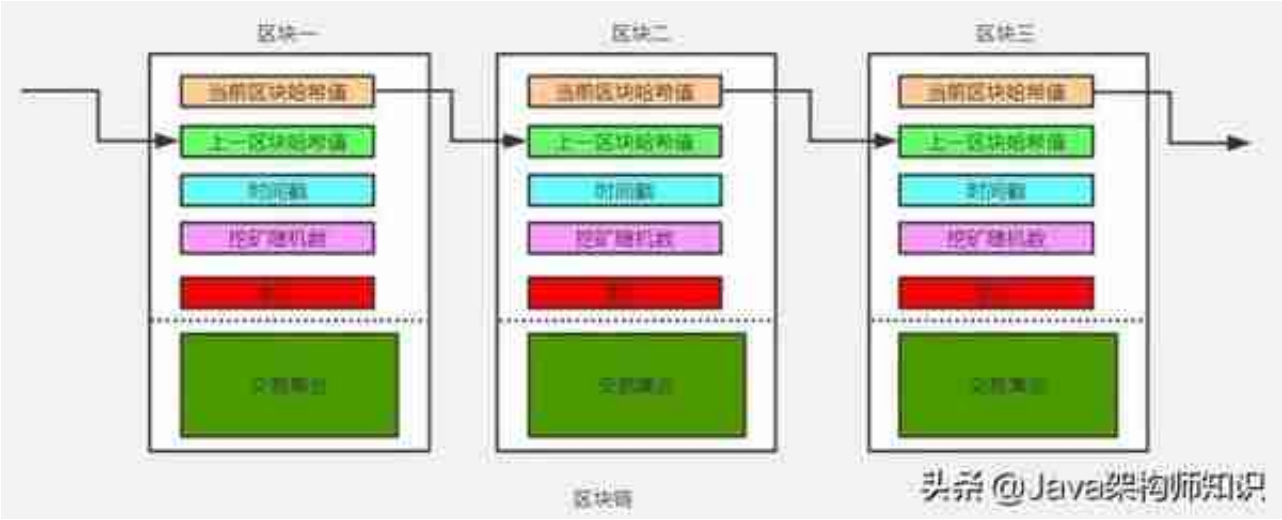
前言



区块中的交易集合记录的是一些特定的信息，在BTC网络中主要记录的是交易信息，在其他区块链网络中可以按照业务逻辑来保存相应的业务数据，如审计信息、版权信息、票据信息等，这也是区块链经常用来当做共享账本的原因。

打个比方，可以把区块链当做一个用来记账的笔记本，一个区块就相当于一页纸，上面记录了某一时间段内所有的账务信息，从第一页到最后一页，按照页码顺序排列起来就是一个完整的账本。

（2）区块链网络

实际的区块链系统由多个区块链节点组成，每个节点都运行着相同一套区块链主干网络的副本，且各个节点间通过P2P网络进行交互，并最终形成一个完整的区块链网络系统。

P2P网络具有可靠性、去中心化，以及开放性，各个节点之间交互运作、协同处理，每个节点在对外提供服务的同时也使用网络中其他节点所提供的服务。当某一个区块链节点产生新的区块时，会通过广播的方式告诉其他节点，其他节点通过网络接收到该区块信息时，会对这个区块信息进行验证，当有一定数量的节点都验证通过后，各个节点会把该区块更新到各自现有的区块链上，最终使得整个区块链网络中的各个节点信息保持一致，这也是区块链去中心化、可信任特性的体现。

区块链网络简易模型，如下图所示：

以太坊的概念首次在2013至2014年间由程序员Vitalik Buterin受比特币启发后提出，大意为“下一代加密货币与去中心化应用平台”。虽然以太坊作为平台可以在其上开发新的应用，但是由于以太坊的运行和BTC网络一样，采用的是Token机制，且平台性能不足，经常出现网络拥堵的情况，平台用来学习开发与测试区块链技术还可以，用于实际生产的话不太现实。

（3）联盟链开发框架：Hyperledger Fabric

Hyperledger Fabric 也叫超级账本，它是 IBM 贡献给 Linux 基金会的商用分布式账本，是面向企业应用的全球最大的分布式开源项目。像其他区块链技术一样，它也有一个账本，可以使用智能合约。Fabric的智能合约可以有多种架构，它可以用主流语言编程，例如Go、Java和Javascript，此外也可以使用Solidity。

至今，Fabric已获得了阿里巴巴、AWS、Azure、百度、谷歌、华为、IBM、甲骨文、腾讯等互联网巨头的支持。许多企业的区块链平台都把Fabric作为底层框架来使用，例如甲骨文。不过由于IBM对区块链的定义强调了区块链的分布式和不可变两个元素，对共识机制进行了削弱，采用了Kafka和zookeeper的“排序服务”实现共识，因此部分业内人士也称超级账本是“伪区块链”，但是即便如此，也抵挡不了企业对超级账本的喜爱，目前Fabric 2.0版本已经正式发布。

