東莞理工學院
DONGGUAN UNIVERSITY OF TECHNOLOGY

区块链技术与应用

v0.10.18

第三章：分布式账本

丁烨，网络空间安全学院 副教授
dingye@dgut.edu.cn

目录

分布式账本概述

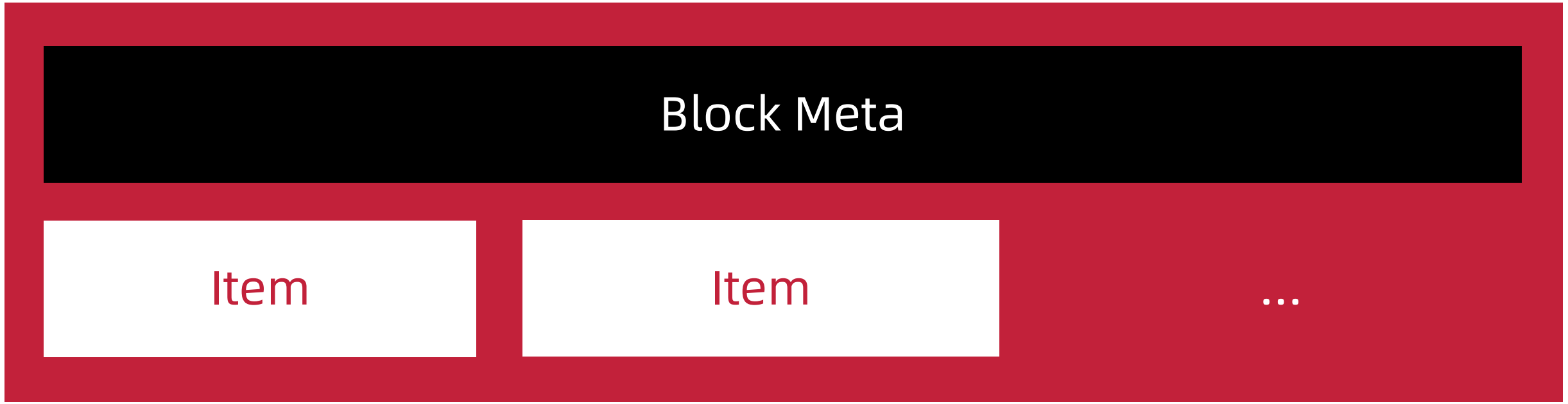
区块链核心思想

- 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

分布式账本概述

区块链核心思想

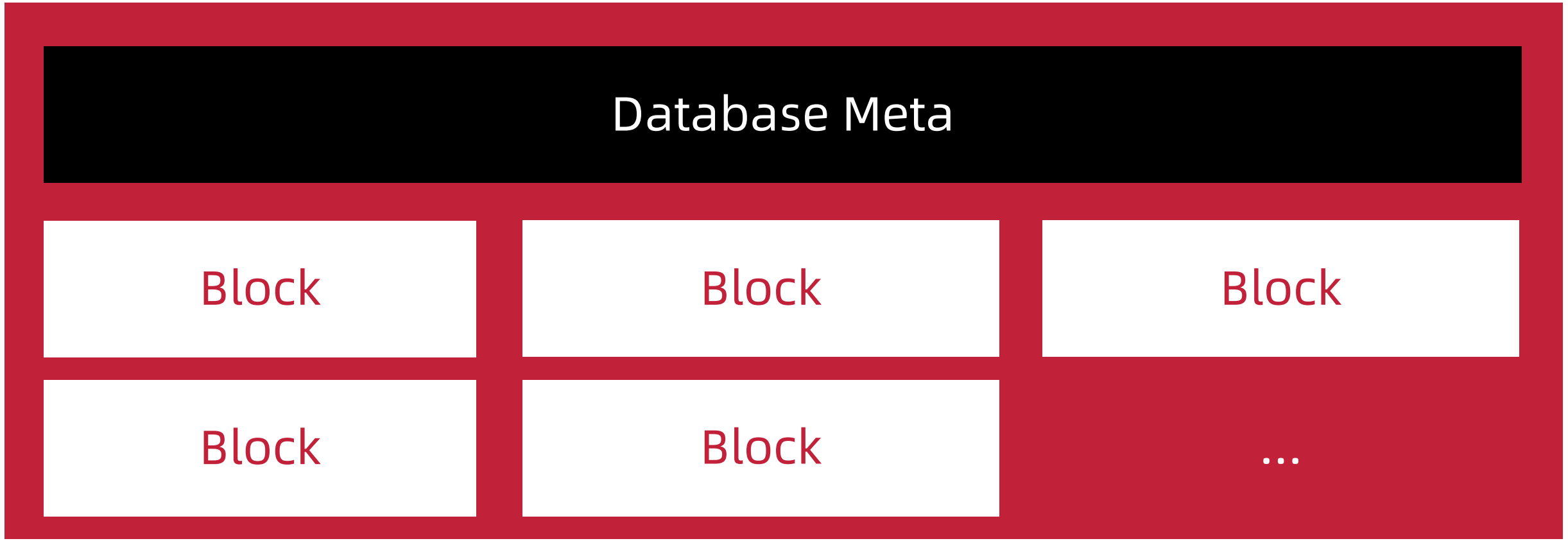
- “区块”（Block）



分布式账本概述

区块链核心思想

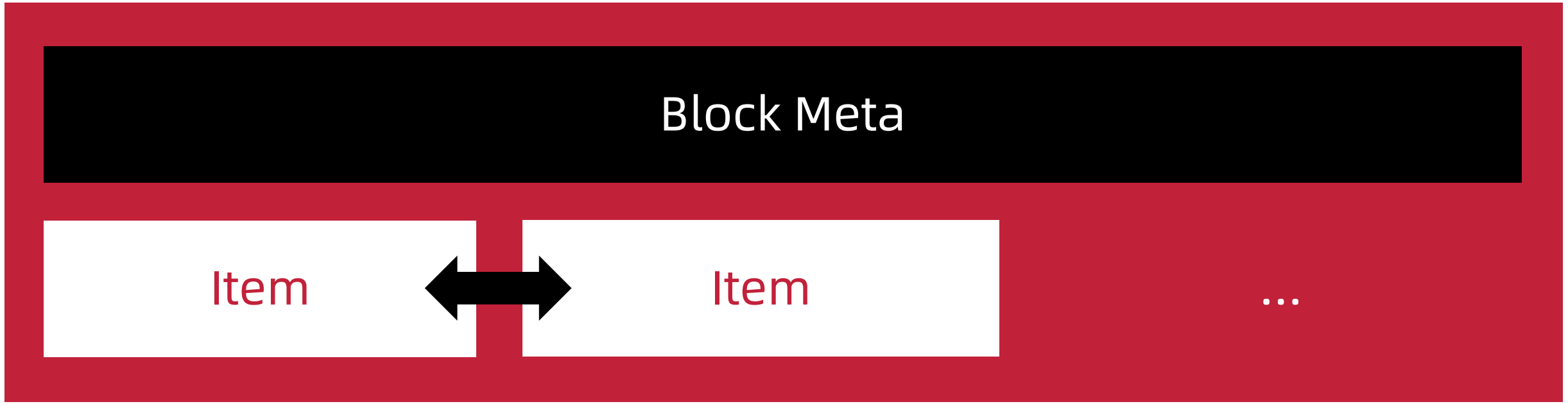
