Understanding *Blockchain* with *Go*

Guilherme Rezende - Globo.com/Tsuru

> *"Blockchain is the greatest innovation since the internet, will disrupt every industry that exists today."*

What is Blockchain?

Blockchain is a distributed and decentralized database, it's a way of storing records of value and transactions.

Why is it called blockchain?

Block

```go
type Block struct {
    Timestamp   time.Time
    PrevBlock   []byte
    Hash        []byte
    Data        []byte
}
```

New Block

```go
func NewBlock(data string, prevBlockHash []byte) *Block {
    return &Block{
        Timestamp: time.Unix(time.Now().Unix(), 0),
        PrevBlock: prevBlockHash,
        Data:      []byte(data),
        Hash:      []byte{},
    }
}
```

Hash

```go
func (b *Block) setHash() {
    timestampStr := strconv.FormatInt(b.Timestamp.Unix(), 10)

    headers := bytes.Join([][]byte{
        b.PrevBlockHash,
        b.Data,
        []byte(timestampStr),
    }, []byte{})

    hash := sha256.Sum256(headers)
    b.Hash = hash[:]
}
```

```go
type Blockchain struct {
    blocks []*Block
}

func (bc *Blockchain) AddBlock(data string) {
    prevBlock := bc.blocks[len(bc.blocks)-1]
    newBlock := NewBlock(data, prevBlock.Hash)
    bc.blocks = append(bc.blocks, newBlock)
}
```

Genesis Block

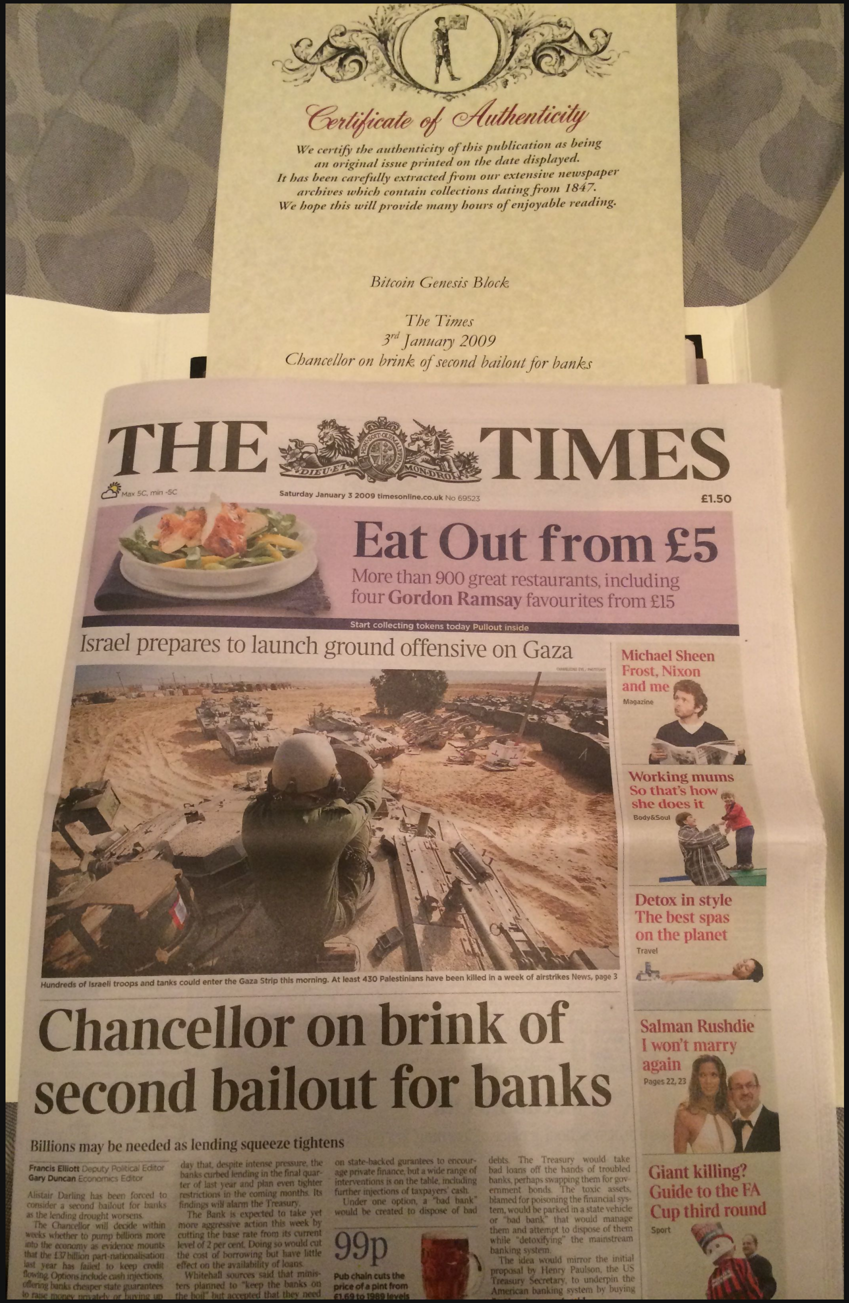
```go
type Blockchain struct {
    blocks []*Block
}

func NewBlockchain() *Blockchain {
    genesis := NewBlock("The Genesis Block", []byte{})
    return &Blockchain{[]*Block{genesis}}
}

func (bc *Blockchain) AddBlock(data string) {
    prevBlock := bc.blocks[len(bc.blocks)-1]
    newBlock := NewBlock(data, prevBlock.Hash)
    bc.blocks = append(bc.blocks, newBlock)
}
```

