Blockchain

Sylvain Conchon

Laboratoire Méthodes Formelles

Université Paris-Saclay, CNRS, ENS Paris-Saclay

sylvain.conchon@universite-paris-saclay.fr

SMART CONTRACTS

Contrats intelligents

Un contrat intelligent, smart contract, est simplement un (bout de) programme stocké sur une blockchain.

D'une manière générale, un smart contract prend la forme d'un objet avec plusieurs méthodes et un état interne (ou un module avec des fonctions et un état).

Les méthodes (ou fonctions) d'un smart contract sont appelées points d'entrées.

Une fois déployé sur la blockchain, on dit originé, un smart contract possède une adresse. Sous Tezos, ces adresses commencent avec le préfixe `KT1`.

Comme n'importe quel compte, un smart contract possède également un solde (balance).

Exemple de smart-contract Michelson

Voici un exemple de smart contract en Michelson. Il s'agit d'un petit langage fonctionnel à pile (voir cours suivant).

```
parameter int;
storage int;
code
 {
  UNPAIR;
  SWAP;
  DROP;
  NIL operation;
  PAIR;
 }
```

On enregistre ce code dans un fichier en utilisant l'extension `.tz`, par exemple `ex1.tz`.

Déployer un smart-contract (1/1)

Pour originer ce contrat, il suffit d'utiliser `octez-client` de la manière suivante :

```
➜  Tezos git:(master) ✗ octez-client originate contract ex1 transferring 0 from sylvain
 running ex1.tz --init '0' --burn-cap 0.1
```

Cette ligne de commande indique :

- Le compte qui origine le contrat : `sylvain`
- Le fichier qui contient le contrat en Michelson : `ex1.tz`
- Le nom du contrat : `ex1`
- Le contenu initial du storage (réduit ici à un entier) : `0`
- Une quantité maximale de tez que l'on accepte de brûler pour payer pour l'espace de stockage utilisé : `0.1` tez

Déployer un smart-contract (2/2)

```
Node is bootstrapped.
Estimated gas: 1411.007 units (will add 100 for safety)
Estimated storage: 299 bytes added (will add 20 for safety)
Enter password for encrypted key:
Operation successfully injected in the node.
Operation hash is 'ooJqiatf7JQj7LRphmBhwnGz7ZjZ3TpQgbF3oJ2HV8Viodjrojg'
Waiting for the operation to be included...
Operation found in block: BL1Cp9wuJVLbSGuZLpyjaPSRPkp6PW8FLxje6s27s6FnWtgmMfv (pass: 3, offset: 0)
This sequence of operations was run:
  Manager signed operations:
    From: tz1b6L85KsqbFSYidjkz1QJGAREkTQoKCcZs
    Fee to the baker: ꜩ0.000424
    Expected counter: 14536097
    Gas limit: 1512
    Storage limit: 319 bytes
    Balance updates:
      tz1b6L85KsqbFSYidjkz1QJGAREkTQoKCcZs ... -ꜩ0.000424
      payload fees(the block proposer) ..... +ꜩ0.000424
    Origination:
      From: tz1b6L85KsqbFSYidjkz1QJGAREkTQoKCcZs
      Credit: ꜩ0
      Script:
        { parameter int ;
          storage int ;
          code { UNPAIR ; SWAP ; DROP ; NIL operation ; PAIR } }
        Initial storage: 0
        No delegate for this contract
        This origination was successfully applied
        Originated contracts:
          KT1Pfo1BRqpP5KG61my6rmEMWLYBj5hDJtRv
        Storage size: 42 bytes
        Paid storage size diff: 42 bytes
        Consumed gas: 1411.007
        Balance updates:
          tz1b6L85KsqbFSYidjkz1QJGAREkTQoKCcZs ... -ꜩ0.0105
          storage fees ..... +ꜩ0.0105
          tz1b6L85KsqbFSYidjkz1QJGAREkTQoKCcZs ... -ꜩ0.06425
          storage fees ..... +ꜩ0.06425

New contract KT1Pfo1BRqpP5KG61my6rmEMWLYBj5hDJtRv originated.
```