（4）小结

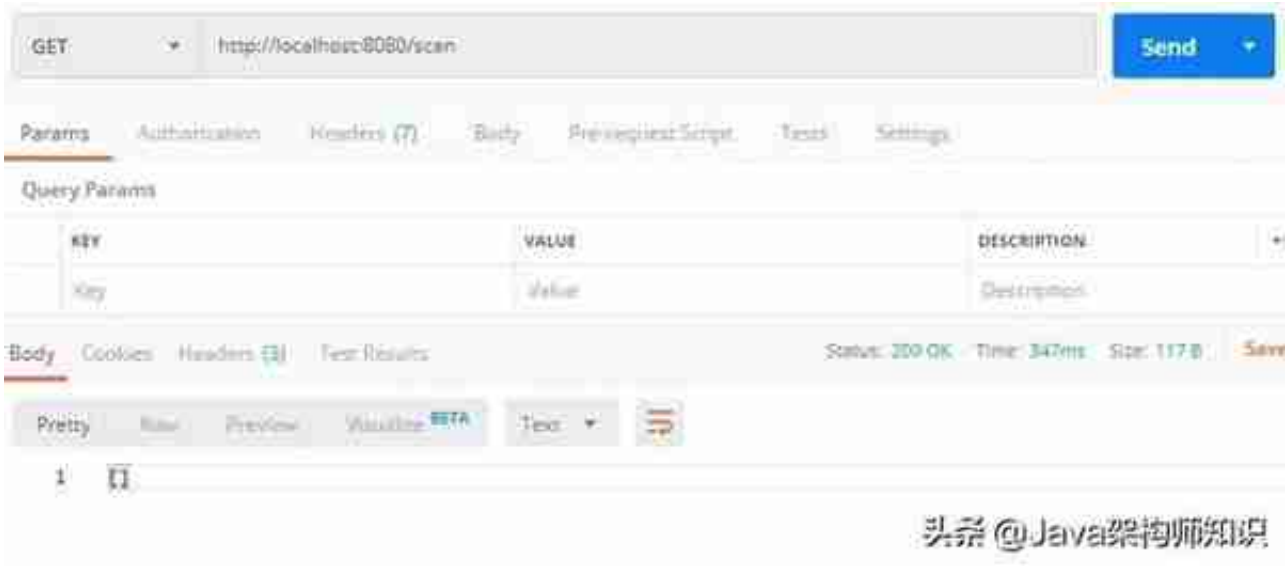
目前公有链在实际应用中并没有太多的业务场景落地，大部分都是以挖矿为主题或者线上宠物饲养的游戏为主，并且由于数字货币的匿名性，有些不法分子

利用这一特点，将数字货币用于洗钱、暗网买卖等违法行为，是各个国家的打击对象，我国政策法规也严厉禁止，因此对于技术人员来说，公有链可以作为研究学习的对象，其他方面暂时没有太多实际意义。

