Octez client: calling a contract

This document will guide you through every step needed to make simple calls to contracts.

Before you go through with this tutorial, make sure you setup your gitpod environment and configure your octez client, following the steps listed here: https://docs.google.com/document/d/1IE-kvRfGcJi4ECiGymNZvJoMvcPxquf4FnQcFWVFLDA/edit?usp=sharing

1. Send Tez to another address

You can transfer tokens from your account, to any account for which you created an alias.

For example, if your account is named alice, and you created an alias for your friend bob, input the following command to transfer 5 tez to bob:

```
octez-client --wait none transfer 5 from alice to bob
```

You could also directly input the destination address instead of the alias.

The transfer is not immediate! You need to wait for a baker to include your transaction in a block, then for a few more blocks after that to make sure that block is final.

Exchange your addresses with others and do a few transactions. (Just be sure you're on a testnet!).

2. Check your transactions on a block explorer

Remember that you can access the status and history of any account, using one of the block explorers available for Tezos.

Go to https://tzkt.io/, then make sure you select the correct network, then search for the address of the account you want to check.

Look for the recent transactions, and check that your balance has been updated.

Remember that it may take a couple of minutes for the block explorer to show the changes.

3. Create an alias for a smart contract

In the exercise, you have been provided the address of a smart contract, that you will need to call. Smart contract addresses start with KT.

Start by creating an alias for this contract, using the same command as for a regular account.

For example, if you want to use the alias contract1, input this command:

```
octez-client remember contract contract1 [address of the contract]
```

4. Check the balance of the contract

Just like a regular account, a contract has a balance, that you can check with the following command (using the alias you selected earlier):

```
octez-client get balance for contract1
```

5. Call a simple contract with only an int as parameter

The first contract address you were given, is a very simple contract, that stores a single value in its storage, a number, of type int.

All you can do with it is to call it, and send a new value, that it will store in its storage.

Pick a new value. For example here, we pick 432, but pick your own, so that it's different from other participants.

Then Input this command to call the contract, so that it will replace its storage with your value:

```
octez-client --wait none transfer 5 from alice to contract1 --arg '432'
```

Remember to use the correct aliases instead of alice and contract1.

Note that the symbol ' has variants like ‘ or ’ that look very similar but will not be accepted.

Note that here, you also transferred 5 tez to the contract. You didn't need to and could have saved 5 tez, by putting 0 instead.

6. Check the output and see that it was successful

The output is quite long, and contains lots of information. In particular, look for the new value of the contract's storage and this message.

```
This transaction was successfully applied
      Updated storage: 10
```

In some cases, you may get an error, and not see this, for example if there was some connection issue with the node.

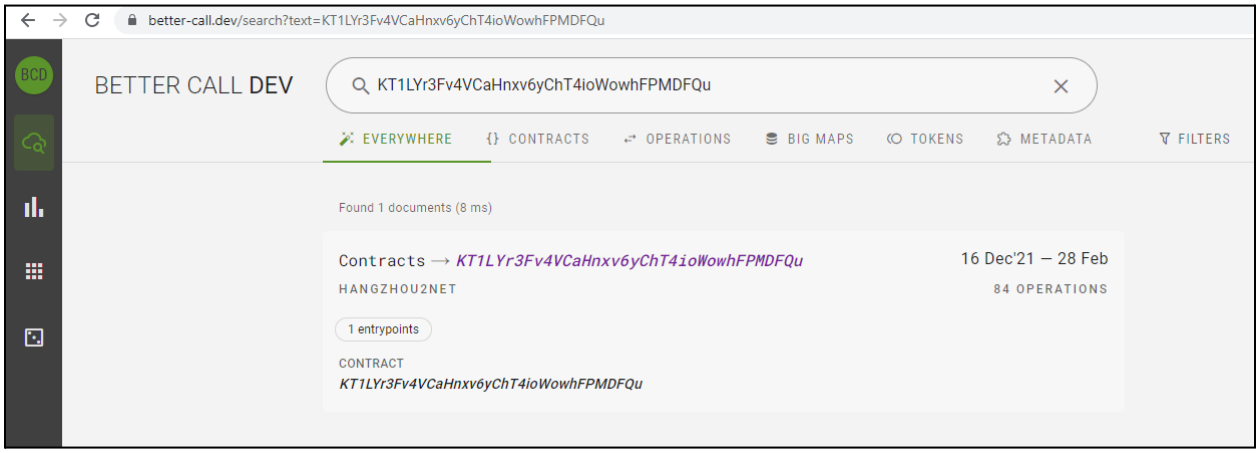
This doesn't mean the transaction didn't happen. If that's the case, check the history of transactions of your account on an explorer. Remember to wait a couple of minutes if you don't see it immediately.