- “区块”（Block）



分布式账本概述

区块链核心思想

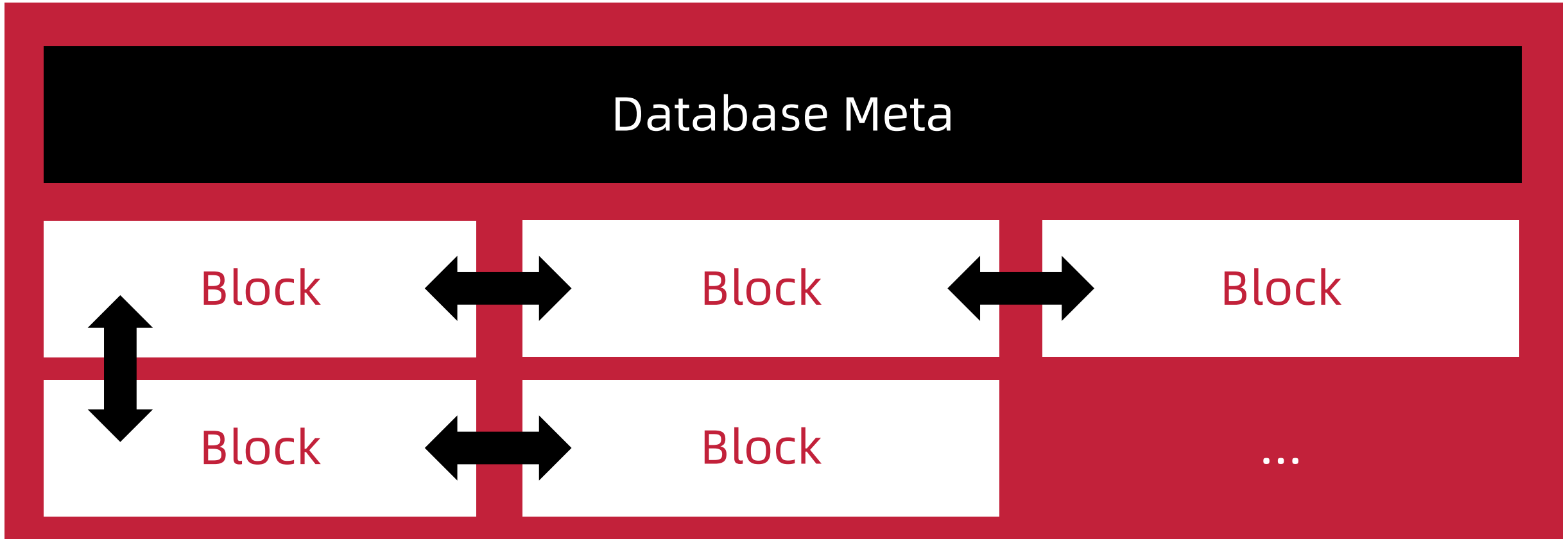
- “链”（Chain）



分布式账本概述

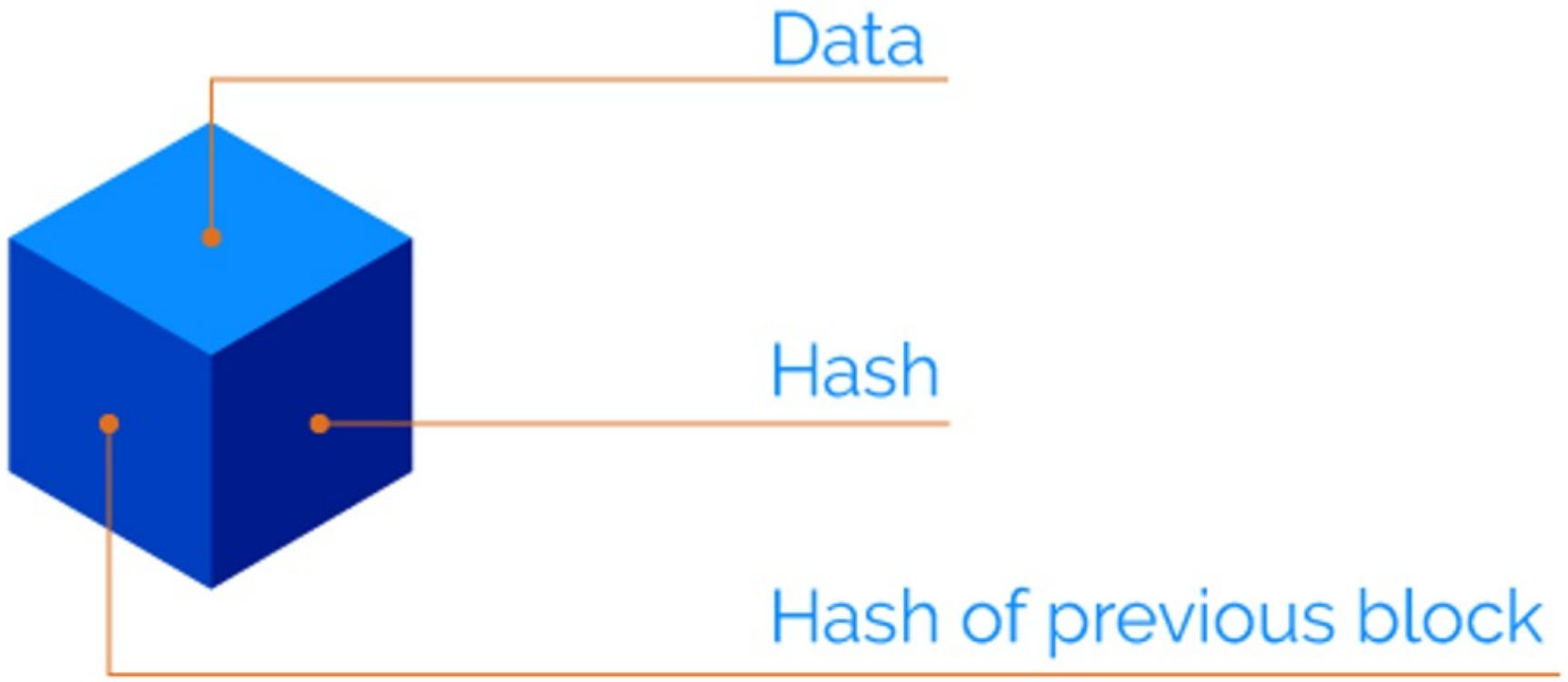
区块链核心思想

- “链”（Chain）



分布式账本概述

区块链核心思想



分布式账本概述

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）

分布式账本概述

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）