Bitcoin (Blockchain) Genesis Block

```
// Genesis block
const char* pszTimestamp = "The Times 03/Jan/2009 Chancellor on b
CTransaction txNew;
txNew.vin.resize(1);
txNew.vout.resize(1);
txNew.vin[0].scriptSig = CScript() << 486604799 << CBigNum(4) << 
txNew.vout[0].nValue = 50 * COIN;
CBigNum bnPubKey;
bnPubKey.SetHex("0x5F1DF16B2B704C8A578D0BBAF74D385CDE12C11EE50455
txNew.vout[0].scriptPubKey = CScript() << bnPubKey << OP_CHECKSIG
CBlock block;
block.vtx.push_back(txNew);
block.hashPrevBlock = 0;
block.hashMerkleRoot = block.BuildMerkleTree();
block.nVersion = 1;
block.nTime    = 1231006505;
```



Certificate of Authenticity

We certify the authenticity of this publication as being
an original issue printed on the date displayed.
It has been carefully extracted from our extensive newspaper
archives which contain collections dating from 1847.
We hope this will provide many hours of enjoyable reading.

Bitcoin Genesis Block

The Times
3rd January 2009
Chancellor on brink of second bailout for banks

THE TIMES

Max 5C, min -5C

Saturday January 3 2009 timesonline.co.uk No 69523

£1.50

Israel prepares to launch ground offensive on Gaza

CHAMELEONS EYE / PHOTOSHOT

Hundreds of Israeli troops and tanks could enter the Gaza Strip this morning. At least 430 Palestinians have been killed in a week of airstrikes News, page 3

Michael Sheen Frost, Nixon and me
Magazine

Working mums So that's how she does it
Body&Soul

Detox in style The best spas on the planet
Travel

Salman Rushdie I won't marry again
Pages 22, 23

Giant killing? Guide to the FA Cup third round
Sport

Chancellor on brink of second bailout for banks

Billions may be needed as lending squeeze tightens

Francis Elliott Deputy Political Editor
Gary Duncan Economics Editor

Alistair Darling has been forced to consider a second bailout for banks as the lending drought worsens.

The Chancellor will decide within weeks whether to pump billions more into the economy as evidence mounts that the £37 billion part-nationalisation last year has failed to keep credit flowing. Options include cash injections, offering banks cheaper state guarantees to raise money privately or buying up

day that, despite intense pressure, the banks curbed lending in the final quarter of last year and plan even tighter restrictions in the coming months. Its findings will alarm the Treasury.

The Bank is expected to take yet more aggressive action this week by cutting the base rate from its current level of 2 per cent. Doing so would cut the cost of borrowing but have little effect on the availability of loans.

Whitehall sources said that ministers planned to "keep the banks on the boil" but accepted that they need

on state-backed gurantees to encourage private finance, but a wide range of interventions is on the table, including further injections of taxpayers' cash.

Under one option, a "bad bank" would be created to dispose of bad debts. The Treasury would take bad loans off the hands of troubled banks, perhaps swapping them for government bonds. The toxic assets, blamed for poisoning the financial system, would be parked in a state vehicle or "bad bank" that would manage them and attempt to dispose of them while "detoxifying" the mainstream banking system.

