Smart Contract Tools

https://github.com/bashalex/BuildStuff

Workshop

Alexey Bashlykov
Chief Technology Officer at Zerion
alexey@zerion.io
SOLIDITY

- Turing complete
- Statically typed
- Supports inheritance
- Supports libraries
- Similar to JavaScript
ALTERNATIVES

Serpent
- similar to python
- written in C++

Viper
- similar to python as well
- new and experimental
- introduce such features as bounds and overflow checking etc
REMIX

https://ethereum.github.io/browser-solidity/

→ Online IDE
→ Compilation and gas cost estimation
→ JavaScript VM
→ Deployment
→ Debug
pragma solidity ^0.4.2;

contract StandardToken {

    /* Data structures */
    mapping (address => uint256) balances;
    mapping (address => mapping (address => uint256)) allowed;

    uint256 public totalSupply;

    /* Events */
    event Transfer(address indexed from, address indexed to, uint256 value);
    event Approval(address indexed owner, address indexed spender, uint256 value);

    /* Read and write storage functions */
    /// @dev Transfers sender's tokens to a given address. Returns success.
    /// @param _to Address of token receiver.
    /// @param _value Number of tokens to transfer.
    /// @return success true if transfer succeeds, false if it fails.
    function transfer(address _to, uint256 _value) returns (bool success) {
        if (balances[msg.sender] >= _value && _value > 0) {
            balances[msg.sender] -= _value;
            balances[_to] += _value;
            Transfer(msg.sender, _to, _value);
            return true;
        } else {
            return false;
        }
    }

    /// @dev Allows allowed third party to transfer tokens from one address to another.
    /// @param _from Address from where tokens are withdrawn.
    /// @param _to Address to where tokens are sent.
    /// @param _value Number of tokens to transfer.
    function transferFrom(address _from, address _to, uint256 _value) returns (bool success) {
        if (balances[_from] >= _value && allowed[_from][msg.sender] >= _value && balances[_to] >= _value) {
            balances[_from] -= _value;
            balances[_to] += _value;
            balances[msg.sender] -= _value;
            approved[_from][msg.sender] -= _value;
            transfer(msg.sender, _to, _value);
            return true;
        } else {
            return false;
        }
    }
}

// SPDX-License-Identifier: MIT

import { StandardToken } from './StandardToken.sol';
Truffle + TestRPC

- Testing and debugging
- Instant blocks
- Dapps deployment
- Easy configuration
Thank you!

Alexey Bashlykov
Chief Technology Officer at Zerion
alexey@zerion.io