

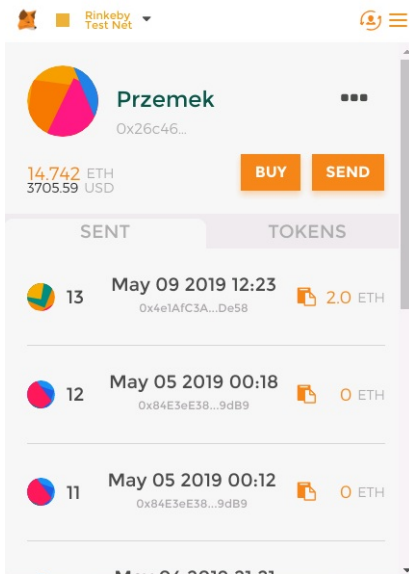
ETH Token

Wymagane komponenty

- portfel na kryptowaluty (np. Metamask),
- Remix (IDE do języka Solidity - [LINK](#))
- npm (tworzenie projektu React, instalacja potrzebnych pakietów)

Konfiguracja Metamask

Metamask można zainstalować pod dwiema postaciami, jako rozszerzenie do przeglądarek opartych na Chromium (Chrome, Opera) lub razem z przeglądarką Brave Browser. Obie opcje dostępne są pod adresem <https://metamask.io>. Po założeniu konta ekran główny aplikacji powinien wyglądać tak jak na poniższym zrzucie ekranu (lub podobnie).



Jeżeli w lewym górnym rogu pojawiła się inna sieć należy ją zmienić na Rinkeby Test Net poprzez kliknięcie lewym przyciskiem myszy i wybranie odpowiedniej opcji z listy.

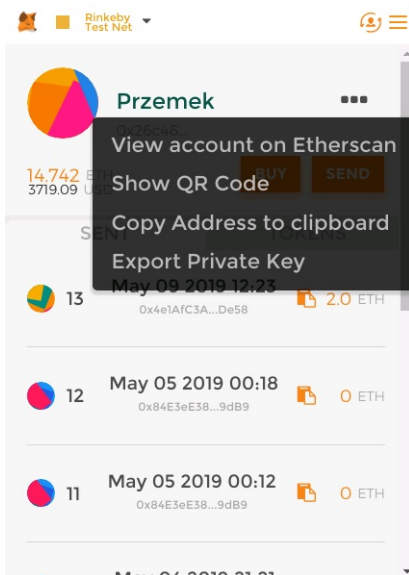
W celu uruchomienia naszego kontraktu na sieci Rinkeby będziemy potrzebowali Etheru, którym zapłacimy za deploy kontraktu. Darmowy Ether możemy uzyskać pod adresem <https://faucet.rinkeby.io>. Strona powinna wyglądać tak jak poniżej.

Rinkeby Authenticated Faucet

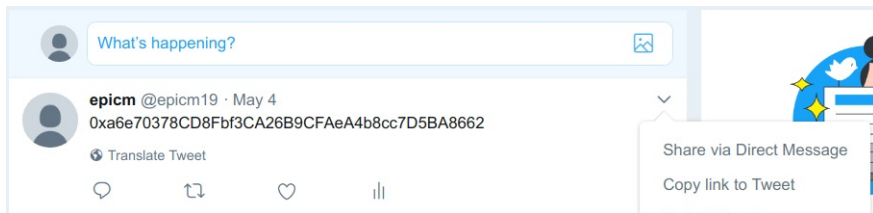
Give me Ether ▼

4 peers 4428108 blocks 9.046256971665328e+56 Ethers 281969 funded

W celu uzyskania Etheru konieczne jest (w celach walidacyjnych) napisanie posta, w dowolnych mediach społecznościowych, który będzie zawierał adres naszego portfela. Sam adres możemy uzyskać z aplikacji Metamask poprzez kliknięcie trzech kropek przy nazwie konta i wybieraniu opcji Copy address to clipboard.



Adres publikujemy na dowolnym serwisie społecznościowym (np. Twitter) w poniższy sposób.



Następnie wybieramy opcję Copy link to Tweet i otrzymany adres wklejamy na stronie <https://faucet.rinkeby.io> jednocześnie wybierając dowolną z opcji po prawej stronie.

