3 Creating Tokens

In CyberWay everybody can create his/her own type of tokens (In EOS this right applies to validators only). The tokens of the application should be deployed on a separate account from `cyber.token` . When creating tokens a developer can use a reference implementation of the `cyber.token` contract as a base by loading its contents into developer's workspace.

3.1 Download contract source files

Enter the directory created for the contracts and load a copy of the remote repository with the contract source files into it.

```
cd CONTRACTS_DIR
git clone https://github.com/cyberway/cyberway.contracts --branch <branch na
```

The cyberway.contracts repository contains several contracts, but the `cyber.token` contract is required to create tokens.

```
cd cyberway.contracts/cyber.token
```

3.2 Create an account for the contract

Before you deploy a token-contract, you need to create an account of this contract by executing the following command line:

```
cleos create account olga olga.token <public key>
```

Command line parameters:
olga — an account name for the contract being created;
olga.token — a name of the token contract downloaded from the `cyberway.contracts` repository with the source files.

As an example a contract account has been created with the `olga.token` name.

3.3 Compile a contract

```
eosio-cpp -I include -o olga.token.wasm src/olga.token.cpp --abigen
```

The contract is compiled into a web assembler file wasm format. The presence of the `--abigen` option indicates that the `abi/olga.token.abi` file will also be generated.

3.4 Install Token contract

```
cleos set contract olga.token CONTRACTS_DIR/cyberway.contracts/olga.token --a
```

Parameter:
`olga.token@active` — a name that will be used to authorize the request.

The token contract will be considered successfully installed if the resulting output of the command being executed contains information of the form:

```
executed transaction:  ...
#         eosio <= eosio::setcode              {"account":"olga.token","vm
#         eosio <= eosio::setabi               {"account":"olga.token","ab
warning: transaction executed locally, but may not be confirmed by the netw
```

3.5 Create a Token

To create a new token use the action `create`. The argument is the type of the token `symbol_name`, which contains two values — the maximum value of the sentence and the symbol of the token. The call to this action has the form:

```
cleos push action olga.token create '{"issuer":"olga", "maximum_supply":"1000
```

The `-p olga.token@active` option authorizes the `olga.token` contract to perform this action.

Creating a token is considered successful if the following information appears on the monitor:

```
executed transaction: <info>
#   olga.token <= olga.token::create          {"issuer":"olga","maximum_sup
```

As a result, a new token SYS will be created, which has an accuracy of four decimal digits. The maximum allowable number of tokens in circulation should be limited to the value of 1,000,000,000. To create this token, a permission of the contract `olga.token` is required. The name `olga.token@active` will be used when authorizing the request.

3.6 Releasing a Token

The author of a token can issue tokens to an already existing account, for example, to an account with the name «alice» by executing:

```
cleos push action olga.token issue '[ "alice", "100.0000 SYS", "memo" ]' -p o
```

The following information should appear as a result of the command execution:

```
executed transaction:  ...
#   olga.token <= olga.token::issue           {"to":"user","quantity":"100.(
>> issue
#   olga.token <= olga.token::transfer        {"from":"olga","to":"alice","(
>> transfer
#         olga<= olga.token::transfer        {"from":"olga","to":"alice","qu
#          user <= olga.token::transfer        {"from":"olga","to":"alice",'
```

The output contains one `issue` action and three `transfer` actions. At `issue` run time, three internal calls are additionally generated notifying the sender and receiver of tokens.

3.7 Token transfer

Some tokens can be transferred from the balance of one account to the balance of another account. For example, to transfer an amount of 25 tokens from the «alice» account balance to «bob» account balance you need to use the following command line:

```
cleos push action olga.token transfer '[ "alice", "bob", "25.0000 SYS", "m" ]
```

To perform this action you need permission from the «alice» sender account — the `-p alice@active` option is available.

The transfer of tokens is considered as successfully completed if upon its completion the following information appears in the command window:

```
executed transaction:  ...
#   olga.token <= olga.token::transfer        {"from":"alice","to":"bob","qu
>> transfer
#          user <= olga.token::transfer        {"from":"alice","to":"bob","(
#        tester <= olga.token::transfer        {"from":"alice","to":"bob","(
```

To control the token transfer, you can use the `get_currency_balance` call to get the balance data of the sender and the receiver accounts by executing:

```
cleos get currency balance olga.token alice SYS
cleos get currency balance olga.token bob SYS
```