googlecode.jsonrpc4j.Base64;  
import com.googlecode.jsonrpc4j.JsonRpcHttpClient;  
  
import java.net.URL;  
import java.util.HashMap;  
  
*/\*\*  
 \* <p>  
 \*  
 \* </p>  
 \*  
 \* @author 郭海斌  
 \* @since 2019/7/2  
 \*/*public class MainTest {  
 public static void main(String[] args) throws Throwable {  
 //账号密码 - 改为bitcoin.conf中的账号密码  
 String cred = Base64.*encodeBytes*("root:root".getBytes());  
 //json-rpc的头，放入账号密码  
 HashMap<String, String> headers = new HashMap<>();  
 headers.put("Authorization", "Basic " + cred);  
 //指定钱包 - 默认钱包：/wallet/  
 JsonRpcHttpClient client = new JsonRpcHttpClient(new URL("http://192.168.1.15:8332/wallet/testnet3\\wallets"), headers);  
 //不指定钱包  
// JsonRpcHttpClient client = new JsonRpcHttpClient(new URL("http://127.0.0.1:8332"), headers);  
 //获取区块hash值  
// String result = client.invoke("getblockhash", new Object[]{1}, Object.class).toString();  
 //获取区块链信息  
// String result = client.invoke("getblockchaininfo", new Object[]{}, Object.class).toString();  
 if(!*isConnected*(client)) {  
 System.*out*.println("连接失败，账号或密码错误！");  
 return;  
 }  
 *sendToAddress*(client, "3PE1fzCoxCfhStsbmULhkn77UJGHc9XsH3", 1);  
 }  
  
 //检测连接是否成功  
 public static boolean isConnected(JsonRpcHttpClient client) {  
 try {  
 client.invoke("getblockchaininfo", new Object[]{}, Object.class);  
 return true;  
 } catch (Throwable e) {  
 return false;  
 }  
 }  
  
 //获取钱包余额  
 public static void getBalance(JsonRpcHttpClient client) throws Throwable {  
 System.*out*.println("获取钱包余额...");  
 String result = client.invoke("getbalance", new Object[]{}, Object.class).toString();  
 System.*out*.println("钱包余额：" + result);  
 }  
  
 //生成钱包地址 - 为了让别人转钱进来  
 public static String getNewAddress(JsonRpcHttpClient client) throws Throwable {  
 System.*out*.println("生成钱包地址...");  
 String result = client.invoke("getnewaddress", new Object[]{}, Object.class).toString();  
 System.*out*.println("钱包地址：" + result);  
 return result;  
 }  
  
 //验证地址是否有效  
 public static boolean validateAddress(JsonRpcHttpClient client, String address) throws Throwable {  
 System.*out*.println("验证地址是否有效...");  
 */\*\*  
 \* 无论传进来的地址是否有效，都会返回一个JSON对象，都包含isvalid字段，可通过此字段判断是否地址有效  
 \* 正确时包含其他信息  
 \*/* String result = client.invoke("validateaddress", new Object[]{address}, JSONObject.class).toString();  
 System.*out*.println(result);  
 JSONObject object = JSONObject.*parseObject*(result);  
 boolean isvalid = (boolean) object.get("isvalid");  
 return isvalid;  
 }  
  
 //转账 - 转账失败和余额不够都会报错  
 public static void sendToAddress(JsonRpcHttpClient client, String address, double value) {  
 System.*out*.println("转账...");  
 try {  
 if(*validateAddress*(client, address)) {  
 //转账金额不能为0  
 if (0 == value) {  
 System.*out*.println("转账金额不能为0！");  
 return;  
 }  
 //金额不够或者转0都会报异常，0可以判断，那么异常就都是余额不足的情况  
 String result = client.invoke("sendtoaddress", new Object[]{address, value}, Object.class).toString();  
 System.*out*.println("转账成功，转账结果：" + result);  
 }  
 } catch (Throwable e) {  
 System.*out*.println("您的余额不足！");  
 }  
 }  
}

yml：

bitcoin:  
 *# 钱包url - bitcoin钱包url* url: http://192.168.1.15:8332  
 *# 钱包账号密码* username: root  
 password: root  
 *# 钱包* wallets:  
 one: /wallet/testnet3\wallets  
 two: /wallet/  
 *# json-rpc api* json-rpc:  
 *# 获取区块链信息* getblockchaininfo: getblockchaininfo  
 *# 获取钱包余额* getbalance: getbalance  
 *# 生成钱包地址* getnewaddress: getnewaddress  
 *# 检测钱包地址是否有效* validateaddress: validateaddress  
 *# 转账* sendtoaddress: sendtoaddress

1. Json-rpc协议（包含错误相应码）

<https://www.cnblogs.com/cielosun/p/6762550.html>

1. 总结：
   1. 调用很简单，需求很难搞懂（mmp）
   2. 找的海枯石烂，结果却很easy（mmp\*2）
   3. 接下来的以太坊，搞快点