Rinkeby Authenticated Faucet

Give me Ether ▾

4 peers

4428108 blocks

9.046256971665328e+56 Ethers

28196

3 Ethers / 8 hours

7.5 Ethers / 1 day

18.75 Ethers / 3 days

Po wybraniu jednej z opcji środki powinny zostać przelane na odpowiednie konto.

Deploy kontraktu

Każda operacja na blockchainie wykonywana jest poprzez wykorzystanie tzw. kontraktu, który definiuje wszystkie wykonywane akcje. Poniżej znajduje się kod pięciu kontraktów. Cztery z nich to kontrakty pomocnicze, natomiast ostatni jest kontraktem głównym, który zawiera wszelkie informacje o naszym kontrakcie.

Kontrakt Safe-Math

Podstawowym celem tego kontraktu jest zapewnienie bezpiecznych operacji dodawania, odejmowania, mnożenia i dzielenia liczb z wykluczeniem możliwości wystąpienia przepełnienia.

```
// Safe Math - used to deal with overflows and divide_by_zero exceptions
contract SafeMath {
    function add(uint a, uint b) public pure returns (uint c) {
        c = a + b;
        require(c >= a);
    }

    function sub(uint a, uint b) public pure returns (uint c) {
        require(b <= a);
        c = a - b;
    }

    function mult(uint a, uint b) public pure returns (uint c) {
        c = a * b;
        require(a == 0 || c / a == b);
    }

    function div(uint a, uint b) public pure returns (uint c) {
        require(b > 0);
        c = a / b;
    }
}
```

Kontrakt ERC20Interface

Poniższy kontrakt definiuje standardowy interfejs dla tokenów opartych na Ethereum. Zawiera on podstawowe metody takie jak sprawdzenie stanu konta pod podanym adresem czy przelew środków z jednego portfela na drugi.

```
// -----
// ERC Token Standard #20 Interface
// https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20-token-standard.md
// -----
contract ERC20Interface {
    function totalSupply() public view returns (uint);
    function balanceOf(address tokenOwner) public view returns (uint balance);
    function allowance(address tokenOwner, address spender) public view returns (uint remaining);
    function transfer(address to, uint tokens) public returns (bool success);
    function approve(address spender, uint tokens) public returns (bool success);
    function transferFrom(address from, address to, uint tokens) public returns (bool success);

    event Transfer(address indexed from, address indexed to, uint tokens);
    event Approval(address indexed tokenOwner, address indexed spender, uint tokens);
}
```

Kontrakt ApproveAndCallFallback

Kontrakt ten zawiera funkcję wymaganą według standardu ERC20, pozwala ona aplikacjom na transfer środków z konta wydającego (po otrzymaniu zgody).

```
// -----
// Contract function to receive approval and execute function in one call
//
// Source: MiniMeToken
// -----
contract ApproveAndCallFallback {
    function receiveApproval(address from, uint256 tokens, address token, bytes memory data) public;
}
```

Kontrakt Owned

Kontrakt ten określa właściciela tokenu oraz jakie dodatkowe akcje na nim może on przeprowadzić.

```
contract Owned {
    address public owner;
    address public newOwner;

    event OwnerChanged(address indexed _from, address indexed _to);

    constructor() public {
        owner = msg.sender;
    }

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }
}
```

Konstruktor kontraktu zapewnia, że pierwszym właścicielem kontraktu będzie osoba, która dokonała jego deployu (sender).

Główny kontrakt tokenu (TeacheCoin)

Dwie najważniejsze części kontraktu to przedstawione poniżej mapowanie adresów, które określa balans każdego konta oraz konstruktor, w którym definiujemy główne informacje o naszym kontrakcie.

Mapowanie stanów konta do adresów.

