# Santander Brazil Hackathon

Ethereum private blockchain

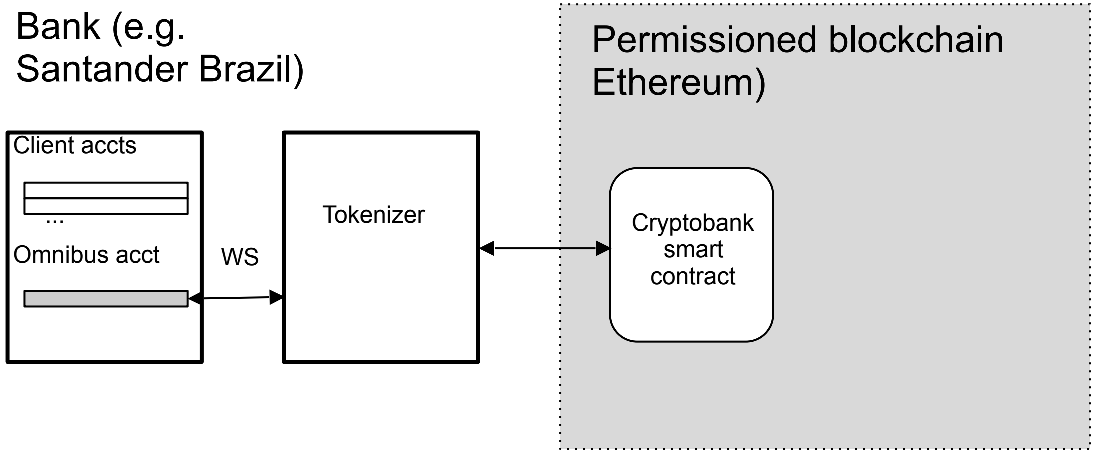
Sao Paulo, December 10-11 2016

Santander

## Context and objectives

- ✓ Objective: to create and demonstrate applications of tokenized representations of money in a private (Ethereum) Blockchain
- ✓ Both types of applications interesting:
  - New banking applications (i.e. to be offered as a product by the bank, using the participant's technology)
  - New business models (i.e. the participant catering directly to final customers using the bank's platforms)
- ✓ The intent is using a private Blockchain, ultimately to be deployed within the bank's firewall; additional, external participants in this Blockchain possible (e.g. an external bank), on a reasonably selective basis
- ✓ Ok to establish connections to external parties outside the firewall (including public blockchains)

