ORS Token Sale

Release 1

SICOS

May 04, 2018

CONTENTS

1 ORSToken 1

2 ORSTokenSale 3

CHAPTER

ONE

ORSTOKEN

```
pragma solidity 0.4.23;

import "../zeppelin-solidity/contracts/token/ERC20/CappedToken.sol";
import "../zeppelin-solidity/contracts/token/ERC20/PausableToken.sol";
import "../zeppelin-solidity/contracts/token/ERC20/StandardBurnableToken.sol";


/// @title ORSToken
/// @author Autogenerated from a Dia UML diagram
contract ORSToken is CappedToken, StandardBurnableToken, PausableToken {

    string public name = "ORS Token";
    string public symbol = "ORS";
    uint8 public decimals = 18;

    /// @dev Constructor
    /// @param _cap A positive number
    constructor(uint _cap) public CappedToken(_cap) {
        pause();
    }

}
```

CHAPTER

TWO

ORSTOKENSALE

```
pragma solidity 0.4.23;

import "./ORSToken.sol";
import "../eidoo-icoengine/contracts/ICOEngineInterface.sol";
import "../eidoo-icoengine/contracts/KYCBase.sol";
import "../zeppelin-solidity/contracts/math/SafeMath.sol";
import "../zeppelin-solidity/contracts/ownership/Ownable.sol";


/// @title ORSTokenSale
/// @author Autogenerated from a Dia UML diagram
contract ORSTokenSale is KYCBase, ICOEngineInterface, Ownable {

    using SafeMath for uint;

    uint constant public MAINSALE_CAP = 500000000e18;   // 500,000,000 e18
    uint constant public PRESALE_CAP = 0;               // TBD!
    uint constant public BONUS_CAP = 64460000e18;       //  64,460,000 e18
    uint constant public TEAM_CAP = 83333333e18;        //  83,333,333 e18
    uint constant public COMPANY_CAP = 127206667e18;    // 127,206,667 e18
    uint constant public ADVISORS_CAP = 58333333e18;    //  58,333,333 e18

    uint public mainsaleRemaining = MAINSALE_CAP;
    uint public presaleRemaining = PRESALE_CAP;
    uint public bonusRemaining = BONUS_CAP;

    address public teamWallet;
    address public companyWallet;
    address public advisorsWallet;

    ORSToken public token;

    uint public rate;  // integral token units (10^-18 tokens) per wei

    uint public openingTime;
    uint public closingTime;

    address public wallet;

    address public eidooSigner;

    bool public isFinalized = false;

    /// @dev Log entry on rate changed
    /// @param newRate A positive number
    event RateChanged(uint newRate);

    /// @dev Log entry on token purchased
    /// @param buyer An Ethereum address
```

```
50      /// @param value A positive number
51      /// @param tokens A positive number
52      event TokenPurchased(address indexed buyer, uint value, uint tokens);
53
54      /// @dev Log entry on buyer refunded
55      /// @param buyer An Ethereum address
56      /// @param value A positive number
57      event BuyerRefunded(address indexed buyer, uint value);
58
59      /// @dev Log entry on finalized
60      event Finalized();
61
62      /// @dev Constructor
63      /// @param _token An ORSToken
64      /// @param _rate A positive number
65      /// @param _openingTime A positive number
66      /// @param _closingTime A positive number
67      /// @param _wallet An Ethereum address
68      /// @param _teamWallet An Ethereum address
69      /// @param _companyWallet An Ethereum address
70      /// @param _advisorsWallet An Ethereum address
71      /// @param _kycSigners A list where each entry is an Ethereum address
72      constructor(
73          ORSToken _token,
74          uint _rate,
75          uint _openingTime,
76          uint _closingTime,
77          address _wallet,
78          address _teamWallet,
79          address _companyWallet,
80          address _advisorsWallet,
81          address[] _kycSigners
82      )
83          public
84          KYCBase(_kycSigners)
85      {
86          require(_token != address(0x0));
87          require(_token.cap() == MAINSALE_CAP + PRESALE_CAP + BONUS_CAP + TEAM_CAP + COMPANY_CAP +␣
    ↪ADVISORS_CAP);
88          require(_rate > 0);
89          require(_openingTime > now && _closingTime > _openingTime);
90          require(_wallet != address(0x0));
91          require(_teamWallet != address(0x0) && _companyWallet != address(0x0) && _advisorsWallet !=␣
    ↪address(0x0));
92          require(_kycSigners.length >= 2);
93
94          token = _token;
95          rate = _rate;
96          openingTime = _openingTime;
97          closingTime = _closingTime;
98          wallet = _wallet;
99          teamWallet = _teamWallet;
100         companyWallet = _companyWallet;
101         advisorsWallet = _advisorsWallet;
102
103         eidooSigner = _kycSigners[0];
104     }
105
106     /// @dev Set rate
107     /// @param newRate A positive number
108     function setRate(uint newRate) public onlyOwner {
109         require(newRate > 0);
110
```

```
111         if (newRate != rate) {
112             rate = newRate;
113 
