1. 下载Ethereum-Wallet钱包

<https://github.com/ethereum/mist/releases/tag/v0.11.1>

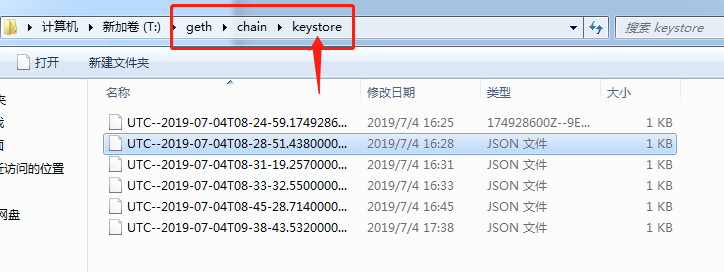
1. 下载Geth客户端

<https://geth.ethereum.org/downloads/>

1. 运行Geth客户端（注意方式）
2. geth --identity "mainnet etherum" --datadir "./chain" --rpc --rpcapi "admin,web3,eth,personal,miner,net" --rpcaddr 0.0.0.0 --rpcport 8545 --cache=1024 console
3. 参数解释

<https://www.jianshu.com/p/e2630bc1d510>

1. 运行Ethereum-Wallet
2. 如果一直卡在创建中应该就是Geth没运行起来
3. 创建钱包后会存储在Geth中的keystore目录下



1. 后台使用web3j调用接口
2. Demo
   1. 导入jar

<dependency>  
 <groupId>org.web3j</groupId>  
 <artifactId>core</artifactId>  
 <version>4.2.0</version>  
</dependency>

<!-- 或者下面，稍微不同就是解析钱包文件时多了个参数 -->

<dependency>  
 <groupId>org.web3j</groupId>  
 <artifactId>web3j-spring-boot-starter</artifactId>  
 <version>1.6.0</version>  
</dependency>

* 1. 代码

package cn.mb.wallets;  
  
import cn.mb.utils.FileUtil;  
import org.web3j.crypto.Credentials;  
import org.web3j.crypto.WalletUtils;  
import org.web3j.protocol.Web3j;  
import org.web3j.protocol.core.DefaultBlockParameterName;  
import org.web3j.protocol.core.methods.response.TransactionReceipt;  
import org.web3j.protocol.http.HttpService;  
import org.web3j.tx.Transfer;  
import org.web3j.utils.Convert;  
  
import java.io.File;  
import java.math.BigDecimal;  
import java.math.BigInteger;  
import java.util.HashMap;  
import java.util.Map;  
  
*/\*\*  
 \* <p>  
 \* Ethereum-Wallet钱包相关操作  
 \* </p>  
 \*  
 \** ***@author*** *郭海斌  
 \** ***@since*** *2019/9/9  
 \*/*public class ETH {  
  
 public static void main(String[] args) throws Exception {  
 String url = "http://159.138.39.181:8545/";  
 String address = "0xf561437c494d7049eefd02d81ebef11cf112dbae";  
 // 钱包文件存放目录  
 String dir = "/wallets/eth";  
 // 钱包文件密码  
 String password = "1051127705";  
 Web3j web3j = *connect*(url);  
 boolean validateAddress = *validateAddress*(address);  
 System.*out*.println("钱包地址是否有效：" + validateAddress);  
 String addr = *getAddress*(dir, password);  
 System.*out*.println("地址：" + addr);  
 BigDecimal balance = *getBalance*(web3j, address);  
 System.*out*.println("余额：" + balance);  
 Map<String, Object> map = new HashMap<>();  
 map.put("address", address);  
 map.put("toAddress", "0xa28622fc63e05b3e0f790d0e1a3f13177c38731e");  
 map.put("count", 10.5);  
 *tranferMoney*(web3j, password, dir, map);  
 }  
  
