sCrypt



https://scrypt.io/

https://docs.scrypt.io/

> sCrypt is an embedded Domain Specific Language (eDSL) based on TypeScript for writing smart contracts on Bitcoin.

Priceで有料プランがあるのはプラットフォームを提供しているからか？

チュートリアルで動かした範囲では以下のことができた。

- sCryptで TypeScript のフォーマットに合わせたコードを書く
- 書いたコードから bitcoinjs-lib を使ったアプリに置き換えてくれる(ライブラリにもできるようだ)
- テンプレートの deploy を実行すると、スクリプトへの送金とそのスクリプトを解いて払い戻すところまでやってくれる
  - BitcoinSV の testnet や Bitcoin の signet が使える(デフォルトは BitcoinSV)
- 自動でBitcoinスクリプトに置き換えてくれるが、思いもよらないスクリプトになっていた。わざとそうしているのか？

チュートリアル

Tutorial 1: Hello World

console

```
$ npx @scrypt-inc/cli-btc help
Usage: scrypt-cli-btc [options] [command]

Options:
  -h, --help                display help for command

Commands:
  version                   Output the version number
  system                    Show system info
  project [options] <name>  Create a new smart contract project
```

```
  compile [options]         Compile smart contracts in current project
  deploy [options]          Deploy a smart contract
  help [command]            display help for command
$ npx @scrypt-inc/cli-btc version
v1.0.0
$ npx @scrypt-inc/cli-btc project helloworld
✓ Initialize Git repo
✓ Set up project
✓ Set project name
✓ Configure package.json

Project helloworld was successfully created!

Add your Git repo URL and you're good to go:
cd helloworld && git remote add origin <your-repo-url>
```

コントラクトのコードについては修正不要。

src/contracts/helloworld.ts

```
export class Helloworld extends SmartContract {
    @prop()
    hash: Sha256

    constructor(hash: Sha256) {
        super(...arguments)
        this.hash = hash
    }

    @method()
    public unlock(message: ByteString) {
        assert(sha256(message) == this.hash, 'Hash does not match')
    }
}
```

スクリプトは `OP_SHA256 <hash> OP_EQUALVERIFY <pubkey> OP_CHECKSIG` で witness が `<sig> <preimage>` というところか。

署名はこのコードには書いていないが、省略されているのか自動で追加されるのかはわからん。

`npx @scrypt-inc/cli-btc compile` すると `artifacts/contracts/helloworld.json` などが生

成される。見ても分からん。
この JSONファイルは TypeScript で `import` することができ、これは既に `src/index.ts` に書かれてある。

(`watch` はスルー)

`npm run genprivkey` で秘密鍵などを作るのだが、これは BitcoinSV の testnet 向けになっている。
faucetサイトの紹介はあるのだが、作られたアドレスは P2TR に対して faucet サイトは P2PKH しか認めてなさそうなのだ。
`@scrypt-inc` でサポートしているネットワーク `SupportedNetwork` は以下の3種類だけのようだ。

```
'btc-signet'
'fractal-mainnet'
'fractal-testnet'
```

テンプレートで作っただけのコードということもあるだろうが、アドレスの生成にはネットワークを指定していないなどあまりできがよろしくない。。。
`tests/utils/privateKey.ts` を適当に改造。
`.env` を削除して `npm run genprivkey` で `'btc-signet'` のアドレスを生成。コンソールにはscrypto.io のURLが載っているが、あれも固定文字列だ。秘密鍵はWIF形式で `.env` に出力されている。そのファイルがあれば再生成はしない。
signet faucetで送金してもらう。

console

```
$ npm run genprivkey

> helloworld@0.1.0 genprivkey
> npx tsx tests/utils/privateKey.ts

Private key generated and saved in ".env"
Publickey: 03eff41f451dd9462b558de75122b7df53d8b7f456d5f07d5f83f59d492cd27def
You can fund its address
'tb1pnx2h7aavv4sc9vf2jdepelatn53a602errmuel7yq847elz75qaqpyr9kt' from the sCrypt
faucet https://scrypt.io/faucet
ueno@hozan:~/Bitcoin/js/scrypt/helloworld$ cat .env
PRIVATE_KEY="cPzmnBGkJPvRFDwjCXvuPhDgjqigTk6H4koHD7uutwa7rcK3hnpp"
```

入金されたらこのスクリプトに支払う。

既に `deploy.ts` があるので `npx tsx deploy.ts` を実行するもエラーになる。

出力ログは `npx ts-node deploy.ts` になっているが、これもエラーになる。

- `ts-node` よりも `tsx` の方がよさそう？ stackoverflow
- `import { Helloworld }` のパスが足りてないだけのようで `'./src/contracts/helloworld'` にした

少し進むがまたエラー。

- `Artifact not loaded for contract Helloworld!` とあるが、これは `src/index.ts` のことか？
- 上で修正した `import { Helloworld }` のパスを `./src/` にして `src/index.ts` が export している方を読み込む

また進むがエラー。

- `Invalid network version` は、これは BitcoinSV から signet にしたためか。
- `tests/utils/txHelper.ts` がネットワーク未指定の場合 `'fractal-testnet'` を使うようになっている
- これらを `'btc-signet'` に変更し、 `.env` にも `NETWORK='btc-signet'` を追加

