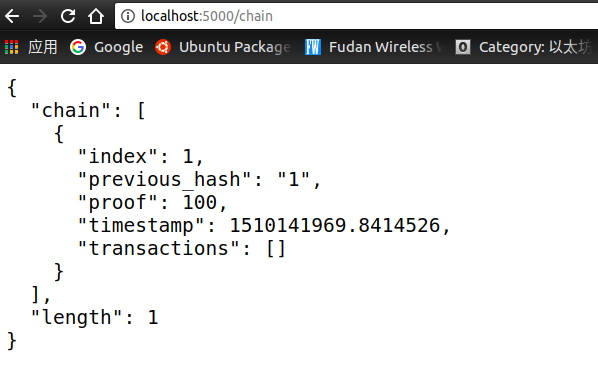
一个简易的blockchain模型

程序默认localhost：5000端口，运行时可指定端口模拟多节点

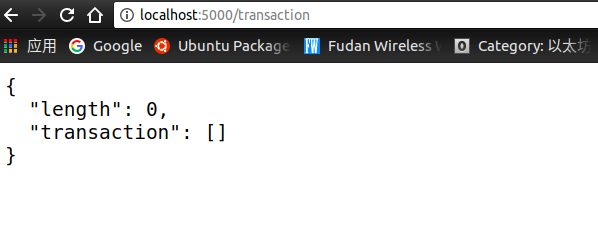
python blockchain.py -p 5001

**一、初始状态**

初始化一个创世区块：



初始交易池



**二、添加新的交易（transaction）** new\_transaction（）

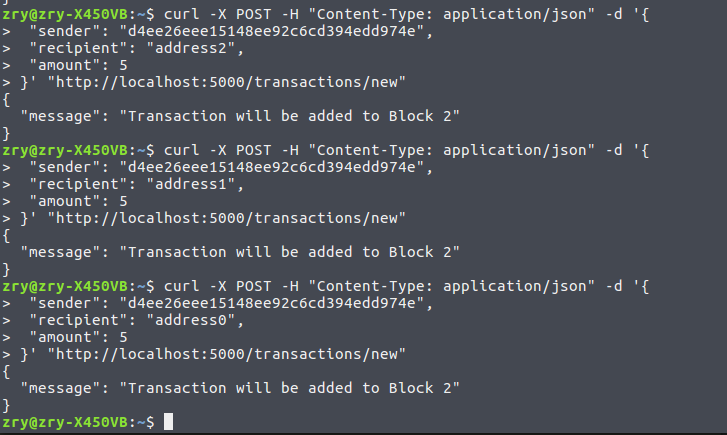
curl -X POST -H "Content-Type: application/json" -d '{

"sender": "d4ee26eee15148ee92c6cd394edd974e",

"recipient": "address2",

"amount": 5

}' "http://localhost:5000/transactions/new"

****

**新的交易池状态**

****

1. **挖矿**

1、新生成一个交易（区块的最后一个交易），将挖矿获得的奖励发送到挖矿者账户。

2、默认交易池中的交易均合法合理

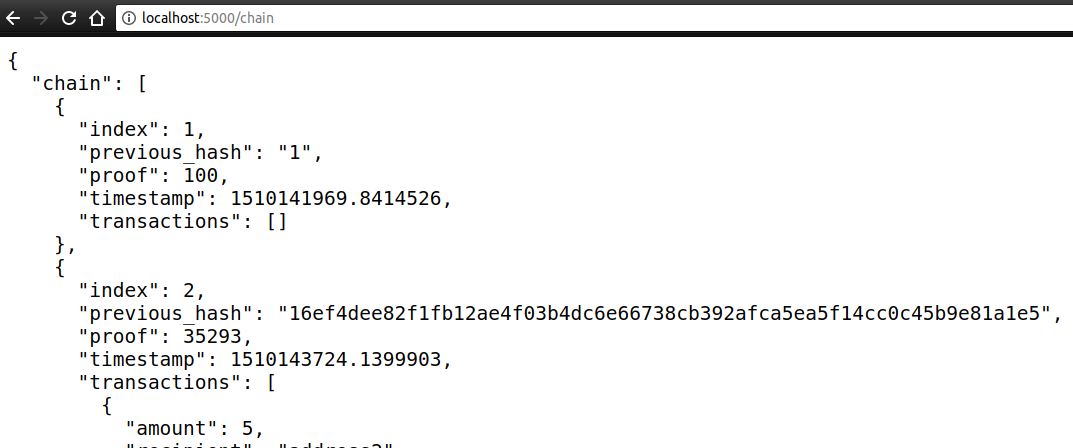
3、工作量证明，计算proof使得满足hash值的前四位为0。

proof\_of\_work（）、valid\_proof（）

4、将新的区块链接到区块链上，并清空交易池中的交易。 new\_block（）



**区块脸上新增了一个去快**



**四、实现简易的共识机制**

**1、注册新的节点**

python blockchain.py -p 5001、5002、、、每个端口认为是一个结点

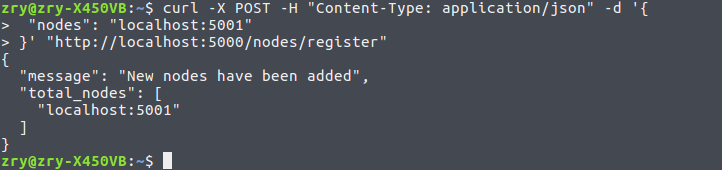
向相邻节点发送消息，注册节点

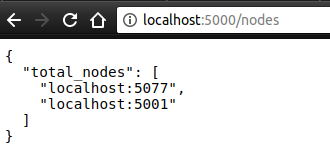
curl -X POST -H "Content-Type: application/json" -d '{

"nodes": "localhost:5001"

}' "http://localhost:5000/nodes/register"

每个节点都存储了附近存在的其他节点的信息





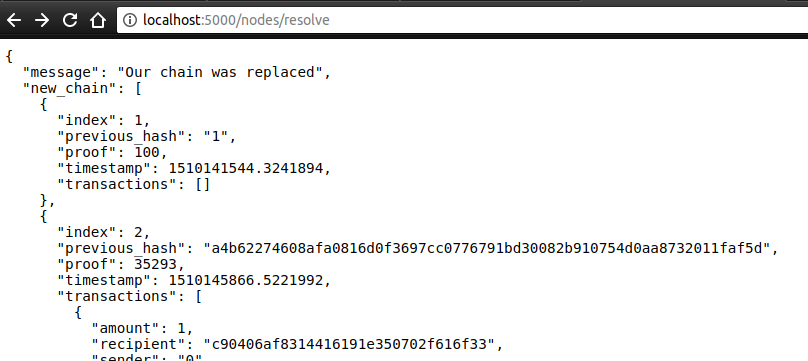
1. **实现共识算法**

方法valid\_chain（）检测区块是否合法

1. 验证区块头部previous\_hash是否与前一个区块的hash相同;
2. 检验工作量证明的随机数proof是否满足工作量的要求。

方法resolve\_conflicts()解决各个节点之间区块链数据不一致的问题

1. 获取其他节点的区块链数据，先检测其区块数据是否合法;
2. 若其他合法区块链的长度比自身区块链的长度长，则取最长的区块链替换掉自身的区块（Our chain was replaced）



否则无视其他节点的数据（Our chain is authoritative）。