Definiowanie podstawowych właściwości kontraktu

```
constructor() public {  
    symbol = "ThC";  
    name = "TeacheCoin";  
    decimals = 18;  
    _totalSupply = 100000000000000000000000000000000; // 1e+30  
    userBalances[<DEFAULT_ADDRESS>] = _totalSupply;
```

```
} emit Transfer(address(0), <DEFAULT_ADDRESS>, _totalSupply);
}
```

W powyższym kodzie symbol oznacza skrótową nazwę naszego tokenu, name oznacza nazwę pod którą będzie on znany, natomiast pole decimals określa na jak małe części będzie można rozbić nasz token (standardową i zalecaną wartością jest 18). Kolejne pole _totalSupply określa ile tokenów jest w obiegu (nieużywane tokeny mogą być przetrzymywane na głównym koncie). W ostatnich dwóch liniach ustawiamy stan naszego konta głównego tak, by zawierał wszystkie możliwe tokeny, a następnie wysyłamy wiadomość o transferze środków.

Na samym początku kodu definiujemy kompilator, z którego chcemy skorzystać.

```
pragma solidity ^0.5.8;
```

Przesyłanie kontraktów na blockchain

Na głównym ekranie Remix IDE wybieramy kompilator, który zostanie użyty do skompilowania kontraktów (po prawej stronie), a następnie wybieramy opcję Start to compile.

```
1 pragma solidity ^0.5.8;
2
3 // -----
4 // Teache Token Contract
5 // Deploy address: 0xe325ba712b216e9619439a5a0d08995237a5cf0e
6 // Symbol: ThC
7 // Name: Teache Coin
8 // Total supply: 100000000
9 // Decimals: 18
10 // -----
11
12 // Safe Math - used to deal with overflows and divide_by_zero exceptions
13 contract SafeMath {
14     function add(uint a, uint b) public pure returns (uint c) {
15         c = a + b;
16         require(c >= a);
17     }
18
19     function sub(uint a, uint b) public pure returns (uint c) {
20         require(b <= a);
21         c = a - b;
22     }
23
24     function mult(uint a, uint b) public pure returns (uint c) {
25         c = a * b;
26         require(a == 0 || c / a == b);
27     }
28
29     function div(uint a, uint b) public pure returns (uint c) {
30         require(b > 0);
31         c = a / b;
32     }
33 }
```

Current
version:0.5.8+commit.23d335f2.Emscripten.clang

Select new compiler version

☐ Auto compile ☐ Enable Optimization ☐ Hide warnings

Start to compile (Ctrl-S)

ApproveAndCallFallBack

Details ABI Bytecode

Static Analysis raised 7 warning(s) that requires your attention. Click here to show the warning(s).

ApproveAndCallFallBack

ERC20Interface

Owned

SafeMath

TeacheCoin

Przechodzimy do zakładki Run i wybieramy nasz główny kontrakt z listy (reszta kontraktów zostanie uruchomiona automatycznie). Następnie zmieniamy Environment na Injected Web3 (pozwoli to na wykrycie naszego konta Metamask). Następnie klikamy w przycisk Deploy i zatwierdzamy dokonanie transakcji.

Environment: Injected Web3 Rinkeby (4)

Account: 0x26c...237df (14.742569685 ether)

Gas limit: 3000000

Value: 0 wei

TeacheCoin

Deploy

or

At Address: Load contract from Address

MetaMask Notification - Opera

CONFIRM TRANSACTION Rinkeby Test Net

Przemek 26c46D...37Df 14,742 ETH 3637.49 USD New Contract

Amount: 0.00 ETH 0.00 USD

Gas Limit: 1807089 UNITS

Gas Price: 1 GWEI

Max Transaction Fee: 0.001807 ETH 0.45 USD

Max Total: 0.001807 ETH 0.45 USD

Data included: 6658 bytes

RESET SUBMIT REJECT

Po dokonaniu tych czynności powinniśmy uzyskać adres dokonanej transakcji (znajduje się on na dole ekranu w jego centralnej części).

```
creation of TeacheCoin pending...
https://rinkeby.etherscan.io/tx/0xefcc685bf29da7ed52b97113e87344f4b190d4622fdfa1985d8935bdc8e6b9d
[block:4428500 txIndex:8] from:0x26c...237df to:TeacheCoin.(constructor) value:0 wei
data:0x608...00029 logs:1 hash:0xefc...eb9d
```

Transaction Details

Overview

Event Logs (1)

[This is a Rinkeby Testnet Transaction Only]

