1> web3 脚本语言可通过RPC层来操作任意的以太坊node节点，因此你可以用其开发出属于自己的应用来操作你的私链。

2>web3两个主要的操作对象，分别是 eth（web3.eth 主要用于和以太链进行交互） 和 ssh（web3.ssh 主要用于协议交互）。当然，后续还会对更多的对象实现web3 协议的包裹实现。

3>在使用web3 之前，你需要将web3引入你的项目中，可通过一下集中方式引入到项目工程中：

* npm: npm install web3
* bower: bower install web3
* meteor: meteor add ethereum:web3
* vanilla: link the dist./web3.min.js

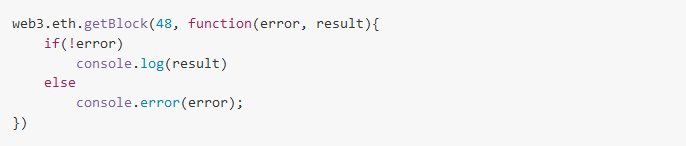
（在这里提供一个web3 源码下载地址：<https://github.com/ethereum/web3.js/tree/master> ）

4> 当将web3 引入到项目中后，你就需要去创建一个实例，可以用官方文档提供的下面一段代码进行创建：

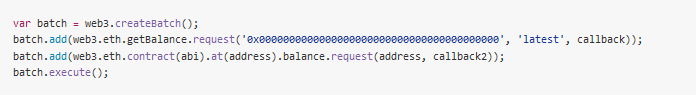


     在创建完一个实例对象之后，你就可以通过API进行操作了。

5>回调函数：对于为本地RPC节点设计的API来说，它们在默认情况下都是使用的同步Http 请求。如果你想使用异步请求，那么你需要向最后一个参数传入一个回调函数，例如：

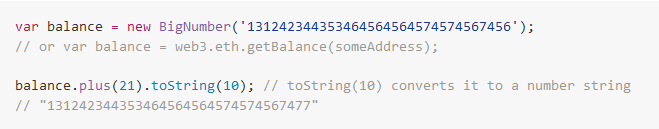


6>批处理请求：其使用队列进行处理请求，并可以做到一次性处理。（注意：批处理请求并不是很快，事实上在某些情况下一次性处理多笔请求时很快的，比如异步处理请求！批处理请求主要用于确保串行处理的请求。）例如：

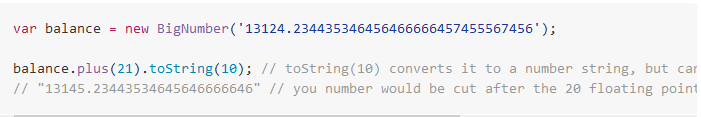


7>对于大数字的支持：

     web3 对于大数字的支持是通过 [BigNumber Library](https://github.com/MikeMcl/bignumber.js/) 来支持的。例如：



     但是对于大数子小数点后的位数要控制在wei单位内，如果超出，则不再支持。下面的列子是小数位到达20位儿导致不支持的例子：



8>web3 API 介绍

|  |  |  |
| --- | --- | --- |
| C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(5).png | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(6).png | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\2c84d3ffe4c4eb33672d0c14db88a6a4.png |
| C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(7).png | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(8).png |  |

