A living programming environment for a living blockchain
Disclaimer!

This is not a blockchain mechanisms talk!
( Sorry disappoint you :) )
General technology explanation
Starting by the fruit: Smart contracts

- Digital reification of contracts
  - Emulate the logic of contractual clauses
  - Self-executing
  - Self-enforcing
- Reduce transactional costs
- Minimise exceptions
Following by the branch: Ethereum

- Blockchain based technology
- Open source & public network
- Smart contracts
  - State stored in a blockchain
  - Byte-code executed in the turing complete EVM
  - Many development languages (solidity, serpent, etc)
Arriving to the trunk: Blockchain

- Open and distributed ledger
- Records a constantly-growing list of transactions in between two parties. (blocks)
- Resistant to modification by design
- Cryptocurrency: Paying to reinforce the social engagement with the security
First-citizens in Blockchain

- Block: stamped batch of transactions
- Transaction: Representation of mutations of state
  - Movements of money
  - Method activation
- Account: Source and target of transactions (account in the accountancy meaning)
- Contracts (Specific in ethereum)
So what? Architecture of a proposed application
Pharo
Pharo: Why?

• Blockchain is a multiple actors always growing environment.

• Blockchain is a living environment
  • Transactions move money (ether - bitcoin) from one place to other
  • Transactions execute smart contracts

• Ethereum is a distributed runtime. Nothing better than a live environment for a living distributed runtime.

• A lot of code analysis and inspection state-of-the-art tools
Fog

- Pharo client for the Ethereum client (GEth)
- github.com/sbragagnolo/Fog
Fog - features

• Connection, communication, marshalling, etc.
• Block fetching
• Query and create transactions
• Query and create contracts
• Remote method invocation
Fog - features

- Development support
  - First-class citizen navigation (GT-Tools)
    - Accounts
    - Blocks
    - Transactions
    - Contracts
  - Automatic contract mirror generation
  - Automatic contract proxy building
Fog - features

• Cache
• General
• Connection
• Session
Some fancy slides :)
Block inspection

• Navigating blocks

• Inspecting blocks individually

• Overview of a collection of blocks through statistics

• Overview of the transactions of a collection of blocks
Navigating in blocks
Blocks overview
Transactions overview
pragma solidity ^0.4.2;

contract StructTestContract {

    enum myenum { A, B, C }

    struct mystruct {
        bool boolean;
        myenum uservalue;
        uint32 commonvalue;
    }

    address _owner;
    bool bool1;
    int16 midint;
    mystruct simpleExample;
    bool bool2;
    mystruct[] arrayExample;

    function StructTestContract (){
        _owner = msg.sender;
        bool1 = true;
        bool2 = true;
        midint = 32;
        simpleExample.boolean = true;
        simpleExample.uservalue = myenum.B;
        simpleExample.commonvalue = 6355432;
        arrayExample.push(mystruct(true, myenum.A, 134));
        arrayExample.push(mystruct(false, myenum.B, 235));
        arrayExample.push(mystruct(true, myenum.C, 34));
    }

    function kill() {
        suicide(_owner);
    }
}
Inspecting contract

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_owner</code></td>
<td>'0xb4ebf466889c4a0239379125a7d0f9c4e8bf2a14'</td>
</tr>
<tr>
<td>'bool1'</td>
<td>true</td>
</tr>
<tr>
<td>'midint'</td>
<td>32</td>
</tr>
<tr>
<td>'simpleExample'</td>
<td>a Dictionary [3 items] ('boolean'-&gt;true 'commonvalue'-&gt;6355432 'uservalue'-&gt;'B')</td>
</tr>
<tr>
<td>'bool2'</td>
<td>true</td>
</tr>
<tr>
<td>'arrayExample'</td>
<td>' an array of 3 elements '</td>
</tr>
</tbody>
</table>

Quick selection field. Given your INPUT, it executes: self select: [:each | INPUT ]
Inspecting structs
Yet to implement
Fog - Demo
Fog - future

- Finishing session management
- Events support
- Transactional message send recognition
- New AST Definition (Henrique Rocha)
THANKS :)}!

by Santiago Bragagnolo - PharoDays - 2017
santiago.bragagnolo@gmail.com
santiago.bragagnolo@inria.fr
skype:santiago.bragagnolo
@sbragagnolo