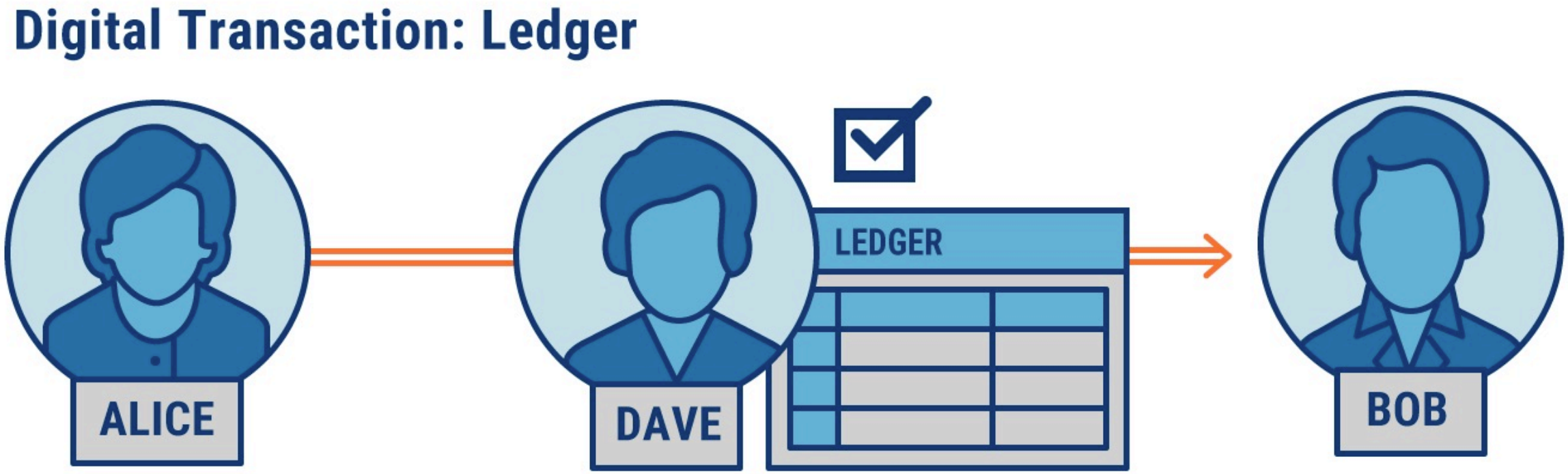
分布式账本概述

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

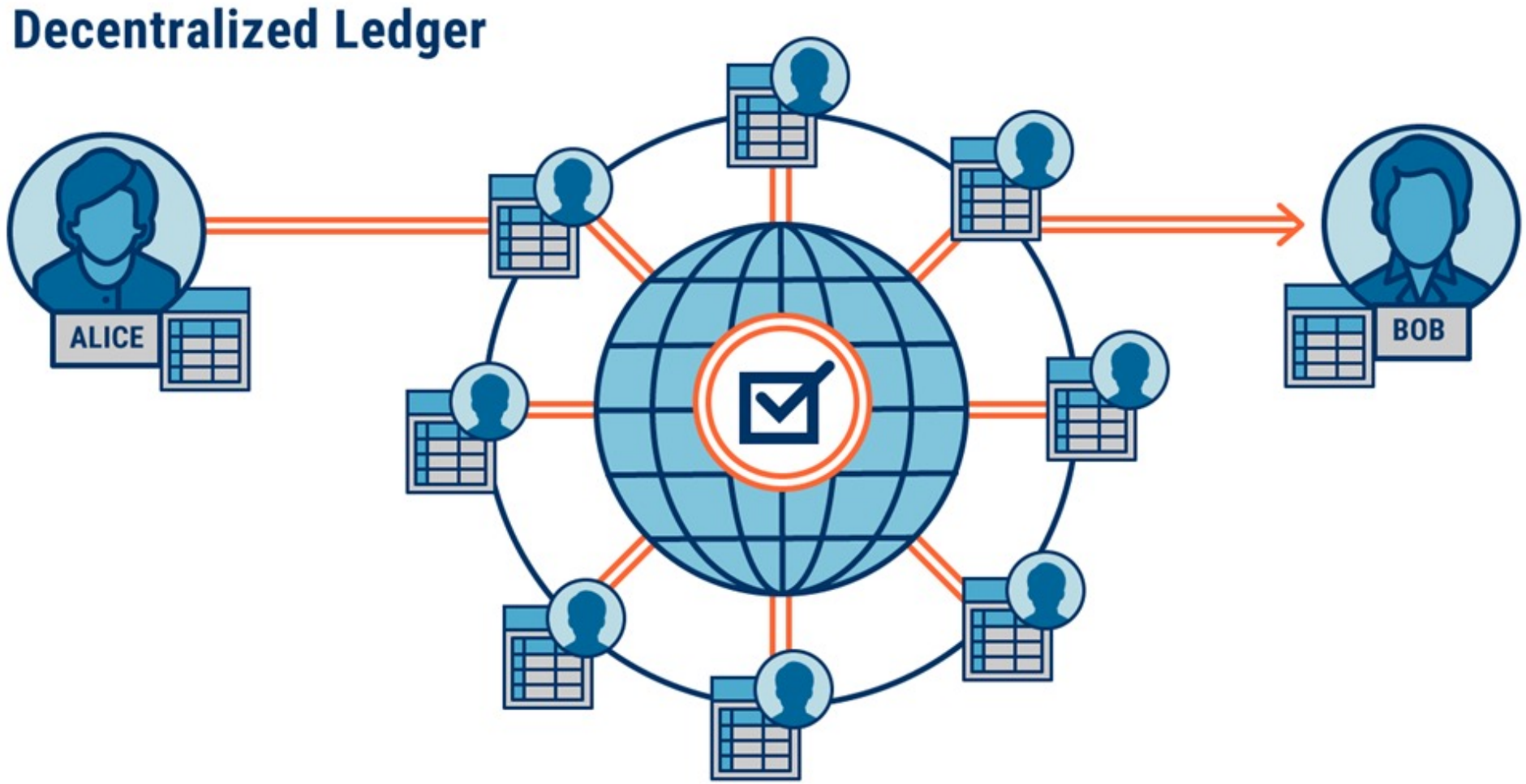
分布式账本概述

分布式账本特点



分布式账本概述

分布式账本特点



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

目录

- 分布式账本概述
- 实现简单的分布式账本

实现简单的分布式账本

编程基础

https://websitesetup.org/wp-content/uploads/2020/04/Python-Cheat-Sheet.pdf
https://perso.limsi.fr/pointal/_media/python:cours:mementopython3-english.pdf

实现简单的分布式账本

可做散列函数的基类

- basic.py

```
import hashlib
from wrenchbox.object import Dict2StrSafe

class Base(Dict2StrSafe):
    def hash(self):
        return hashlib.sha1(str(self.__dict__).encode()).hexdigest()
```

实现简单的分布式账本

可做散列函数的基类

- Hashlib
- https://docs.python.org/3/library/hashlib.html
- Python 原生功能，提供了常见的散列函数

实现简单的分布式账本

可做散列函数的基类

```
~/Workspace » python3
Python 3.8.5 (default, Jan 27 2021, 15:41:15)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import hashlib
>>> hashlib.sha224(b"Nobody inspects the spammish repetition").hexdigest()
'a4337bc45a8fc544c03f52dc550cd6e1e87021bc896588bd79e901e2'
>>>
```

