東莞理工學院
DONGGUAN UNIVERSITY OF TECHNOLOGY

区块链技术与应用

v0.11.1

第六章：共识（上）

丁烨，网络空间安全学院 副教授
dingye@dgut.edu.cn

目录

分布式账本回顾

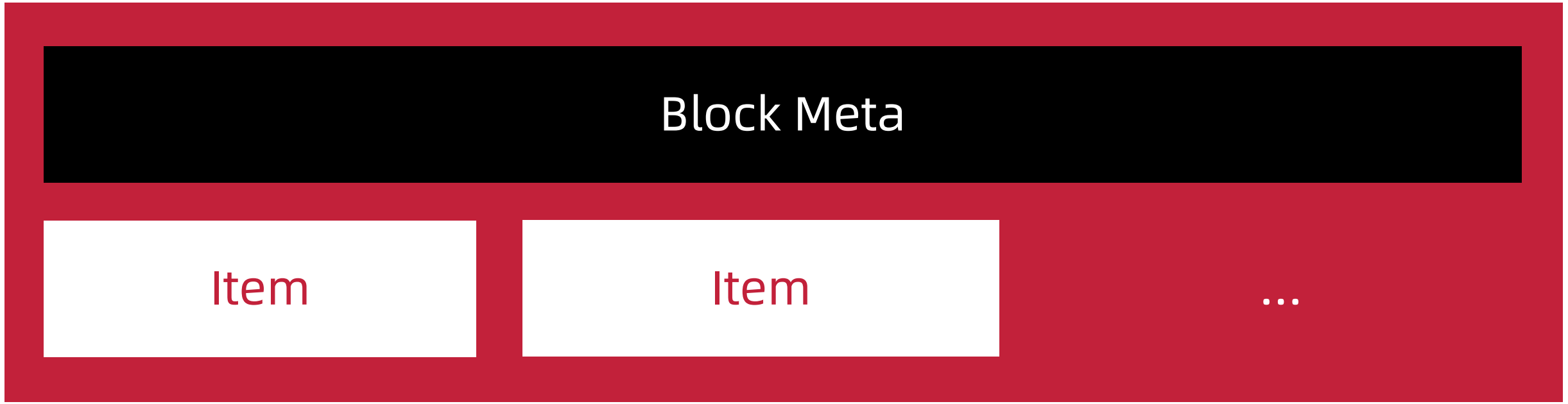
区块链核心思想

- 区块链（Blockchain）本质上是一个数据库
- 该数据库的设计目的为：安全、防止篡改
- 但是，区块链为了安全付出了运行缓慢的代价

分布式账本回顾

区块链核心思想

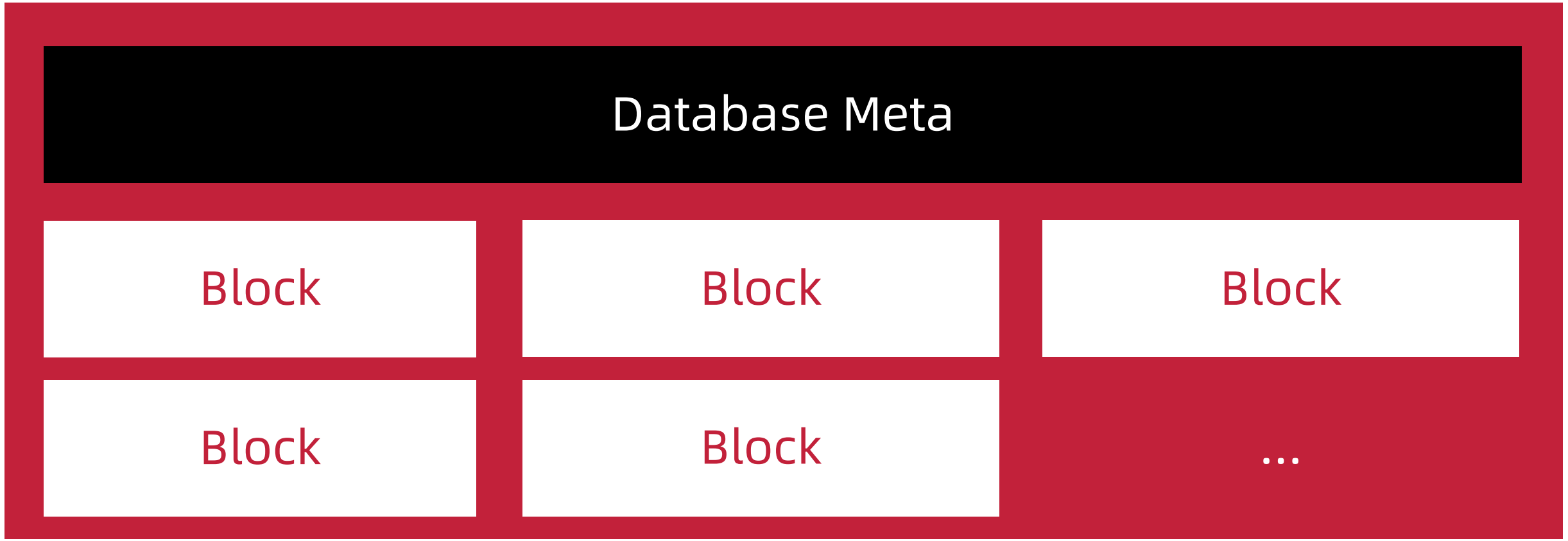
- “区块”（Block）



分布式账本回顾

区块链核心思想

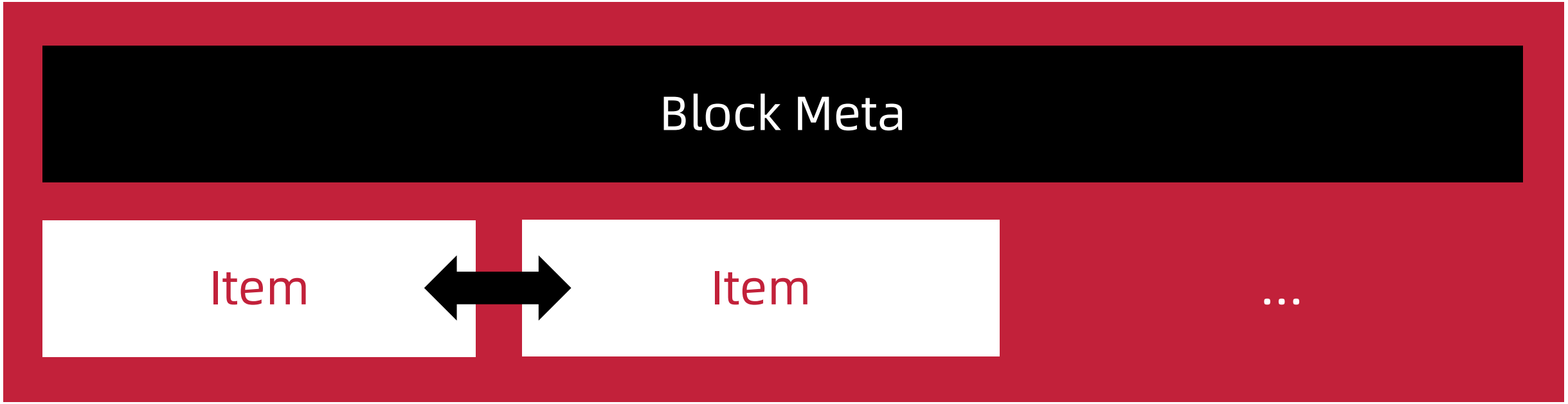
- “区块”（Block）



分布式账本回顾

区块链核心思想

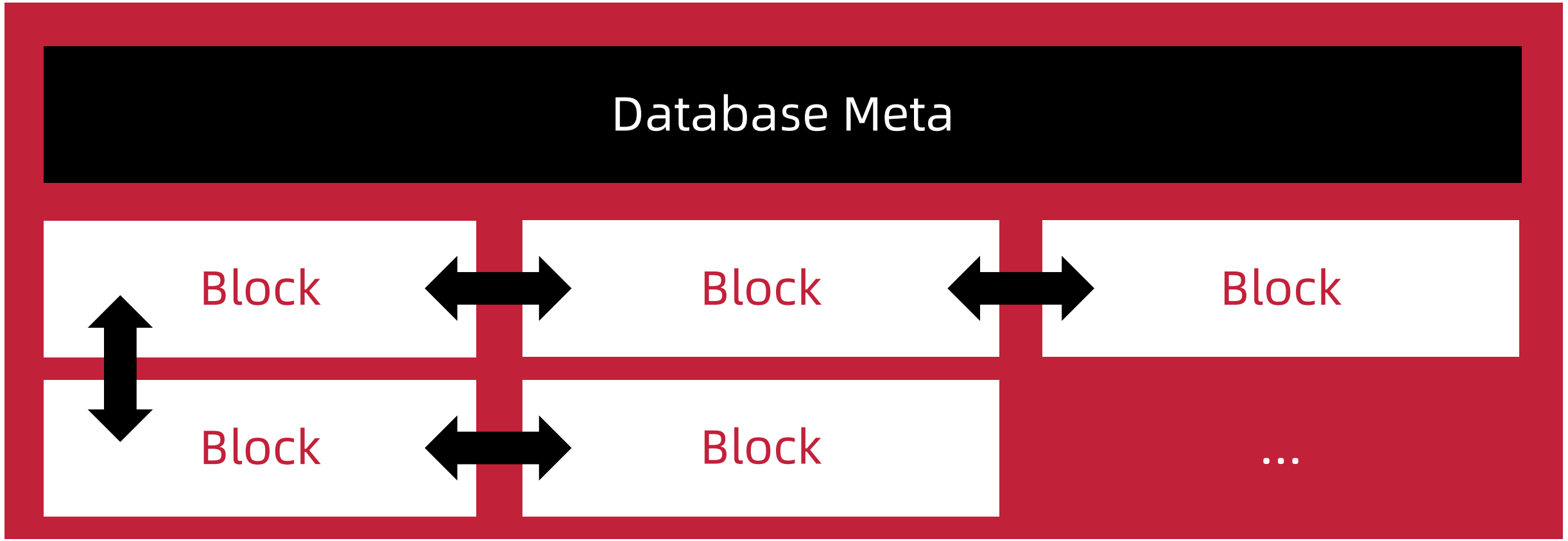
- “链”（Chain）



分布式账本回顾

区块链核心思想

- “链”（Chain）



分布式账本回顾

金融账本基础

- 交易（Transaction）
- 又称贸易、交换、互市，是买卖双方对有价物品及服务进行互通有无的行为
- 可以是以货币为交易媒介的过程，也可以是以物易物
- 在区块链账本系统中，一条记录（Item）通常为一条交易（Transaction）

分布式账本回顾

金融账本基础

- 账本（Ledger）
- 具有一定格式与若干账页组成，以会计凭证为依据，对所有经济业务进行序时分类记录的本籍，也就是通常我们所说的账册
- 简单来讲，一个账本包含多条交易记录