114             emit RateChanged(newRate);
115         }
116     }
117 
118     /// @dev Distribute presale
119     /// @param investors A list where each entry is an Ethereum address
120     /// @param tokens A list where each entry is a positive number
121     /// @param bonuses A list where each entry is a positive number
122     function distributePresale(address[] investors, uint[] tokens, uint[] bonuses) public onlyOwner
↪{
123         require(ended() && !isFinalized);
124         require(investors.length == tokens.length && bonuses.length == tokens.length);
125 
126         for (uint i = 0; i < investors.length; ++i) {
127             presaleRemaining = presaleRemaining.sub(tokens[i]);
128             bonusRemaining = bonusRemaining.sub(bonuses[i]);
129 
130             token.mint(investors[i], tokens[i].add(bonuses[i]));
131         }
132     }
133 
134     /// @dev Started
135     /// @return True or false
136     function started() public view returns (bool) {
137         return now > openingTime;
138     }
139 
140     /// @dev Ended
141     /// @return True or false
142     function ended() public view returns (bool) {
143         return now > closingTime || mainsaleRemaining == 0;
144     }
145 
146     // time stamp of the starting time of the ico, must return 0 if it depends on the block number
147     /// @dev Start time
148     /// @return A positive number
149     function startTime() public view returns (uint) {
150         return openingTime;
151     }
152 
153     // time stamp of the ending time of the ico, must retrun 0 if it depends on the block number
154     /// @dev End time
155     /// @return A positive number
156     function endTime() public view returns (uint) {
157         return closingTime;
158     }
159 
160     // returns the total number of the tokens available for the sale, must not change when the ico␣
↪is started
161     /// @dev Total tokens
162     /// @return A positive number
163     function totalTokens() public view returns (uint) {
164         return MAINSALE_CAP;
165     }
166 
167     // returns the number of the tokens available for the ico. At the moment that the ico starts it␣
↪must be
168     // equal to totalTokens(), then it will decrease. It is used to calculate the percentage of␣
↪sold tokens as
169     // remainingTokens() / totalTokens()
```

```
    /// @dev Remaining tokens
    /// @return A positive number
    function remainingTokens() public view returns (uint) {
        return mainsaleRemaining;
    }

    // return the price as number of tokens released for each ether
    /// @dev Price
    /// @return A positive number
    function price() public view returns (uint) {
        return rate;
    }

    /// @dev Finalize
    function finalize() public onlyOwner {
        require(ended() && !isFinalized);
        require(presaleRemaining == 0);

        token.mint(teamWallet, TEAM_CAP);
        token.mint(companyWallet, COMPANY_CAP);
        token.mint(advisorsWallet, ADVISORS_CAP);

        if (bonusRemaining > 0) {
            token.mint(companyWallet, bonusRemaining);
        }

        token.finishMinting();
        token.unpause();
        token.transferOwnership(owner);

        isFinalized = true;

        emit Finalized();
    }

    /// @dev Release tokens to
    /// @param buyer An Ethereum address
    /// @param signer An Ethereum address
    /// @return True or false
    function releaseTokensTo(address buyer, address signer) internal returns (bool) {
        require(started() && !ended());

        uint value = msg.value;
        uint refund = 0;

        uint tokens = value.mul(rate);
        uint bonus = 0;

        if (tokens > mainsaleRemaining) {
            uint valueOfRemaining = mainsaleRemaining.div(rate);

            refund = value.sub(valueOfRemaining);
            tokens = mainsaleRemaining;
            value = valueOfRemaining;
            // Note:
            // To be 100% accurate the buyer should have received only the token amount that
            // correspondents to valueOfRemaining, i.e. tokens = valueOfRemaining.mul(rate)...,
            // because of mainsaleRemaining may not be a multiple of rate.
            // Nevertheless, we deliver all mainsaleRemaining as the worth of these additional
            // tokens (amount < rate) is less than a wei.
        }

        if (signer == eidooSigner) {
```

```
            bonus = tokens.div(20);
        }

        mainsaleRemaining = mainsaleRemaining.sub(tokens);
        bonusRemaining = bonusRemaining.sub(bonus);

        token.mint(buyer, tokens.add(bonus));
        wallet.transfer(value);
        if (refund > 0) {
            buyer.transfer(refund);

            emit BuyerRefunded(buyer, refund);
        }

        emit TokenPurchased(buyer, value, tokens);

        return true;
    }

}
```