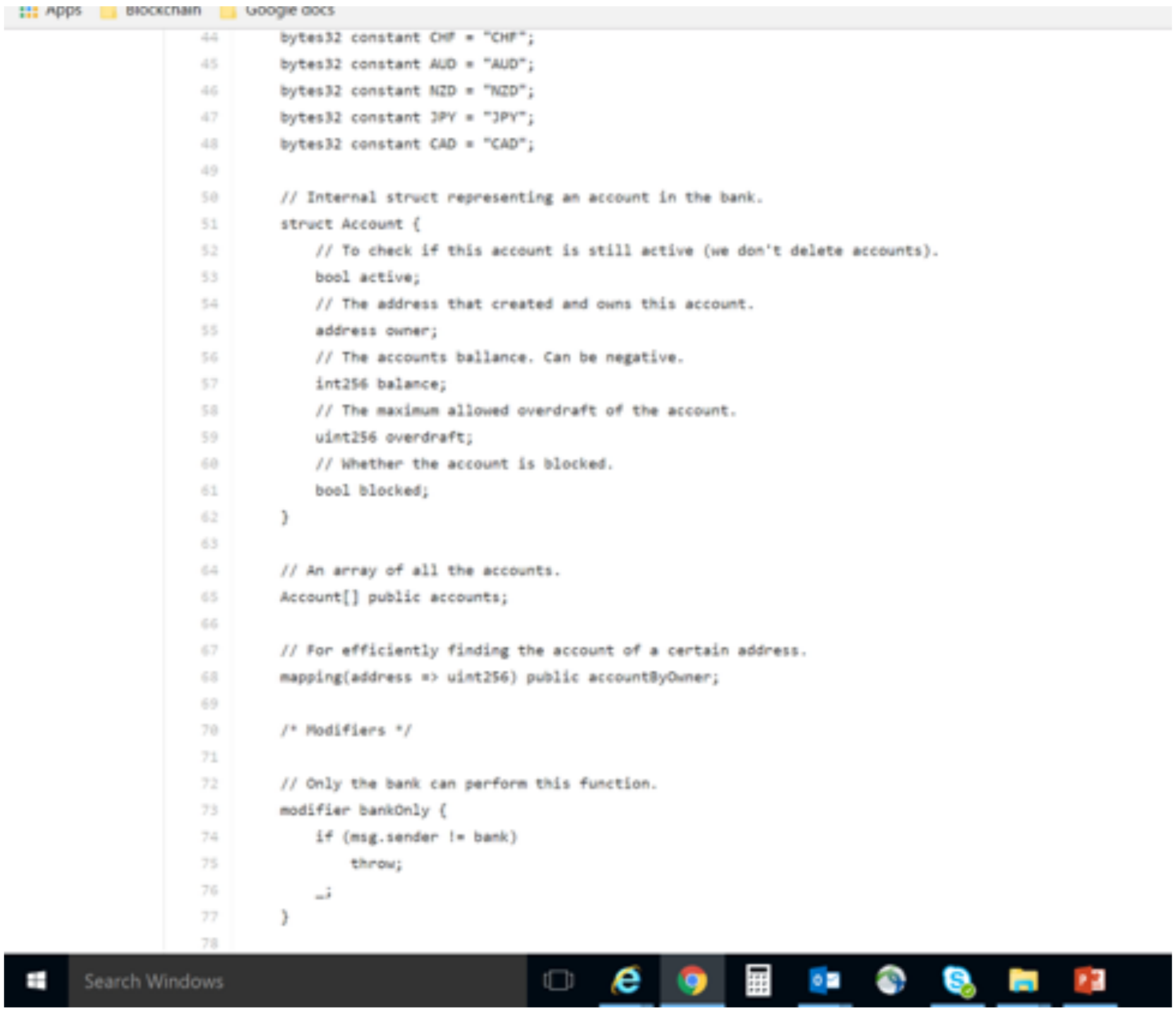
# Tokenization platform



**Multiple banks connected to the network, both domestically and internationally** (different currencies)

- ✓ Client sends money from her bank account to a bank-owned omnibus account, adding her Ethereum address in the "message" field
- ✓ The tokenizer detects the incoming transfer and reflects the balance in the cryptobank smart contract, associated to the given Ethereum address
- ✓ Users and applications use these balances as digital money: making transfers, doing payments, etc.
- ✓ Clients can ask the smart contract to redeem funds, indicating the bank account to receive the funds
- ✓ The tokenizer detects the redemption request and sends the funds back from the omnibus account to the client

## The “cryptobank” smart contract



```
    bytes32 constant CHF = "CHF";
    bytes32 constant AUD = "AUD";
    bytes32 constant NZD = "NZD";
    bytes32 constant JPY = "JPY";
    bytes32 constant CAD = "CAD";

    // Internal struct representing an account in the bank.
    struct Account {
        // To check if this account is still active (we don't delete accounts).
        bool active;
        // The address that created and owns this account.
        address owner;
        // The accounts ballance. Can be negative.
        int256 balance;
        // The maximum allowed overdraft of the account.
        uint256 overdraft;
        // Whether the account is blocked.
        bool blocked;
    }

    // An array of all the accounts.
    Account[] public accounts;

    // For efficiently finding the account of a certain address.
    mapping(address => uint256) public accountByOwner;

    /* Modifiers */

    // Only the bank can perform this function.
    modifier bankOnly {
        if (msg.sender != bank)
            throw;
        _;
    }

```

- ✓ Reasonably standard cryptocurrency contract implementation
- ✓ Methods for:
  - Creating wallets
  - Adding funds
  - Making transfers
  - Redeeming funds
- ✓ ERC20 compatibility layer added

## What will be available for the hackathon

- A private Ethereum network with some miners running on the Hackathon’s Bluemix space. The genesis block json file will be available so participants can set up their own nodes
- Few (~5?) demo banks with the tokenizers connected and the cryptobank smart contracts deployed, some of which will be in different currencies
- Accounts with (fake) money for all participant teams in as many banks as they want. So participants will be able to send money to the cryptobank smart contracts and redeem it back to the bank. Limits will be removed (e.g. to use an account as a merchant account to receive payments)
- The source code of the smart contract
- SanCash wallet applications in all banks (mainly for testing)
- Space in the Bluemix cloud so participants can interact with all this

## Suggested applications

- ✓ Peer to peer and merchant payments
- ✓ Unbanked propositions
- ✓ Corporate payments
- ✓ Micropayments and machine-to-machine payments
- ✓ International payments and remittances
- ✓ Trading applications and settlement
- ✓ … you tell us ;-)

## Important!

- ✓ No hacking demobank and/or SanCash wallet. These are simple, non-industrial grade applications for development purposes only
- ✓ Ok to try to find hacks to the Ethereum protocol and/or the smart contract – security enhancement tips will be rewarded!
- ✓ But no malicious attacks (e.g. DoS, etc.), since this will be deployed in a private, trusted setting, with cost of gas set to zero

Santander