- 在区块链账本系统中，一个账本（Ledger）即为一个区块链（Blockchain）
- 区块（Block）相当于为账本的分页（Page）

Transactions

For Block 12063111

A total of 174 transactions found

First | Page 1 of 4 | Last

Txn Hash	Block	Age	From	To	Value	Txn Fee
0x055e5d126752c97...	12063111	47 secs ago	0x8511ae8331461cf7...	0xbacd83514447d57...	0.108 Ether	0.004641
0x06b3735e3881d0a...	12063111	47 secs ago	0x0724985d776e35a...	0xe890c54fbc11c59c...	0.1 Ether	0.004662
0x8168427d5a8c1d0...	12063111	47 secs ago	0xf3bbc4574feb7b0e...	0xa15bfa50d2169ef5...	0 Ether	0.01140502
0x52d07ccae56b3c2...	12063111	47 secs ago	0xc314996e416e3c5...	Uniswap V2: Router 2	6 Ether	0.02872129
0x944f9d6e64dbda04...	12063111	47 secs ago	0x2b68309207e40fdc...	Uniswap V2: Router 2	1.5 Ether	0.02476062
0x5399b054d520fc90...	12063111	47 secs ago	0x88d4b3d3c3794ac...	0x73d68491cbf7a147...	1.000750562922191 Ether	0.0046893
0x1697c8d8fcb92842...	12063111	47 secs ago	0xc5d55ab5c0346fb8...	0xeb953eda0dc65e3...	0 Ether	0.00991072
0xba517eb147f6c756...	12063111	47 secs ago	0x0c544f463600eb7d...	Uniswap V2: Router 2	0 Ether	0.03183977
0x40da51898380139f...	12063111	47 secs ago	0x0c544f463600eb7d...	Uniswap V2: Router 2	0 Ether	0.02853974
0xab47059b9f36891c...	12063111	47 secs ago	0xbe8292aad96770a...	0x96f7171242c9ffc1d...	0.00046308 Ether	0.0046893
0xd209d826119f9470...	12063111	47 secs ago	0x5444f07c28a378d1...	0x4a4e3ad9a80f75d2...	0.016694397916539 Ether	0.0046893
0xdbf47fd8504a06ce...	12063111	47 secs ago	0xa0b8df820341b79b...	Stacktical: DSLA Token	0 Ether	0.01052412