まだエラー。

- `bc1.... has an invalid prefix` と、最初は signet のアドレスなのだがブロックチェーンを参照する `mempoolProvider.ts` に渡すアドレスが mainnet になっている。
- `./tests/utils/txHelper.ts` の `getDefaultSigner()` で最後にネットワークを設定していなかった。

まだエラー。

- invalid prefixだが、アドレスはsignetになっている。
- `deploy.ts` の `Covenant.createCovenant()` でネットワークを設定していないため。

通った！

console

```
$ npx tsx ./deploy.ts
Private key already present ...
You can fund its address
'tb1pnx2h7aavv4sc9vf2jdepelatn53a602errmuel7yq847elz75qaqpyr9kt' from the sCrypt
faucet https://scrypt.io/faucet
Private key already present ...
You can fund its address
'tb1pnx2h7aavv4sc9vf2jdepelatn53a602errmuel7yq847elz75qaqpyr9kt' from the sCrypt
faucet https://scrypt.io/faucet
Helloworld contract deployed:
369053426075607fb8296dc7e498052b293da1a984fe23ad22d92ae8103f757b
Helloworld contract called:
```

`05cb2bd4c5bf0c9533e3ee1b6d4d5be82d20c46e6784bb4777d58b06efa322e2`

最初に取得したアドレスへの送金が2つあったせいか、2つ同時に処理された。

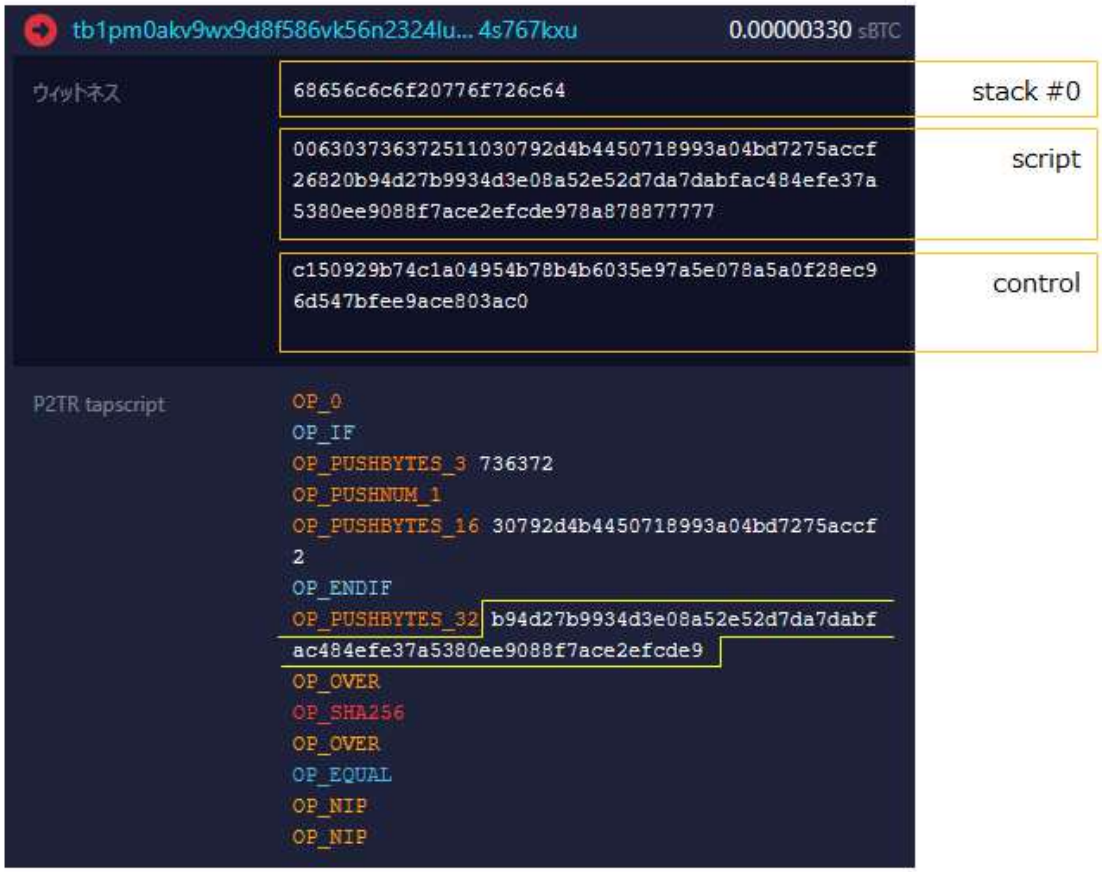
contract deployed

contract called

`hello world` をSHA256する

と `b94d27b9934d3e08a52e52d7da7dabfac484efe37a5380ee9088f7ace2efcde9` 。

contract calledのVIN#0のスクリプトに入っている。



が、予想していたスクリプトと結構違う。

`OP_OVER` は2番目のスタックを一番上にコピー、 `OP_NIP` は2番目のスタックを削除ということで、OKになる。

				b94d...			
		6865...	b94d...	b94d...	01		
	b94d...	b94d...	b94d...	b94d...	b94d...	01	
6865...	6865...	6865...	6865...	6865...	6865...	6865...	01
		OP_OVER	OP_SHA256	OP_OVER	OP_EQUAL	OP_NIP	OP_NIP