Appeler un contrat (1/2)

L'appel à un contrat `KT1` est identique à un transfert vers un compte `TZ1`, à ceci près que l'on doit spécifier l'argument à envoyer au (point d'entrée du) contrat.

```
➜  Tezos git:(master) ✗ octez-client transfer 0.2 from sylvain to ex1 --arg '42'
```

Comme nous le verrons plus loin, le paramètre permet de définir

- le point d'entrée du contrat
- la valeur envoyée à ce point d'entrée.

Appeler un contrat (2/2)

```
Node is bootstrapped.
Estimated gas: 2110.640 units (will add 100 for safety)
Estimated storage: no bytes added
Enter password for encrypted key:
Operation successfully injected in the node.
Operation hash is 'opPmmP3QLPEuthau4RXv6dAMvTBbML3iiE4QGbSsCabYfyqaJry'
Waiting for the operation to be included...
Operation found in block: BLQuZTA4ijDbT1CcXbZYbKEg2caKiyjEQTZ3y9FQohV9KsXFWm7 (pass: 3, offset: 0)
This sequence of operations was run:
  Manager signed operations:
    From: tz1b6L85KsqbFSYidjkz1QJGAREkTQoKCcZs
    Fee to the baker: ꜩ0.000482
    Expected counter: 14536099
    Gas limit: 2211
    Storage limit: 0 bytes
    Balance updates:
      tz1b6L85KsqbFSYidjkz1QJGAREkTQoKCcZs ... -ꜩ0.000482
      payload fees(the block proposer) ....... +ꜩ0.000482
    Transaction:
      Amount: ꜩ0.2
      From: tz1b6L85KsqbFSYidjkz1QJGAREkTQoKCcZs
      To: KT1Pfo1BRqpP5KG61my6rmEMWLYBj5hDJtRv
      Parameter: 42
      This transaction was successfully applied
      Updated storage: 42
      Storage size: 42 bytes
      Consumed gas: 2110.640
      Balance updates:
        tz1b6L85KsqbFSYidjkz1QJGAREkTQoKCcZs ... -ꜩ0.2
        KT1Pfo1BRqpP5KG61my6rmEMWLYBj5hDJtRv ... +ꜩ0.2
```

Chaînes de caractères en paramètre

Lorsque vous transmettez le paramètre du contrat à `octez-client`, il faut faire attention à l'utilisation des apostrophes et des guillemets.

Une bonne pratique est de toujours mettre la valeur du paramètre entre apostrophes.

Pour passer une chaîne de caractères dans la valeur envoyée en paramètre, il faut mettre cette chaîne entre guillemets, le tout à l'intérieur des apostrophes (`'"Hello"'`).

Attention : sur Tezos, les chaînes de caractères ne peuvent pas contenir d'accents.

Paiement du coût de stockage

Lorsque vous déployez un contrat, ou lorsque vous appelez un contrat d'une manière qui augmente la taille du stockage, vous devez payer pour cet espace.

Il n'y a pas de bénéficiaire particulier pour ces tokens (l'espace supplémentaire utilisé impacte l'ensemble de la communauté), donc ces tokens sont brûlés (détruits).

On utilise l'option `--burn-cap` pour indiquer la quantité maximale que l'on accepte de brûler lors de cette transaction. Il s'agit d'une limite maximale, mais seule la quantité réellement nécessaire sera dépensée.

Spécificités des *smart contracts*

- Programmes immuables
  - Pas de correction possible
  - *Code is law*

Spécificités des *smart contracts*

- Programmes immuables
  - Pas de correction possible
  - *Code is law*
- Programmes et données publiques
  - Pas de confidentialité
  - Tout est visible (code + données)
  - Historique des appels dans la blockchain