7. Check the result on a block explorer

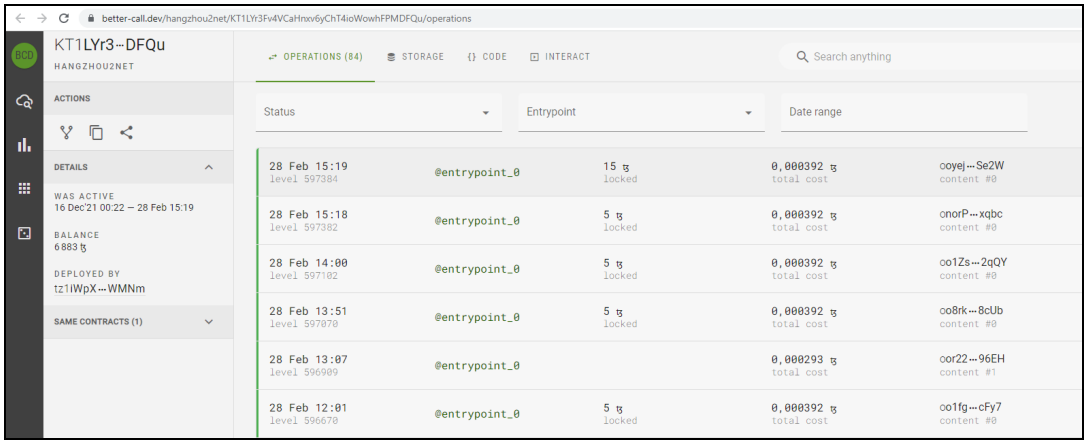
To check the activity on a contract, we will use a different block explorer, better call dev.

Go to https://better-call.dev/

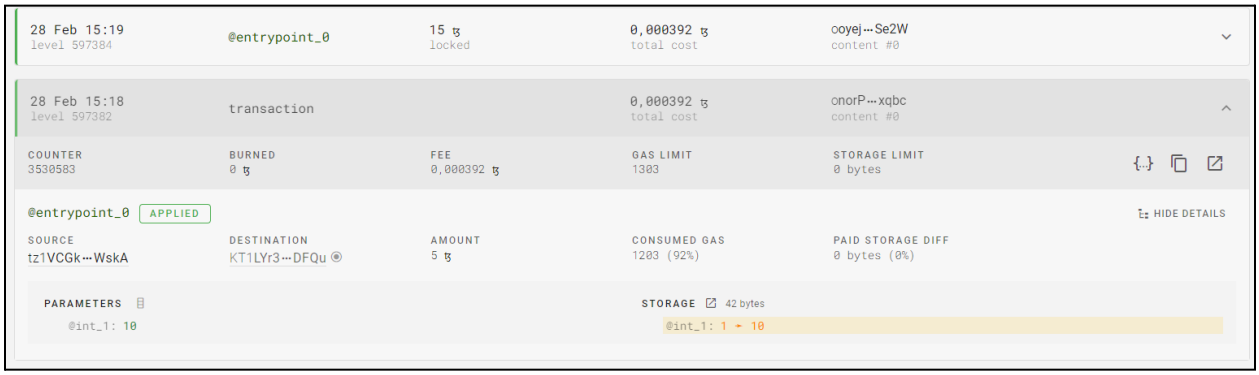
In the search area, input the address of the contract you want to check.



When you see the contract in the results, click on it to open the page. You will then see the list of recent transactions for this contract :



Click on a transaction to see the details:



You should be able to find your transactions, as well as other people's transactions on that contract.

8. Call a contract with a string as a parameter, that increases the storage

In the next exercise, you are asked to call a contract that can be called with a string as a parameter. The contract stores a string, and after every call, it will add a comma to the storage, then the value of your parameter.

A string is a sequence of characters, that you can delimit between two double-quotes: "

For example the string `Hello World!` is expressed as `"Hello World!"`

When you input a command, remember that the value of the parameter should be put between two single quotes, for example `'432'` for a number. For a string, this means you will have single quotes outside, and double quotes inside, like this:

```
'"Hello World!"'
```

Note that just like ', the symbol " has variants like ‶ or ″ that look similar but will not be accepted.

When you call that contract, the amount of data it stores will increase. You will need to pay for that extra storage, by burning some tez. To indicate how many tez you are accepting to burn at the most, add this at the end of your command. For example, if you accept to burn 0.1 tez:

```
--burn-cap 0.1
```

Your final command, will look like this (this should be a single line)

```
octez-client --wait none transfer 0 from alice to contract2 --arg
'"Hello"' --burn-cap 0.1
```

9. Check that the result is correct

Make sure the output is correct and you see the expected value of the new storage.

Also make sure you explore this contract in a block explorer, and see your transaction there, and the new value of the contract's storage