Transaction Hash:

0xfec685bf29da7ed52b97113e87344f4b190d4622dfa1985d8935bdc8e6b9d

Status:

Success

Block:

4428500

6 Block Confirmations

Timestamp:

1 min ago (May-22-2019 08:31:20 PM +UTC)

From:

0x26c46d907d0c8f5b604f7421521220b99c9237df

To:

[Contract 0x3f9f37bc57415a7fcf2ff6c2407804f1e69e4f30 Created]

Tokens Transferred:

From 0x0000000000000000... To 0x26c46d907d0c8f5... For 100,000,000 ERC-20 (TES)

Value:

0 Ether (\$0.00)

Transaction Fee:

0.001807089 Ether (\$0.000000)

Click to see More

Na stronie transakcji klikamy w adres znajdujący się w polu To, a następnie przechodzimy do zakładki Code.

Contract 0x3F9F37Bc57415a7fCf2fF6c2407804F1E69E4f30

Contract Overview

Balance:

0 Ether

More Info

My Name Tag:

Not Available

Contract Creator:

0x26c46d907d0c8f5... at txn 0xfec685bf29da7e...

Token Tracker:

TestCoin (TES)

Transactions

Code

Events

Are you the contract creator? Verify and Publish your contract source code today!

Switch to Opcodes View

Find Similar Contracts

0x60806040526004361061011f5760003560e01c806395d89b41116100a0578063b67d77c511610064578063b67d77c514610673578063cae9ca51146106cc578063d4ee1d90146107d6578063dc39d06d1461082d578063dd62ed3e146108a05761011f565b806395d89b411461046d5780639aa727f6146104fd578063a391c15b14610556578063a6f9dae1146105af578063a9059cbb146106095761011f565b80633eaaaf86b116100e75780633eaaaf86b1461031657806379a0823114610341578063771602f7146103a657806379ba5097146103fff5780638da5cb5b146104165761011f565b806306fdd0314610124578063095ea7b3146101b457806318160ddd1461022757806323b872dd14610252578063313ce567146102e5575b600080fd5b34801561013057600080fd5b50610139610925565b6040518080602001828103825283818151815260200191508051906020019080838360005b8381101561017957808201518184015260208101905061015e565b505050905090810190601f1680156101a65780820380516001836020036101000a031916815260200191505b509250505060405180910390f35b3480156101c057600080fd5b5061020d600480360360408110156101d757600080fd5b81019080803573fffffffffffffffffffffffffffffffffffff169060200190929190803590602001909291905050506109c3565b604051808215151515815260200191505060405180910390f35b34801561025e57600080fd5b506102cb0048036036060081101561027557600080fd5b81019080803573fffffffffffffffffffffffffffffffffffff169060200190929190803573fffffffffffffffffffffffffffffffffffff16906020019092919080359060200190929190505050610b0b00565b604051808215151515815260200191505060405180910390f35b3480156102f157600080fd5b506102fa610d90565b604051808260ff1660ff16815260200191505060405180910390f35b34801561032257600080fd5b5061032b610da3565b6040518082815260200191505060405180910390f35b34801561034d57600080fd5b506103906004803603602081101561036457600080fd5b81019080803573fffffffffffffffffffffffffffffffffffff169060200190929190505050610da9565b6040518082815260200191505060405180910390f35b3480156103b257600080fd5b506103e9600480360360408110156103c957600080fd5b810190808035906020019092919080359060200190929190505050610df2565b6040518082815260200191505060405180910390f35b34801561040b57600080fd5b50610414610e0c565b6005b34801561042257600080fd5b5061042b610fa9565b604051808273fffffffffffffffffffffffffffff

Swarm Source

bzzr://e76648584befb31292ae00b8b8f3c9a4da91304e452b6f5fd1ac026b52da11e0

W celu weryfikacji i opublikowania naszego kontraktu klikamy w opcję Verify and Publish. Następnie wybieramy odpowiedni kompilator oraz jego wersję.