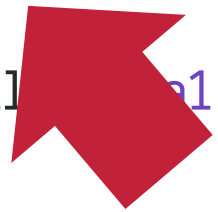
实现简单的分布式账本

可做散列函数的基类

- basic.py

```
import hashlib
from wrenchbox.object import Dict2StrSafe

class Base(Dict2StrSafe):
    def hash(self):
        return hashl      1(str(self.__dict__).encode()).hexdigest()
```



实现简单的分布式账本

可做散列函数的基类

- Wrenchbox
- https://github.com/valency/wrenchbox
- 提供了一些常用的 Python 工具

实现简单的分布式账本

可做散列函数的基类

- Dict2StrSafe
- https://github.com/valency/wrenchbox/blob/master/wrenchbox/object.py
- 使得 `__str__` 函数能够循环调用内部对象的 `__dict__` 函数
- 最终使得 `__str__` 函数能够输出容易理解的字符串

实现简单的分布式账本

可做散列函数的基类

```python
class A:
    def __init__(self):
        self.x = 1

class B:
    def __init__(self):
        self.a = A()

b = B()
print(b)
```

```
~/Downloads/foxchain-master » python3 test2.py
<__main__.B object at 0x7f59cd6ad9d0>
```

实现简单的分布式账本

可做散列函数的基类

```python
from wrenchbox.object import Dict2StrSafe

class A1(Dict2StrSafe):
    def __init__(self):
        self.x = 1

class B1(Dict2StrSafe):
    def __init__(self):
        self.a = A1()

b = B1()
print(b)
```

```
~/Downloads/foxchain-master » python3 test2.py
{"a": {"x": 1}}
```

实现简单的分布式账本

可做散列函数的基类

- basic.py

```
import hashlib
from wrenchbox.object import Dict2StrSafe

class Base(Dict2StrSafe):
    def hash(self):
        return hashlib.sha1(str(self.__dict__).encode()).hexdigest()
```

实现简单的分布式账本

交易记录类

- transaction.py

```
from datetime import datetime
from uuid import UUID

from .basic import Base

class Transaction(Base):
    def __init__(self,
                 sender: str, receiver: str,
                 amount: int, t: float = None,
                 prev_hash: str = None):
        ...
```

实现简单的分布式账本

交易记录类

- transaction.py

```
class Transaction(Base):
    def __init__(self, …):
        assert UUID(sender, version=4)
        assert UUID(receiver, version=4)
        self.sender = sender
        self.receiver = receiver
        self.amount = amount
        self.t = t if t is not None else datetime.now().timestamp()
        self.prev_hash = prev_hash
```

实现简单的分布式账本

区块链类

- blockchain.py

```
from .basic import Base

class Block(Base):
    def __init__(self, prev_hash: str = None):
        self.items = []
        self.prev_hash = prev_hash

    ...
```

实现简单的分布式账本

区块链类

- blockchain.py

```
class Block(Base):
    ...

    def add(self, item):
        item.prev_hash = self.items[-1].hash() if len(self.items) else None
        self.items.append(item)

    ...
```

实现简单的分布式账本

区块链类

- blockchain.py

```
class Block(Base):
    ...

    def validate(self):
        for i in range(len(self.items)):
            assert i == 0 or self.items[i].prev_hash == self.items[i - 1].hash()
```

实现简单的分布式账本

区块链类

- blockchain.py

```
class Blockchain(Base):
    def __init__(self):
        self.blocks = []

    def add(self, block):
        block.prev_hash = self.blocks[-1].hash() if len(self.blocks) else None
        self.blocks.append(block)

    ...
```

实现简单的分布式账本

区块链类

- blockchain.py

```
class Blockchain(Base):
    ...

    def validate(self):
        for i in range(len(self.blocks)):
            assert i == 0 or self.blocks[i].prev_hash == self.blocks[i - 1].hash()
            self.blocks[i].validate()
```

实现简单的分布式账本

测试本地账本

- test.py

```
import json
import random
from uuid import uuid4

from foxchain.blockchain import Blockchain, Block
from foxchain.transaction import Transaction

if __name__ == '__main__':
    chain = Blockchain()
    ...
```

实现简单的分布式账本

测试本地账本

- test.py

```
...
if __name__ == '__main__':
    chain = Blockchain()
    for i in range(3):
        block = Block()
        for j in range(10):
            transaction = Transaction(sender=str(uuid4()), receiver=str(uuid4()),
                amount=random.randint(1, 1000000)
            )
            block.add(transaction)
        chain.add(block)
    ...
```