The idea would mirror the initial proposal by Henry Paulson, the US Treasury Secretary, to underpin the American banking system by buying

99p
Pub chain cuts the price of a pint from £1.69 to 1989 levels

Mining

Proof-of-Work:
One CPU == One Vote

Proof-of-Work is implemented by incrementing a nonce in the block until a value is found that gives the block's hash the required zero bits.

The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, **forming a record that cannot be changed without redoing the proof-of-work.**

If a majority of CPU power is controlled by honest nodes, the honest chain will grow the fastest and outpace any competing chains.

To modify a past block, an attacker would have to redo the proof-of-work of the block and all blocks after it and then catch up with and surpass the work of the honest nodes.

Refactoring

```go
type Block struct {
    Timestamp    time.Time
    PrevBlock    []byte
    Hash         []byte
    Data        []byte
    Bits         uint32
    Nonce        uint32
}
```

Refactoring

```go
func (b *Block) calcHash() []byte {
    header := new(bytes.Buffer)

    header.Write(b.PrevBlock)
    binary.Write(header, binary.BigEndian, b.Data)
    binary.Write(header, binary.BigEndian, b.Timestamp.Unix())
    binary.Write(header, binary.BigEndian, b.Bits)
    binary.Write(header, binary.BigEndian, b.Nonce)

    hash := sha256.Sum256(header.Bytes())
    return hash[:]
}
```

Refactoring

```go
func (b *Block) setHash() {
    var hash []byte
    target := big.NewInt(1)
    target.Lsh(target, uint(256 - b.Bits))

    for b.Nonce < math.MaxUint32 {
        hash = b.calcHash()
        if b.validateHash(hash, target) {
            break
        }
        b.Nonce++
    }
    b.Hash = hash[:]
}
```

```go
func (b *Block) validateHash(hash []byte, target *big.Int) bool {
    var hashInt big.Int
    hashInt.SetBytes(hash[:])
    if hashInt.Cmp(target) == -1 {
        return true
    }
    return false
}
```

