Blockstack-Clarity

1. 搭建Clarity环境，跑通简单的智能合约并将过程截图提交到screenshot文件夹中

docker安装

```
➜ blockstack docker pull blockstack/blockstack-core:clarity-developer-preview
clarity-developer-preview: Pulling from blockstack/blockstack-core
16ea0e8c8879: Already exists
50024b0106d5: Already exists
ff95660c6937: Already exists
9c7d0e5c0bc2: Already exists
29c4fb388fdf: Pull complete
ee0ab7fd0ac4: Pull complete
368e16c97600: Pull complete
b6c4facc9fd2: Pull complete
89624671b9f3: Pull complete
f28e4bea9e1e: Pull complete
Digest: sha256:8f4a7dab9a2d133722a568a2cf40ebbaceb6cbe82149bc49dcfe760d651f4a67
Status: Downloaded newer image for blockstack/blockstack-core:clarity-developer-preview
docker.io/blockstack/blockstack-core:clarity-developer-preview
➜ blockstack docker run -it -v $HOME/blockstack-dev-data:/data/ blockstack/blockstack-core:clarity-developer-preview bash
root@352b66942252:/src/blockstack-core# cd /src/blockstack-core
root@352b66942252:/src/blockstack-core# ls -l
total 176
-rw-r--r--  1 root root  3690 Dec 11 20:53 CONTRIBUTORS.md
-rw-r--r--  1 root root 56015 Dec 11 20:54 Cargo.lock
-rw-r--r--  1 root root  1380 Dec 11 20:53 Cargo.toml
-rw-r--r--  1 root root  2517 Dec 11 20:53 Jenkinsfile
-rw-r--r--  1 root root 35141 Dec 11 20:53 LICENSE
-rw-r--r--  1 root root    40 Dec 11 20:53 OWNERS
-rw-r--r--  1 root root    68 Dec 11 20:53 OWNERS_ALIASES
-rw-r--r--  1 root root    69 Dec 11 20:53 README.md
drwxr-xr-x  2 root root  4096 Dec 11 20:53 benches
drwxr-xr-x  2 root root  4096 Dec 11 20:53 build-scripts
drwxr-xr-x  4 root root  4096 Dec 11 20:53 charts
-rw-r--r--  1 root root  2789 Dec 11 20:53 circle.yml
drwxr-xr-x  2 root root  4096 Dec 11 20:53 deployment
-rw-r--r--  1 root root   212 Dec 11 20:53 detect
drwxr-xr-x  2 root root  4096 Dec 11 20:53 docs
drwxr-xr-x  6 root root  4096 Dec 11 20:53 integration_tests
drwxr-xr-x  2 root root  4096 Dec 11 20:53 release_notes
-rw-r--r--  1 root root     6 Dec 11 20:53 rust-toolchain
drwxr-xr-x  2 root root  4096 Dec 11 20:53 sample-programs
drwxr-xr-x  2 root root  4096 Dec 11 20:53 sip
-rw-r--r--  1 root root   721 Dec 11 20:53 skaffold.yaml
drwxr-xr-x 10 root root  4096 Dec 11 20:53 src
drwxr-xr-x  1 root root  4096 Dec 11 20:58 target
```

钱包&数据库初始化

```
root@352b66942252:/src/blockstack-core# clarity-cli generate_address
SP118V5ASAJ5S8MZVQ6M8WA82N66JNQ5Q75HBFJVH
root@352b66942252:/src/blockstack-core# export DEMO_ADDRESS=SP118V5ASAJ5S8MZVQ6M8WA82N66JNQ5Q75HBFJVH
root@352b66942252:/src/blockstack-core# clarity-cli initialize /data/db
Database created.
```

合约检查&部署

```
root@352b66942252:/src/blockstack-core# cd sample-programs/
root@352b66942252:/src/blockstack-core/sample-programs# clarity-cli check tokens.clar /data/db
Checks passed.
root@352b66942252:/src/blockstack-core/sample-programs# clarity-cli launch $DEMO_ADDRESS.tokens tokens.clar /data/db
Contract initialized!
```

2. 分析token.clar代码，将带有注释的token.clar代码提交到screenshot文件夹中

```
; 定义map, key: 账户, value: 余额
(define-map tokens ((account principal)) ((balance uint)))

; 定义函数get-balance()
(define-private (get-balance (account principal))
  ; 从map中根据账户获取余额, 默认0
  (default-to u0 (get balance (map-get? tokens (tuple (account account))))))

; 定义函数token-credit!, 账户余额增加
(define-private (token-credit! (account principal) (amount uint))
  (if (<= amount u0)
      ; 若0打印错误日志
      (err "must move positive balance")
      ; 当前余额
      (let ((current-amount (get-balance account)))
        (begin
         ; 新的余额写入map
          (map-set tokens (tuple (account account))
                      (tuple (balance (+ amount current-amount))))
          (ok amount)))))

; 定义函数token-transfer, 转账
(define-public (token-transfer (to principal) (amount uint))
  (let ((balance (get-balance tx-sender)))
    (if (or (> amount balance) (<= amount u0))
        ; 若大于账户余额或金额<0, 打印错误日志
        (err "must transfer positive balance and possess funds")
        (begin
          ; from减少指定数量
          (map-set tokens (tuple (account tx-sender))
                      (tuple (balance (- balance amount))))
          ; to增加指定数量
          (token-credit! to amount)))))

; 定义函数mint!, 铸币
(define-public (mint! (amount uint))
   ; from获得指定数量的代币
   (let ((balance (get-balance tx-sender)))
     (token-credit! tx-sender amount)))
```

思考题目

- 题目一：根据今天对于智能合约的讲解，你认为智能合约可以解决哪些现有互联网无法解决的问题？又会带来哪些问题?

  智能合约提供了信任基础设施，可以无需中间商完成用户需求。但目前尚未有只能由智能合约解决，可中心化互联网无法解决的问题，而且后者效率更高、成本更低、用户体验更好。

  应用智能合约，可以部分解决中心化互联网存在的跑路问题，典型场景如会员的押金或预存卡被挪用甚至跑路，导致无法兑付提现，实际是做到了100%透明的准备金（监管账户）。

  但是又会引入一些问题，如TPS、块确认的时间延迟（ETH一般12个块、EOS大概100多秒）、token价格波动、出入金、用户体验等。