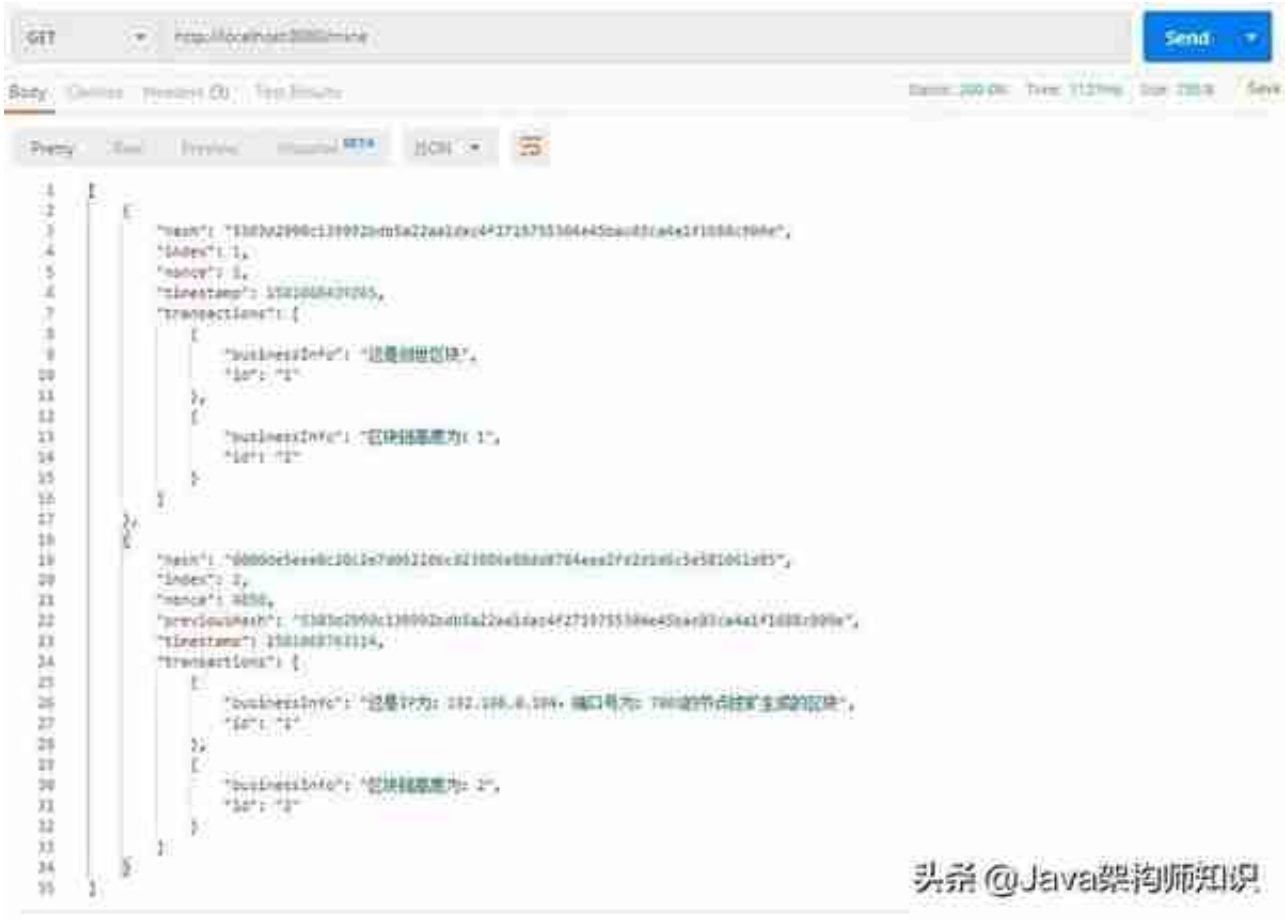
目前大部分区块链企业的研究方向主要是针对企业的联盟链和私有链，并且国家层面也在大力支持区块链技术的发展，特别是区块链底层核心技术的研发，倡导把区块链作为核心技术自主创新的重要突破口，明确主攻方向，加大投入力度，着力攻克一批关键核心技术，加快推动区块链技术和产业创新发展。不过现在市面上主流的区块链平台大部分还是以国外公司主导的为主，国内区块链底层核心技术的发展，还需要技术人员的加倍努力。

二、区块链技术Java实现

1、区块链技术架构



然后我们调用创建创世区块的方法，查看返回结果：



我们来看一下，系统后台计算的过程，此次计算共花费1048ms计算出满足条件的Hash值，共计算4850次：

	Node1	Node2
IP地址	192.168.0.104	192.168.0.112
http端口号	8080	8090
websocket服务端口号	7001	7002
websocket服务地址	ws://192.168.0.104:7001	

通过mvn package -Dmaven.test.skip=true命令对工程进行打包，生成可直接运行的jar包，打包前对工程进行配置，配置信息如下图：

节点Node1打包：

分别打包之后，生成两个节点的可执行jar包，如下：

第二步，对两个节点进行测试

首先，两个节点启动后，用postman分别执行http://192.168.0.104:8080/scan和http://192.168.0.112:8090/scan请求，可以看到两个节点的区块链内容都为空。

然后，在节点1上分别执行http://192.168.0.104:8080/create和http://192.168.0.104:8080/mine请求，来生成创世区块，以及通过挖矿产生第二个区块，执行后查看节点1的区块链信息如下：
执行
http://192.168.0.104:8080/scan结果：


```
[    {        "hash": "5303d2990c139992bdb5a22aa1dac4f271975
5304e45bac03ca4a1f1688c909e",        "index": 1,        "non
ce": 1,        "timestamp": 1581064647736,        "transacti
ons": [            {                "businessInfo": "??????"
,                "id": "1"            },            {
        "businessInfo": "????????1",                "id": "2
"            }        ]    },    {        "hash": "0000de5ee
a0c20c2e7d06220bc023886e88dd8784eaa2fd2d1d6c5e581061d85",
     "index": 2,        "nonce": 4850,        "previousHash"
: "5303d2990c139992bdb5a22aa1dac4f2719755304e45bac03ca4a1f16
88c909e",        "timestamp": 1581064655139,        "transac
tions": [            {                "businessInfo": "??IP?
?192.168.0.104??????7001??????????",                "id": "1
"            },            {                "businessInfo":
"????????2",                "id": "2"            }        ]
  }]
```


最后，我们来验证节点2是否已经完成了节点1生成的区块链信息的网络同步，Postman执行
http://192.168.0.112:8090/scan请求，查看返回结果：

反过来，我们在节点2上再执行一次挖矿操作，可以看到节点1上，已经接收到节点2挖矿新产生的区块信息，并添加到节点1的区块链上：