分布式账本回顾

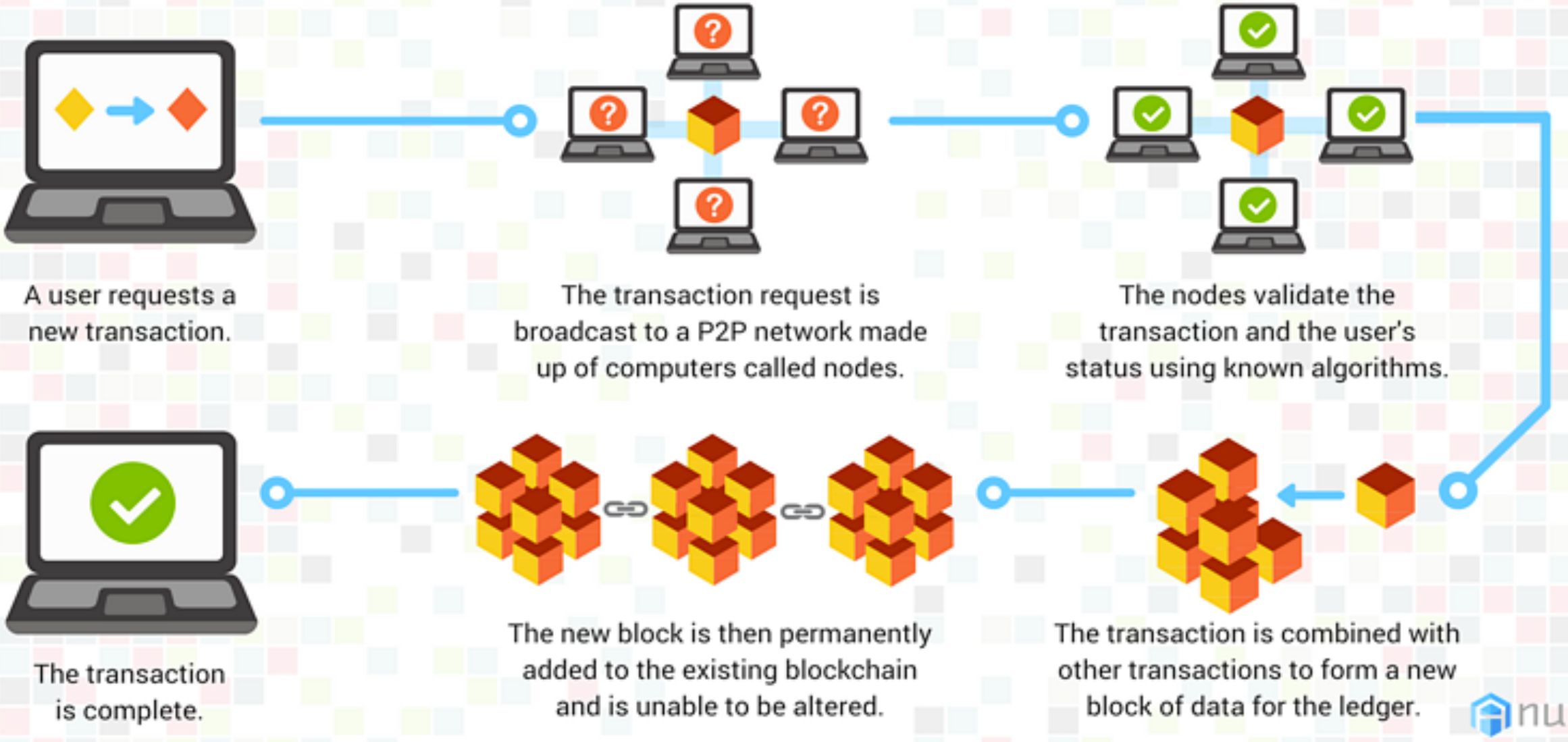
分布式账本特点

- 分布式账本（Distributed Ledger）
- 又称共享账本（Shared Ledger）、分散式账本技术（Distributed ledger Technology，DLT）
- 是一个由多站点、多国家或多家机构所组成的在网络上进行电子数据复制、共享及同步的共识机制
- 通常不依赖也不存在中央管理员或集中的数据存储
- 对等网络与共识机制确保了跨节点间的数据能被正确复制
- 区块链系统是分布式账本的一种实现方法

分布式账本概述

分布式账本特点

- 成本低（Cheap）
- 完全电子化，无人管理
- 可信度高（Trustworthy）
- 通过共识机制保证数据无法篡改（Immutable）
- 匿名（Anonymous）
- 从机制上保证安全性，无须验证使用者的社会信用


A user requests a new transaction.
The transaction request is broadcast to a P2P network made up of computers called nodes.
The nodes validate the transaction and the user's status using known algorithms.
The transaction is combined with other transactions to form a new block of data for the ledger.
The new block is then permanently added to the existing blockchain and is unable to be altered.
The transaction is complete.
nura

分布式账本回顾

分布式账本工作流程

1. 本地创建交易记录（Transaction）
2. 传送给分布式账本，存放于记录池（Pool）
3. 分布式账本节点之间通过共识不断同步记录池
4. 不断尝试让记录池满足一定条件（例如，完成写入证明）
5. 校验记录池中的交易记录并生成区块（Block）
6. 加入新的区块并生成新的区块链
7. 分布式账本节点之间通过共识不断同步区块链

分布式账本回顾

分布式账本工作流程

1. 本地创建交易记录（Transaction）
2. 传送给分布式账本，存放于记录池（Pool）
3. 分布式账本节点之间通过共识不断同步记录池
4. 不断尝试让记录池满足一定条件（例如，完成写入证明）
5. 校验记录池中的交易记录并生成区块（Block）
6. 加入新的区块并生成新的区块链
7. 分布式账本节点之间通过共识不断同步区块链

目录

- 分布式账本回顾
- 实现支持共识的分布式账本

实现支持共识的分布式账本

修改 P2P 网络服务并使其支持区块链的记录池

- app.py

```
from ..blockchain.ledger import Ledger
from ..blockchain.transaction import Transaction
```

实现支持共识的分布式账本

修改 P2P 网络服务并使其支持区块链的记录池