实现简单的分布式账本

测试本地账本

- test.py

```
...
if __name__ == '__main__’:
    ...
    print(json.dumps(json.loads(str(chain)), indent=4, sort_keys=True))
    print(chain.validate())
    ...
```

```
~/Downloads/foxchain-master » python3 test.py                    valency@Aorus-Master
{
    "blocks": [
        {
            "items": [
                {
                    "amount": 356347,
                    "prev_hash": null,
                    "receiver": "37ef4c46-ff77-403f-9e6d-98ba2bf1dee6",
                    "sender": "fcad5575-b26c-428b-9838-19ffd61550d2",
                    "t": 1616080992.596395
                },
                {
                    "amount": 81010,
                    "prev_hash": "a591128e038be3e10ee2bcf9a5958d4161161aca",
                    "receiver": "0aec5738-92d5-4476-95c7-adadbd2a2fc3",
                    "sender": "c477faa2-745c-44af-bc13-6473161022b6",
                    "t": 1616080992.596419
                },
```

```
            "amount": 176919,
            "prev_hash": "ca2ebeb7800be3c9ed5ceb71cc9d2a2b4afc051e",
            "receiver": "68e197cf-6d78-408d-8930-16be4de080f6",
            "sender": "e2ad571f-bf7c-4aef-a440-e4cfe4c93359",
            "t": 1616080992.596561
        }
    ],
    "prev_hash": null
},
{
    "items": [
        {
            "amount": 561962,
            "prev_hash": null,
            "receiver": "2b2f739e-ac64-4989-9e93-51441319436a",
            "sender": "e076bf94-c427-4b69-b8a8-e05f72b1efad",
            "t": 1616080992.596578
        },
        {
            "amount": 62726,
```

```
            {
                "amount": 562685,
                "prev_hash": "569db9a871dbb5a05f39c23a105f1749ef91286b",
                "receiver": "f0ba9efa-6d18-4070-a213-f76de5056a96",
                "sender": "602fb937-14e0-4706-90f9-348511f4fe53",
                "t": 1616080992.596876
            },
            {
                "amount": 661301,
                "prev_hash": "e86c661515d935a0234c7c229b48e20b37ae3a66",
                "receiver": "fb564195-9bdd-4367-90d3-b22da9392174",
                "sender": "c62d172c-dbc1-4fb3-b015-c90786eac802",
                "t": 1616080992.596894
            }
        ],
        "prev_hash": "f830a6e33f652408d07a2ade01a922f36f6f50e8"
    }
]
}
None
```


实现简单的分布式账本

测试本地账本

- test.py

```
...
if __name__ == '__main__’:
    ...
    chain.blocks[0].items[0].receiver = str(uuid4())
    print(json.dumps(json.loads(str(chain)), indent=4, sort_keys=True))
    print(chain.validate())
```

```
                    "receiver": "aa77a86d-0f26-45c4-83ed-64426782cad5",
                    "sender": "7a342e40-4a54-41a0-8b07-347cc3b8bcc9",
                    "t": 1616081241.470215
                }
            ],
            "prev_hash": "964b2eadebde2ad33a52c4219b79fd2b6e198559"
        }
    ]
}
```


```
Traceback (most recent call last):
  File "test.py", line 24, in <module>
    print(chain.validate())
  File "/mnt/c/Users/Valency/Downloads/foxchain-master/foxchain/blockchain.py", line 29, in
validate
    self.blocks[i].validate()
  File "/mnt/c/Users/Valency/Downloads/foxchain-master/foxchain/blockchain.py", line 15, in
validate
    assert i == 0 or self.items[i].prev_hash == self.items[i - 1].hash()
AssertionError
--------------------------------------------------------------------------------
```

实现简单的分布式账本

分布式账本进阶

第四章、第五章

P2P 网络

（2021-10-27、2021-11-03）

实现简单的分布式账本

分布式账本进阶

第八章

加密货币和智能合约

（2021-11-24）

实现简单的分布式账本

分布式账本进阶

- 实验一：分布式账本
- 2021-10-29
- 10:25-12:00
- 9A-206-1

- 请务必自带电脑
- 电脑需要预装好 Python 3.9

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

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