Please enter the Contract Address you would like to verify

<TUTAJ ADRES KONTRAKTU>

Invalid Length

Please select Compiler Type

Solidity (Single file)

Please select Compiler Version

v0.5.8+commit.23d335f2

☒ Un-Check to show all nightly Commits also

Continue

Reset

W polu Enter the Solidity Contract Code below wklejamy cały kod naszego kontraktu z Remix IDE, a następnie wybieramy opcję Verify and Publish na dole strony. Po udanej operacji zostanie wyświetlona odpowiednia strona, na dole której znajduje się pole ContractABI, którego zawartość pozwoli nam na oddziaływanie z kontraktem na poziomie języka JavaScript.

Contract Address

0x3f9f37bc57415a7fc2ff6c2407804f1e69e4f30

Compiler

v0.5.8+commit.23d335f2

Optimization

No

Enter the Solidity Contract Code below

Fetch from Gist

```
pragma solidity ^0.5.8;

// -----
// Teache Token Contract
// Deploy address:      0xe325ba712b216e9619439a5a0d08995237a5cf0e
// Symbol:              ThC
// Name:                Teache Coin
// Total suply:         100000000
// Decimals:            18
// -----
```

Contract Source Code

Compiler Output

Compiler debug log:

Note: Contract was created during TxHash# 0xf6cc685bf29da7ed52b97113e87344f4b190d4622dfda1985d8935bdc8e6b9d

Successfully generated ByteCode and ABI for Contract Address [0x3f9f37bc57415a7fc2ff6c2407804f1e69e4f30]

ContractABI:

```
[{"constant":true,"inputs":[],"name":"name","outputs":[{"name":"","type":"string"}],"payable":false,"stateMutability":"view","type":"function"}, {"constant":false,"inputs":[{"name":"spender","type":"address"}, {"name":"tokens","type":"uint256"}],"name":"approve","outputs":[{"name":"success","type":"bool"}],"payable":false,"stateMutability":"nonpayable","type":"function"}, {"constant":true,"inputs":[],"name":"totalSupply","outputs":[{"name":"","type":"uint256"}],"payable":false,"stateMutability":"view","type":"function"}, {"constant":false,"inputs":[{"name":"from","type":"address"}, {"name":"to","type":"address"}, {"name":"tokens","type":"uint256"}],"name":"transferFrom","outputs":[{"name":"success","type":"bool"}],"payable":false,"stateMutability":"nonpayable","type":"function"}, {"constant":true,"inputs":[],"name":"decimals","outputs":[{"name":"","type":"uint8"}],"payable":false,"stateMutability":"view","type":"function"}, {"constant":true,"inputs":[],"name":"_totalSupply","outputs":[{"name":"","type":"uint256"}],"payable":false,"stateMutability":"view","type":"function"}, {"constant":true,"inputs":[{"name":"tokenOwner","type":"address"}],"name":"balanceOf","outputs":[{"name":"balance","type":"uint256"}],"payable":false,"stateMutability":"view","type":"function"}, {"constant":true,"inputs":[{"name":"a","type":"uint256"}, {"name":"b","type":"uint256"}],"name":"add","outputs":[{"name":"c","type":"uint256"}],"payable":false,"stateMutability":"pure","type":"function"}, {"constant":false,"inputs":[],"name":"acceptOwnership","outputs":[{"name