- app.py

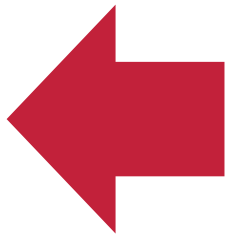
```
def on_message(self, message):
    ...
    elif message['op'] == 'pool':
        self.write_message(json.dumps({
            'status': 200,
            'error': 'OK',
            'response': {
                'pool': [json.loads(str(i)) for i in list(pickle.loads(db.get('pool')))]
            }
        }))
```

实现支持共识的分布式账本

修改 P2P 网络服务并使其支持区块链的记录池

- app.py

```
elif message['op'] == 'merge':
    if 'args' in message and 'pool' in message['args']:
        pool = pickle.loads(db.get('pool'))
        for i in message['args']['pool']:
            pool.add(Transaction(
                sender=i['sender'],
                receiver=i['receiver'],
                amount=i['amount']
            ))
        db.set('pool', pickle.dumps(pool))
```



实现支持共识的分布式账本

修改 P2P 网络服务并使其支持区块链的记录池

- app.py

```
elif message['op'] == 'merge':
    if 'args' in message and 'pool' in message['args']:
        ...
        self.write_message(json.dumps({
            'status': 202,
            'error': 'Accepted'
        }))
```

实现支持共识的分布式账本

修改 P2P 网络服务并使其支持区块链的记录池

❖ app.py

```
elif message['op'] == 'merge':
    if 'args' in message and 'pool' in message['args']:
        ...
    else:
        self.write_message(json.dumps({
            'status': 500,
            'error': 'Operation "merge" requires the following "args": "pool"',
            'response': None
        }))
```