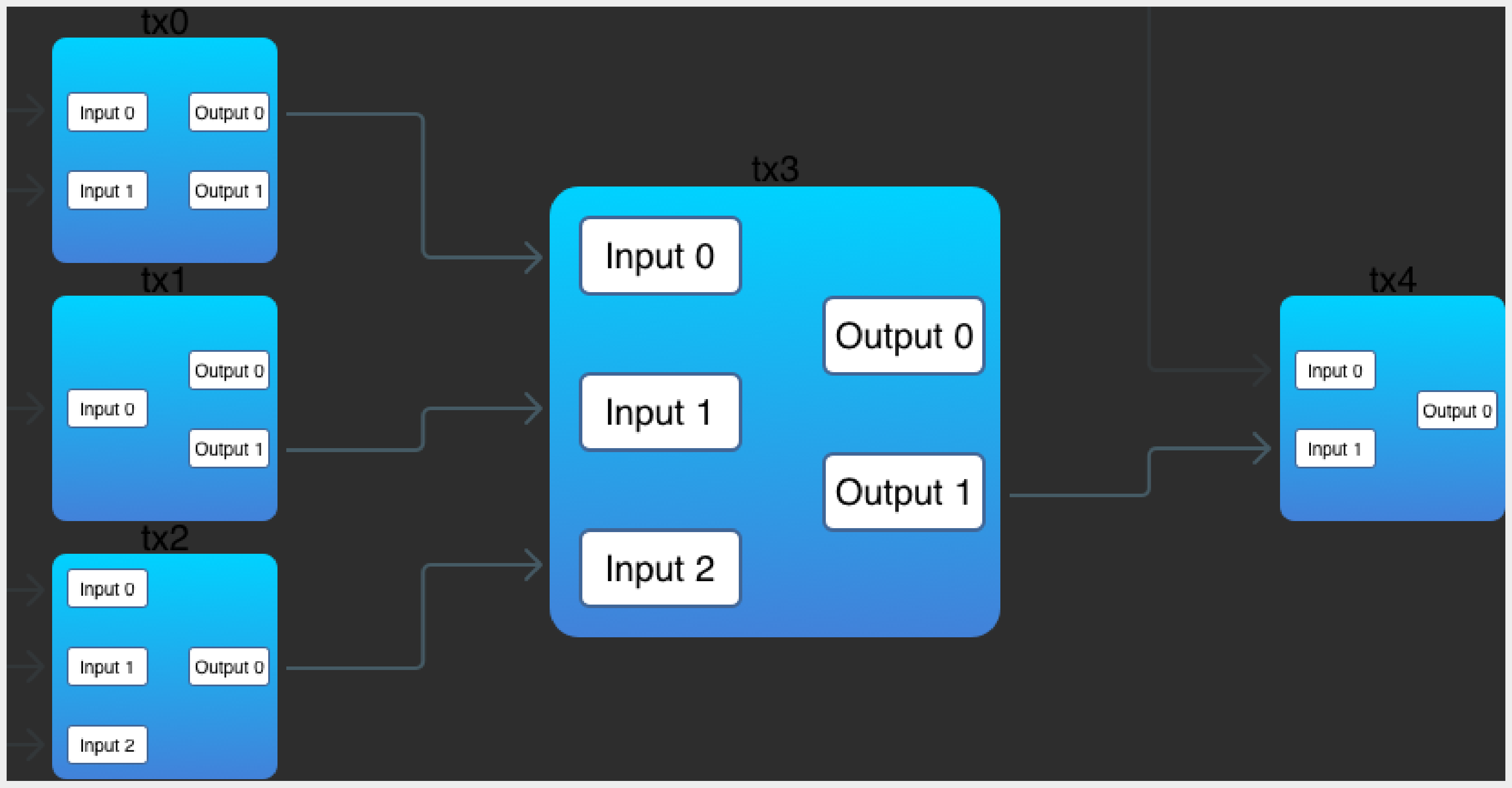
Cryptocurrencies

What is Bitcoin?

Bitcoin is a Peer-to-Peer Electronic Cash System that uses a peer-to-peer network to solve the double-spending problem.

Bitcoin is a blockchain-based system, but blockchain is not a Bitcoin-based system.

Transactions


tx0
Input 0
Input 1
Output 0
Output 1
tx1
Input 0
Output 0
Output 1
tx2
Input 0
Input 1
Input 2
Output 0
tx3
Input 0
Input 1
Input 2
Output 0
Output 1
tx4
Input 0
Input 1
Output 0

```go
type Transaction struct {
    ID      []byte
    Inputs  []TXInput
    Outputs []TXOutput
}

type TXOutput struct {
    Value        int
    PubKeyHash   []byte
}

type TXInput struct {
    Txid      []byte
    Output    int
    Signature []byte
    PubKey    []byte
```

Refactoring

```go
type Block struct {
   Timestamp    time.Time
   PrevBlock    []byte
   Hash         []byte
   Transactions []Transaction
}
```

Refactoring

```go
type Blockchain struct {
    blocks []*Block
    transactions []Transaction
}

func (bc *Blockchain) NewTransaction(tx Transaction) {
   bc.transactions = append(bc.transactions, tx)
}
```

Refactoring

```
func NewBlock(txs []Transactions, prevBlockHash []byte) *Block {
    return &Block{
        Timestamp:      time.Unix(time.Now().Unix(), 0),
        PrevBlock:      prevBlockHash,
        Transactions:   txs,
        Hash:           []byte{},
    }
}
```

Chicken or the Egg?

Reward Payment

```
func NewBlock(txs []Transaction, prevBlockHash []byte) *Block {
    txin := TXInput{[]byte{}, -1, "Reward to Satoshi"}
    txout := TXOutput{50BTC, coinbase.PubKeyHash}
    tx := Transaction{nil, []TXInput{txin}, []TXOutput{txout}}
    tx.SetID()
    txs = append(txs, tx)

    block := &Block{
        Timestamp:    time.Unix(time.Now().Unix(), 0),
        PrevBlock:    prevBlockHash,
        Transactions: txs,
        Hash:         []byte{},
        Bits:         getTargetBits(),
        Nonce:        1,
    }
    return block
```

```
// Genesis block
const char* pszTimestamp = "The Times 03/Jan/2009 Chancellor on b
CTransaction txNew;
txNew.vin.resize(1);
txNew.vout.resize(1);
txNew.vin[0].scriptSig = CScript() << 486604799 << CBigNum(4) <<
txNew.vout[0].nValue = 50 * COIN;
CBigNum bnPubKey;
bnPubKey.SetHex("0x5F1DF16B2B704C8A578D0BBAF74D385CDE12C11EE50455
txNew.vout[0].scriptPubKey = CScript() << bnPubKey << OP_CHECKSIG
CBlock block;
block.vtx.push_back(txNew);
block.hashPrevBlock = 0;
block.hashMerkleRoot = block.BuildMerkleTree();
block.nVersion = 1;
block.nTime    = 1231006505;
```