 */\*\*  
 \* 交易  
 \** ***@param*** *web3j  
 \** ***@param*** *password 文件密码  
 \** ***@param*** *dir 钱包目录  
 \** ***@param*** *map 转账地址、收款地址、转账额  
 \** ***@return*** *\** ***@throws*** *Exception  
 \*/* public static Map<String, Object> tranferMoney(Web3j web3j, String password, String dir, Map<String, Object> map) throws Exception {  
 Map<String, Object> resultMap = new HashMap<>();  
 String toAddress = map.get("toAddress").toString();  
 double count = Double.*parseDouble*(map.get("count").toString());  
 //验证地址是否合法  
 if (!*validateAddress*(toAddress)) {  
 throw new Exception("钱包地址不符合格式，转账失败！");  
 }  
 //转账金额不能为0  
 if (count <= 0) {  
 throw new Exception("转账金额不能小于或等于0！");  
 }  
 //获取钱包余额  
 String address = map.get("address").toString();  
 BigDecimal balance = *getBalance*(web3j, address);  
 if (balance.doubleValue() < count) {  
 throw new Exception("余额不足，无法交易！");  
 }  
 // 获取钱包文件  
 File file = *getWalletFileByAddr*(address, dir);  
 if (!file.exists()) {  
 throw new Exception("钱包文件丢失，请联系管理员！");  
 }  
 */\*\* 交 易 \*/* //加载钱包文件  
 Credentials credentials = WalletUtils.*loadCredentials*(password, file);  
 //交易 - 余额不足时/其他情况会交易失败  
 TransactionReceipt transactionReceipt = null;  
 try {  
 transactionReceipt = Transfer.*sendFunds*(web3j, credentials, toAddress, BigDecimal.*valueOf*(count), Convert.Unit.*ETHER*).send();  
 System.*out*.println(" 转账成功，交易hash为：" + transactionReceipt.getTransactionHash() +"！");  
 resultMap.put("transactionHash", transactionReceipt.getTransactionHash());  
 } catch (Exception e) {  
 throw new Exception("交易过程发生故障，转账失败，请稍后重试！");  
 }  
 */\*\* 另一种交易方式 \*/*// Credentials credentials = WalletUtils.loadCredentials(password, file);  
// // get the next available nonce - 每笔交易对应着一个唯一的nonce  
// BigInteger nonce = web3j.ethGetTransactionCount(address, DefaultBlockParameterName.LATEST).send().getTransactionCount();  
// // create our transaction - 此处小费是0，所以会以一般速度去转账 - 遗留问题：此方式转账只能是整数而非小数  
// RawTransaction rawTransaction = RawTransaction.createEtherTransaction(  
// nonce, BigInteger.ZERO, BigInteger.ZERO, toAddress, BigInteger.valueOf(count));  
// // sign & send our transaction - 签名  
// byte[] signedMessage = TransactionEncoder.signMessage(rawTransaction, credentials);  
// String hexValue = Numeric.toHexString(signedMessage);  
// //开始交易  
// EthSendTransaction ethSendTransaction = web3j.ethSendRawTransaction(hexValue).send();  
// //判断交易是否失败  
// Response.Error error = ethSendTransaction.getError();  
// if(Objects.isNull(error)) {  
// System.out.println("转账成功！");  
// resultMap.put("message", "转账成功！");  
// resultMap.put("code", 1);  
// } else {  
// System.out.println("交易发生异常，异常信息为：" + error.getMessage() + "，转账失败！");  
// resultMap.put("message", "交易发生异常，异常信息为：" + error.getMessage() + "，转账失败！");  
// resultMap.put("code", 0);  
// }  
 return resultMap;  
 }  
  
 */\*\*  
 \* 根据钱包地址及钱包目录获取相对应的钱包文件  
 \** ***@param*** *address  
 \** ***@param*** *dirPath  
 \** ***@return*** *\** ***@throws*** *Exception  
 \*/* private static File getWalletFileByAddr(String address, String dirPath) throws Exception {  
 // 获取存放钱包文件的绝对路径  
 String absPath = FileUtil.*getAbsPath*(dirPath);  
 // 获取该目录  
 File dir = new File(absPath);  
 // 获取目录下所有文件名  
 String[] list = dir.list();  
 System.*out*.println(list.length);  
 for (String str : list) {  
 // 钱包文件名中会包含钱包地址，依据此来获取对应文件  
 if (str.contains(address)) {  
 return new File(absPath + str);  
 }  
 }  
 return null;  
 }  
  
 */\*\*  
 \* 根据钱包地址获取其余额  
 \** ***@param*** *web3j  
 \** ***@param*** *address 地址  
 \** ***@return*** *\** ***@throws*** *Exception  
 \*/* public static BigDecimal getBalance(Web3j web3j, String address) throws Exception {  
 BigInteger balance = web3j.ethGetBalance(address, DefaultBlockParameterName.*EARLIEST*).send().getBalance();  
 return BigDecimal.*valueOf*(balance.doubleValue());  
 }  
  
 */\*\*  
 \* 获取钱包地址  
 \** ***@param*** *dir 钱包文件存放目录  
 \** ***@param*** *password 密码  
 \** ***@return*** *\** ***@throws*** *Exception  
 \*/* public static String getAddress(String dir, String password) throws Exception {  
 // 获取存放钱包文件目录的绝对路径  
 String absPath = FileUtil.*getAbsPath*(dir);  
 // 创建此目录  
 FileUtil.*createDir*(absPath);  
 // 生成钱包文件，可以获取文件名  
 String fileName = WalletUtils.*generateNewWalletFile*(password, new File(absPath), false);  
 // 加载凭证文件  
 Credentials credentials = WalletUtils.*loadCredentials*(password, absPath + "/" + fileName);  
 return credentials.getAddress();  
 }  
  
 */\*\*  
 \* 验证钱包地址是否正确  
 \** ***@param*** *address  
 \** ***@return*** *\*/* public static boolean validateAddress(String address) {  
 return WalletUtils.*isValidAddress*(address);  
 }  
  
 */\*\*  
 \* 连接ETH钱包  
 \** ***@return*** *\*/* public static Web3j connect(String url) {  
 return Web3j.*build*(new HttpService(url));  
 }  
  
}

1. 总结
2. 又搞了两天，以为很简单，结果又陷入坑里面了，但是总算搞完，下面小结
3. 如果是调客户端的API，首先把客户端玩一下
4. 玩大概知道什么个形式后，再开始找如何通过代码的方式去实现
5. 不要一上来就直接代码软件一起搞，特别费时间
6. 不过终究算搞懂了，哎，继续努力了，加油！