实现支持共识的分布式账本

修改 P2P 网络服务并使其支持区块链的记录池

- app.py

```
if __name__ == '__main__':
    ...
    db = Redis(args.redis)
    if b'peers' not in db.keys():
        db.set('peers', pickle.dumps(set([])))
    if b'pool' not in db.keys():
        db.set('pool', pickle.dumps(set([])))
    ...
```

实现支持共识的分布式账本

修改 P2P 网络服务并使其支持区块链的记录池

```
~/Workspace/foxchain(master*) » python3 -u -m foxchain.app.app
[2021-04-26 23:36:05,378] Tornado is listening on port: 9000
```

HISTORY

Clear

- Request #36
- Request #35
- Request #34
- Request #33
- Request #32
- Request #31
- Request #30
- Request #29

Everything is ok, ready to go!

ws://localhost:9000/ws

Disconnect

Send

```
{
  "op": "pool"
}
```

```
{
  "status": 200,
  "error": "OK",
  "response": {
    "pool": []
  }
}
```

HISTORY

Clear

- Request #37
- Request #36
- Request #35
- Request #34
- Request #33
- Request #32
- Request #31
- Request #30
- Request #29

Everything is ok, ready to go!

ws://localhost:9000/ws

Disconnect

Send

```
{
  "op": "merge",
  "args": {
    "pool": [{
      "sender": "00000000-0000-0000-0000-000000000000",
      "receiver": "bc70b89d-4521-454c-bebc-6c2456074bf2",
      "amount": 10000
    }]
  }
}
```

```
{
  "status": 202,
  "error": "Accepted"
}
```

HISTORY

Clear

- Request #38
- Request #37
- Request #36
- Request #35
- Request #34
- Request #33
- Request #32
- Request #31
- Request #30
- Request #29

Everything is ok, ready to go!

ws://localhost:9000/ws

Disconnect

Send

```
{
  "op": "pool"
}
```

```
{
  "status": 200,
  "error": "OK",
  "response": {
    "pool": [{
      "sender": "00000000-0000-0000-0000-000000000000",
      "receiver": "bc70b89d-4521-454c-bebc-6c2456074bf2",
      "amount": 10000,
      "t": 1619451754.012226,
      "prev_hash": null
    }]
  }
}
```

实现支持共识的分布式账本

生成区块链

- builder.py

```
import argparse
import logging
import pickle
import time
import traceback

from redislite import Redis
from wrenchbox.logging import setup_log

from ..blockchain.blockchain import Block
from ..blockchain.ledger import GENESIS
```

实现支持共识的分布式账本

生成区块链

- builder.py

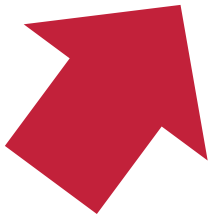
```
class Builder:
    def __init__(self, url):
        self.db = Redis(url)
```

实现支持共识的分布式账本

生成区块链

- builder.py

```
def run(self, k_size, cool_down):
    while True:
        pool = pickle.loads(self.db.get('pool'))
        if len(pool) >= k_size:
            self.db.set('pool', pickle.dumps(set([])))
            logging.info('Pool is cleared.')
            logging.info('Packing %d transactions...', len(pool))
            ...
        else:
            logging.debug('Currently %d records, require: %d', len(pool), k_size)
        time.sleep(cool_down)
```



实现支持共识的分布式账本

生成区块链

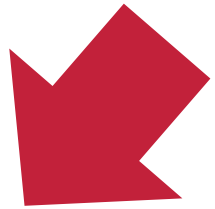
- builder.py

```
ledger = pickle.loads(self.db.get('ledger'))
block = Block()
for transaction in pool:
    if transaction.sender == GENESIS \
            or ledger.balance(transaction.sender) >= transaction.amount:
        block.add(transaction)
    else:
        logging.warning('Dropped transaction due to not enough balance: %s', transaction)
logging.info('Block is created with %d records.', len(block.items))
```

实现支持共识的分布式账本

生成区块链

- builder.py



```
if len(block.items):
    try:
        block.validate()
    except AssertionError:
        logging.error('Block is invalid and dropped.')
        if args.debug:
            traceback.print_exc()
    else:
        ...
```