|  |  |  |
| --- | --- | --- |
| 方法名 | 作用 | 案例 |
| **web3** | 此对象可调用所有的方法 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(9).png |
| web3.version.api | 此方法会返回js API的版本号 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(10).png |
| web3.version.node // or async web3.version.getNode(callback(error, result){ ... }) | 此方法会返回当前连接的客户端/节点的配置信息 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(11).png |
| web3.version.network // or async web3.version.getNetwork(callback(error, result){ ... }) | 此方法会返回网络协议的版本 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(12).png |
| web3.version.ethereum  // or async web3.version.getEthereum(callback(error, result){ ... }) | 此方法会返回以太坊协议的版本 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(13).png |
| web3.version.whisper  // or async web3.version.getWhisper(callback(error, result){ ... }) | 此方法会返回以密语协议的版本 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(14).png |
| web3.isConnected() | 此方法用于检查是否链接到节点，返回值为Boolean类型。链接成功返回值为true | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(15).png |
| web3.setProvider(provider) | 此方法用于设置连接客户端/节点的http信息 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(16).png |
| web3.currentProvider | 如果在集合中存在则会返回当前所存储的供应节点。此方法常被用来测试当前结点是否存在 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(17).png |
| web3.reset(keepIsSyncing) | 用于重新设置web3的状态。除了管理外其他均可以进行重新设置。去除所有的拦截机制，停止所有的查询。返回值为Boolean类型，如果返回值是true，则会去除所有的拦截机制，但会保留 [web3.eth.isSyncing()](https://github.com/ethereum/wiki/wiki/JavaScript-API#web3ethissyncing)。 | web3.reset(); |
| web3.sha3(string, options) | String（加密算法参数 Keccak-256 SHA3 ）  options（可用于设置一些必要的参数） | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(18).png |
| web3.toHex(mixed); | 用于将参数解析成十六进制，参数可以是任意类型： String|Number|Object|Array|BigNumber | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(19).png |
| web3.toAscii(hexString); | 将十六进制字符串转换成Ascii码 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(20).png |
| web3.fromAscii(string [, padding]); | 将任何ascii码数据转换成十六进制字符串，第二个参数可用于设置转换后的十六进制字符串的长度 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(21).png |
| web3.toDecimal(hexString); | 将十六进制（数字）转换成数字格式 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(22).png |
| web3.fromDecimal(number); | 将一个数字或数字字符串转换成十六进制字符串 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(23).png |
| web3.fromWei(number, unit) | 将以wei为单位的数据转换成一下类型的数据：   * kwei/ada * mwei/babbage * gwei/shannon * szabo * finney * ether * kether/grand/einstein * mether * gether * tether | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(24).png |
| web3.toWei(number, unit) | 将以下单位的数据转换成以wei为单位的数据：   * kwei/ada * mwei/babbage * gwei/shannon * szabo * finney * ether * kether/grand/einstein * mether * gether * tether | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(25).png |
| web3.toBigNumber(numberOrHexString); | 将一个给定的数字转换成BigNumber类型 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(26).png |
| web3.net.listening // or async web3.net.getListening(callback(error, result){ ... }) | 这个属性是只读的,表示该节点是否正在监听网络  连接。如果在监听，则会返回true，否则会返回  false | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(27).png |
| web3.net.peerCount // or async web3.net.getPeerCount(callback(error, result){ ... }) | 这个属性是只读的，此方法用于返回节点数 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(28).png |
| web3.eth.defaultAccount | 此方法用于获取节点的默认账户。默认情况下返回值为空。可以进行设置。此处的默认地址主要用于以下两种方法调用：   * [web3.eth.sendTransaction()](https://github.com/ethereum/wiki/wiki/JavaScript-API#web3ethsendtransaction) * [web3.eth.call()](https://github.com/ethereum/wiki/wiki/JavaScript-API#web3ethcall) | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(29).png |
| web3.eth.defaultBlock | 此方法的参数有以下几种：   * 块编号 * "earliest"：创世块 * "latest" ：最后的一个块（当前块） * "pending"：当前被挖的块（包括正在提交的交易）   此方法获取到的默认块主要作为一下方法的参数：   * [web3.eth.getBalance()](https://github.com/ethereum/wiki/wiki/JavaScript-API#web3ethgetbalance) * [web3.eth.getCode()](https://github.com/ethereum/wiki/wiki/JavaScript-API#web3ethgetcode) * [web3.eth.getTransactionCount()](https://github.com/ethereum/wiki/wiki/JavaScript-API#web3ethgettransactioncount) * [web3.eth.getStorageAt()](https://github.com/ethereum/wiki/wiki/JavaScript-API#web3ethgetstorageat) * [web3.eth.call()](https://github.com/ethereum/wiki/wiki/JavaScript-API#web3ethcall) * [contract.myMethod.call()](https://github.com/ethereum/wiki/wiki/JavaScript-API#contract-methods) * [contract.myMethod.estimateGas()](https://github.com/ethereum/wiki/wiki/JavaScript-API#contract-methods) | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(30).png |
| web3.eth.syncing  // or async  web3.eth.getSyncing(callback(error, result){ ... }) | 此方法用于获取同步对象。如果存在，则返回对象，如果不存在，则返回false。  返回对象中有三个参数：   startingBlock: 同步的起始块编号   currentBlock: 正在同步的块编号   highestBlock: 最后需要同步的块编号 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(31).png |
| web3.eth.isSyncing(callback); | 在同步开始，更新或停止时都会调用此方法。此方法返回的是一个同步对象，主要用于以下两种方法使用： syncing.addCallback()--添加另一个回调函数，当开始或结束同步时使用. syncing.stopWatching()--停止同步.  对于回调函数，返回值有两种：  Boolean：true（表示同步开始）false（表示同步结束）  Object(只有在同步开始时才会返回):返回对象中有三个参数：    startingBlock: 同步的起始块编号    currentBlock: 正在同步的块编号    highestBlock: 最后需要同步的块编号 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(32).png |
| web3.eth.coinbase // or async  web3.eth.getCoinbase(callback(error, result){ ... }) | 用于获取节点在挖矿时使用的基础账户 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(33).png |
| web3.eth.mining // or async  web3.eth.getMining(callback(error, result){ ... }) | 用于获取当前节点是否在挖矿。  true：正处于挖矿阶段  false：已停止挖矿 | C:\Users\YSH\AppData\Local\Temp\enhtmlclip\Image(34).png |

其它API可通过[英文文档](https://github.com/ethereum/wiki/wiki/JavaScript-API)查询其用法。