":"success","type":"bool"}],"payable":false,"stateMutability":"nonpayable","type":"function"}, {"constant":true,"inputs":[{"name":"owner","type":"address"}],"name":"owner","outputs":[{"name":"","type":"string"}],"payable":false,"stateMutability":"view","type":"function"}, {"constant":true,"inputs":[{"name":"symbol","type":"string"}],"name":"symbol","outputs":[{"name":"","type":"string"}],"payable":false,"stateMutability":"view","type":"function"}, {"constant":true,"inputs":[{"name":"a","type":"uint256"}, {"name":"b","type":"uint256"}],"name":"mult","outputs":[{"name":"c","type":"uint256"}],"payable":false,"stateMutability":"pure","type":"function"}, {"constant":true,"inputs":[{"name":"a","type":"uint256"}, {"name":"b","type":"uint256"}],"name":"div","outputs":[{"name":"c","type":"uint256"}],"payable":false,"stateMutability":"pure","type":"function"}, {"constant":false,"inputs":[{"name":"newOwner","type":"address"}],"name":"changeOwner","outputs":[{"name":"success","type":"bool"}],"payable":false,"stateMutability":"nonpayable","type":"function"}, {"constant":false,"inputs":[{"name":"to","type":"address"}, {"name":"tokens","type":"uint256"}],"name":"transfer","outputs":[{"name":"success","type":"bool"}],"payable":false,"stateMutability":"nonpayable","type":"function"}, {"constant":true,"inputs":[{"name":"a","type":"uint256"}, {"name":"b","type":"uint256"}],"name":"sub","outputs":[{"name":"c","type":"uint256"}],"payable":false,"stateMutability":"pure","type":"function"}, {"constant":false,"inputs":[{"name":"spender","type":"address"}, {"name":"tokens","type":"uint256"}, {"name":"data","type":"bytes"}],"name":"approveAndCall","outputs":[{"name":"success","type":"bool"}],"payable":false,"stateMutability":"nonpayable","type":"function"}, {"constant":true,"inputs":[{"name":"newOwner","type":"address"}],"name":"newOwner","outputs":[{"name":"","type":"string"}],"payable":false,"stateMutability":"view","type":"function"}, {"constant":false,"inputs":[{"name":"tokenAddress","type":"address"}, {"name":"tokens","type":"uint256"}],"name":"transferAnyERC20Token","outputs":[{"name":"success","type":"bool"}],"payable":false,"stateMutability":"nonpayable","type":"function"}, {"constant":true,"inputs":[{"name":"tokenOwner","type":"address"}, {"name":"spender","type":"address"}],"name":"allowance","outputs":[{"name":"remaining","type":"uint256"}],"payable":false,"stateMutability":"view","type":"function"}, {"inputs":[{"name":"","type":"address"}],"name":"constructor","payable":true,"stateMutability":"payable","type":"fallback"}, {"anonymous":false,"inputs":[{"indexed":true,"name":"","from","type":"address"}, {"indexed":true,"name":"","to","type":"address"}],"name":"OwnerChanged","type":"event"}, {"indexed":true,"name":"","from","type":"address"}, {"indexed":true,"name":"","to","type":"address"}, {"indexed":false,"name":"","tokens","type":"uint256"}],"name":"Approval","type":"event"}]
```