实现支持共识的分布式账本

生成区块链

- builder.py

```
ledger.add(block)
try:
    ledger.validate()
except AssertionError:
    logging.error('Ledger is invalid and dropped.')
    if args.debug:
        traceback.print_exc()
else:
    self.db.set('ledger', pickle.dumps(ledger))
    logging.info('Block is added to the blockchain.')
```

实现支持共识的分布式账本

生成区块链

❖ builder.py

```
if __name__ == '__main__':
    parser = argparse.ArgumentParser()
    parser.add_argument(
        '--debug',
        action='store_true', default=False,
        help='show debug information'
    )
    ...
    args, _ = parser.parse_known_args()
    setup_log(level=logging.DEBUG if args.debug else logging.INFO)
    Builder(args.redis).run(args.size, args.sleep)
```

实现支持共识的分布式账本

生成区块链

- builder.py

```
parser.add_argument(
    '-r', '--redis',
    type=str, default='redis.db',
    help='redis database file, default: redis.db'
)
```

实现支持共识的分布式账本

生成区块链

- builder.py

```
parser.add_argument(
    '-k', '--size',
    type=int, default=3,
    help='# of minimum packed transactions, default: 3'
)
```

实现支持共识的分布式账本

生成区块链

- builder.py

```
parser.add_argument(
    '-t', '--sleep',
    type=int, default=3,
    help='refresh rate in seconds, default: 3'
)
```

实现支持共识的分布式账本

生成区块链

```
~/Workspace/foxchain(master*) » python3 -u -m foxchain.app.builder
```

实现支持共识的分布式账本

修改 P2P 网络服务并使其支持查询区块链

- app.py

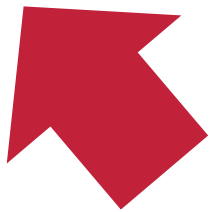
```
elif message['op'] == 'blocks':
    blocks = json.loads(str(pickle.loads(db.get('ledger'))))['blocks']
    if 'args' in message and 'start' in message['args']:
        blocks = blocks[int(message['args']['start']):]
    self.write_message(json.dumps({
        'status': 200,
        'error': 'OK',
        'response': {
            'blocks': blocks
        }
    }))
```

实现支持共识的分布式账本

修改 P2P 网络服务并使其支持查询区块链

- app.py

```
if __name__ == '__main__':
    ...
    db = Redis(args.redis)
    if b'peers' not in db.keys():
        db.set('peers', pickle.dumps(set([])))
    if b'pool' not in db.keys():
        db.set('pool', pickle.dumps(set([])))
    if b'ledger' not in db.keys():
        db.set('ledger', pickle.dumps(Ledger()))
    ...
```



实现支持共识的分布式账本

修改 P2P 网络服务并使其支持查询区块链

```
~/Workspace/foxchain(master*) » python3 -u -m foxchain.app.app
[2021-04-26 23:36:05,378] Tornado is listening on port: 9000
```

HISTORY

Clear

- Request #51
- Request #50
- Request #49
- Request #48
- Request #47
- Request #46
- Request #45
- Request #44
- Request #43
- Request #42
- Request #41
- Request #40
- Request #39

Everything is ok, ready to go!

ws://localhost:9000/ws

Disconnect

Send

```
{
  "op": "blocks"
}
```

```
{
  "status": 200,
  "error": "OK",
  "response": {
    "blocks": []
  }
}
```

HISTORY

Clear

- Request #38
- Request #37
- Request #36
- Request #35
- Request #34
- Request #33
- Request #32
- Request #31
- Request #30
- Request #29

Everything is ok, ready to go!

ws://localhost:9000/ws

Disconnect

Send

```
{
  "op": "pool"
}
```

```
{
  "status": 200,
  "error": "OK",
  "response": {
    "pool": [{
      "sender": "00000000-0000-0000-0000-000000000000",
      "receiver": "bc70b89d-4521-454c-bebc-6c2456074bf2",
      "amount": 10000,
      "t": 1619451754.012226,
      "prev_hash": null
    }]
  }
}
```

HISTORY

Clear

- Request #39
- Request #38
- Request #37
- Request #36
- Request #35
- Request #34

Everything is ok, ready to go!

ws://localhost:9000/ws

Disconnect

Send

```
{
  "op": "merge",
  "args": {
    "pool": [{
      "sender": "00000000-0000-0000-0000-000000000000",
      "receiver": "bc70b89d-4521-454c-bebc-6c2456074bf1",
      "amount": 8000
    }]
  }
}
```

```
{
  "status": 202,
  "error": "Accepted"
}
```