Address

```go
type Wallet struct {
    PrivateKey ecdsa.PrivateKey
    PublicKey  []byte
}

func NewWallet() *Wallet {
    curve := elliptic.P256()
    private, _ := ecdsa.GenerateKey(curve, rand.Reader)
    pubKey := append(private.PublicKey.X.Bytes(), private.PublicK
    wallet := Wallet{private, public}

    return &wallet
}
```

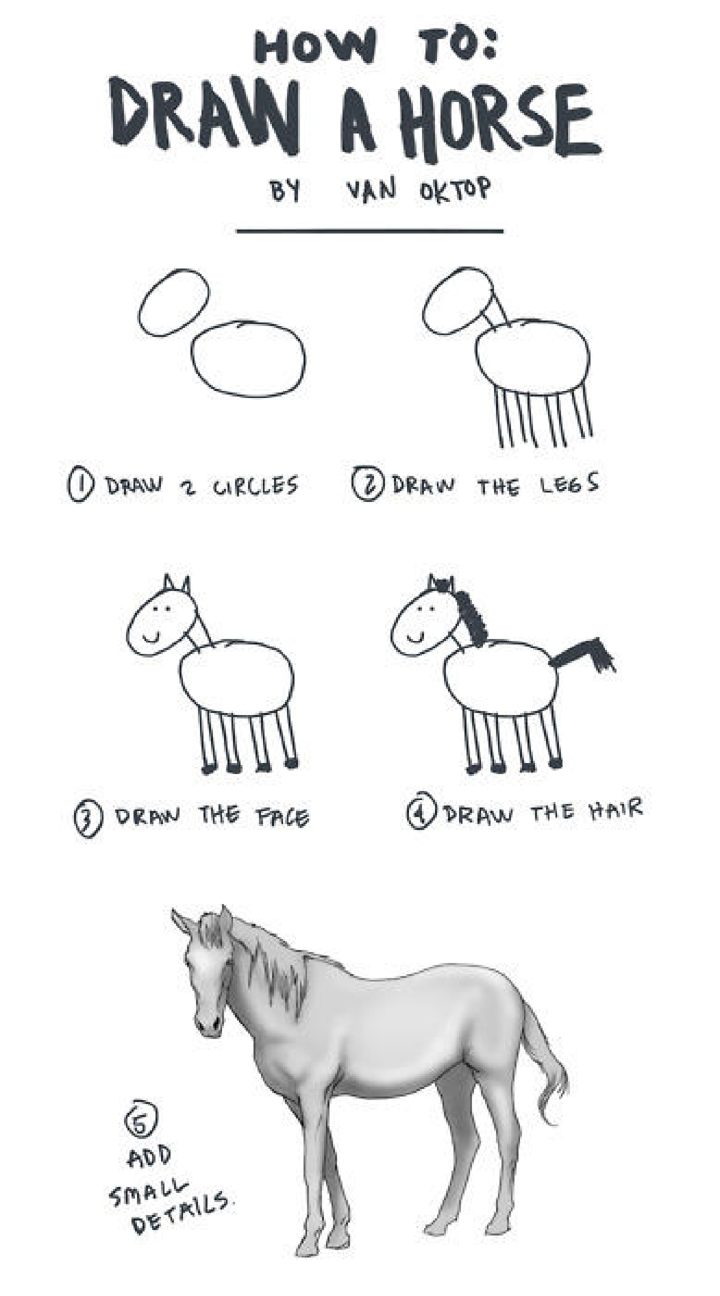
```go
func (w Wallet) GetAddress() []byte {
    pubKeyHash := sha256.Sum256(w.PublicKey)

    address := Base58Encode(pubKeyHash)

    return address
}
```

Missing points

1. smart contracts;
2. merkle tree;
3. p2p network;
4. database;


HOW TO:
DRAW A HORSE
BY VAN OKTOP
① DRAW 2 CIRCLES
② DRAW THE LEGS
③ DRAW THE FACE
④ DRAW THE HAIR
⑤ ADD SMALL DETAILS.

Go Projects

- github.com/btcsuite/btcd
- github.com/decred/dcrd
- github.com/ethereum/go-ethereum
- github.com/hyperledger/fabric

Questions?

Thank you