HISTORY

Clear

- Request #40
- Request #39
- Request #38
- Request #37
- Request #36
- Request #35
- Request #34

Everything is ok, ready to go!

ws://localhost:9000/ws

Disconnect

Send

```
{
  "op": "merge",
  "args": {
    "pool": [{
      "sender": "00000000-0000-0000-0000-000000000000",
      "receiver": "bc70b89d-4521-454c-bebc-6c2456074bf0",
      "amount": 5000
    }]
  }
}
```

```
{
  "status": 202,
  "error": "Accepted"
}
```

HISTORY

Clear

- Request #48
- Request #47
- Request #46
- Request #45
- Request #44
- Request #43
- Request #42
- Request #41
- Request #40
- Request #39
- Request #38
- Request #37
- Request #36

Everything is ok, ready to go!

ws://localhost:9000/ws

Disconnect

Send

```
{
  "op": "pool"
}
```


```
{
  "status": 200,
  "error": "OK",
  "response": {
    "pool": [{
      "sender": "00000000-0000-0000-0000-000000000000",
      "receiver": "bc70b89d-4521-454c-bebc-6c2456074bf2",
      "amount": 10000,
      "t": 1619452890.405376,
      "prev_hash": null
    }, {
      "sender": "00000000-0000-0000-0000-000000000000",
      "receiver": "bc70b89d-4521-454c-bebc-6c2456074bf1",
      "amount": 8000,
      "t": 1619452895.812331,
```

HISTORY

Clear

- Request #49
- Request #48
- Request #47
- Request #46
- Request #45
- Request #44
- Request #43
- Request #42
- Request #41
- Request #40
- Request #39
- Request #38
- Request #37

Everything is ok, ready to go!

ws://localhost:9000/ws

Disconnect

Send

```
{
  "op": "pool"
}
```

```
{
  "status": 200,
  "error": "OK",
  "response": {
    "pool": []
  }
}
```

HISTORY

Clear

- Request #50
- Request #49
- Request #48
- Request #47
- Request #46
- Request #45
- Request #44
- Request #43
- Request #42
- Request #41
- Request #40
- Request #39
- Request #38

Everything is ok, ready to go!

ws://localhost:9000/ws

Disconnect

Send

```
{
  "op": "blocks"
}
```


```
{
  "status": 200,
  "error": "OK",
  "response": {
    "blocks": [{
      "items": [{
        "sender": "00000000-0000-0000-0000-000000000000",
        "receiver": "bc70b89d-4521-454c-bebc-6c2456074bf1",
        "amount": 8000,
        "t": 1619452895.812331,
        "prev_hash": null
      }, {
        "sender": "00000000-0000-0000-0000-000000000000",
        "receiver": "bc70b89d-4521-454c-bebc-6c2456074bf2",
        "amount": 10000,
```

实现支持共识的分布式账本

生成区块链

```
------------------------------------------------------------------------
~/Workspace/foxchain(master*) » python3 -u -m foxchain.app.builder
[2021-04-27 00:06:25,074] Pool is cleared.
[2021-04-27 00:06:25,075] Packing 3 transactions...
[2021-04-27 00:06:25,075] Block is created with 3 records.
[2021-04-27 00:06:25,075] Block is added to the blockchain.
```

实现支持共识的分布式账本

实验课安排

- 实验四：共识（上）
- 2021-11-19
- 10:25-12:00
- 9A-206-1

- 请务必自带电脑
- 电脑需要预装好 Python 3.10

总结

- 袁煜明《区块链技术进阶指南》，机械工业出版社
- http://product.dangdang.com/28538836.html

- 安德烈亚斯·安东诺普洛斯《精通区块链编程：加密货币原理、方法和应用开发》第二版，机械工业出版社
- http://product.dangdang.com/27877333.html

总结

- Bitcoin 比特币
- https://bitcoin.org

- Ethereum 以太坊
- https://ethereum.org


师范无小事
处处

東莞理工學院
DONGGUAN UNIVERSITY OF TECHNOLOGY

Thank You!

丁烨，网络空间安全学院 副教授
dingye@dgut.edu.cn