Zawartość pola ContractABI zostanie umieszczona w kodzie źródłowym naszej strony (src/tokens/TeacheCoin.js) w przedstawionej poniżej postaci.

```
const Coin = {
  address: 'ADRES TOKENU',
  decimal: LICZBA MIEJSC PO PRZECINKU,
  name: 'NAZWA TOKENU',
  symbol: 'SYMBOL TOKENU',
  abi: ZAWARTOŚĆ ContractABI
}

export default Coin;
```

Przedstawiony powyżej kod pozwoli nam na odnoszenie się do naszego kontraktu z poziomu języka JavaScript.

Używanie kontraktu

Do implementacji aplikacji umożliwiającej przelew środków na dowolny portfel ETH wykorzystamy bibliotekę React.js. Aby rozpocząć projekt będziemy potrzebować npm (Link). Po zainstalowaniu menadżera pakietów przechodzimy do folderu, w którym chcemy rozpocząć projekt, uruchamiamy konsolę i wpisujemy polecenie:

```
npm init react-app eth-token
```

Szczegółowe informacje dotyczące inicjacji projektu React.js znajdują się pod adresem https://github.com/facebook/create-react-app.

Po utworzeniu projektu kopiujemy plik TeacheCoin.js stworzony w poprzednim etapie do folderu src/tokens/. Następnie zamieniamy zawartość pliku App.js na poniższą

```
import React, { Component } from 'react';
import TeacheCoin from './tokens/TeacheCoin'
import './App.css';

class SendTokens extends Component {

  constructor() {
    super();

    this.state = {
      balance: 0,
      destWallet: '',
      destAmount: 0
    }

    this.isWeb3 = false;
```

```
this.isWeb3Locked = false;

this.loadBalance = this.loadBalance.bind(this)
this.checkWeb3Compatibility = this.checkWeb3Compatibility.bind(this)
}

componentDidMount() {
  window.addEventListener('load', this.checkWeb3Compatibility)

  if (window.web3) {
    window.web3.currentProvider.publicConfigStore.on('update', () => {
      this.checkWeb3Compatibility()
    })
  }
}

checkWeb3Compatibility() {
  if (window.web3) {
    this.isWeb3 = true;
    window.web3.eth.getCoinbase((error, coinbase) => {
      if (error || coinbase === null) {
        this.isWeb3Locked = true;
      } else {
        this.isWeb3Locked = false;
        this.setState({
          account: coinbase,
          token: window.web3.eth.contract(TeacheCoin.abi).at(TeacheCoin.address)
        }, () => {
          this.loadBalance()
        })
      }
    })
  } else {
    this.isWeb3 = false;
  }
}

loadBalance() {
  if (this.isWeb3) {
    window.web3.eth.getCoinbase((error, coinbase) => {
      let token = this.state.token
      token.balanceOf(coinbase, (error, response) => {
        if (!error) {
          let balance = response.c[0] / 10000
          balance = balance >= 0 ? balance : 0

          this.setState({
            balance: balance,
            symbol: TeacheCoin.symbol,
            decimal: '1e' + TeacheCoin.decimal
          })
        }
      })
    })
  }
}

sendTokens = (event) => {
  event.preventDefault()

  const balance = this.state.balance
  const amount = this.state.destAmount;
  const target = this.state.destWallet
  const token = this.state.token
  const decimals = this.state.decimal;

  if (amount <= balance && amount > 0 && token) {
    token.transfer(target, amount * decimals, (error, response) => {
      if (error || error !== null) {
        alert(error);
      } else {
        alert('Pomyślnie wysłano ' + amount + ' ' + this.state.symbol);
        this.loadBalance()
      }
    })
  }
}

handleAddressChange = (event) => {
  this.setState({
    destWallet: event.target.value
  })
}

handleAmountChange = (event) => {
  this.setState({
    destAmount: event.target.value
  })
}

render() {
  return (
    <div className="main-container">
      <p className="form-label">Stan konta</p>
      <p className="form-balance">{this.state.balance} {this.state.symbol}</p>

      <p className="form-label">Portfel odbiorcy</p>
      <input type="text" className="usr-input" onChange={e => this.handleAddressChange(e)} />

      <p className="form-label">Liczba tokenów do wysłania</p>
      <input type="number" className="usr-input"
        min="0" max={this.state.balance} onChange={e => this.handleAmountChange(e)} />
      <br/>
      <input type="button" className="send-btn" onClick={e => this.sendTokens(e)} value="Wyślij" />
    </div>
  )
}

function App() {
  return (
    <div className="App">
      <SendTokens className="token-container"/>
    </div>
  );
}

export default App;
```

Pobrany w metodzie loadBalance() stan konta jest odpowiednio skalowany w linijce

```
let balance = response.c[0] / 10000
```

W momencie kiedy użytkownik kliknie w przycisk Wyślij uruchamiana jest metoda SendTokens(), która po sprawdzeniu konta użytkownika wykonuje przy użyciu wcześniej zdefiniowanego ABI operację transfer na kontrakcie.

```
token.transfer(target, amount * decimals, (error, response) => {
  if(error || error !== null) {
    alert(error);
  } else {
    alert('Pomyślnie wysłano ' + amount + ' ' + this.state.symbol);
    this.loadBalance();
  }
})
```

Następnie w pliku App.css dopisujemy poniższy kod

```
.token-container {
  text-align: center;
}

.main-container {
  display: block;
}

.form-label {
  font-size: 1.2rem;
}

.form-balance {
  font-size: 0.8rem;
}

.usr-input {
  min-width: 20vw;
  min-height: 3vh;
  font-size: 0.8rem;
  background-color: white;
  border: solid 1px lightgrey;
  border-radius: 5px;
}

.send-btn {
  margin-top: 0.9em;
  min-width: 5vw;
  min-height: 3vw;
  background-color: white;
  border: solid 1px grey;
  border-radius: 5px;
}

.send-btn:active {
  background-color:lightgrey;
}
```

Efekt końcowy powinien wyglądać następująco

Stan konta

98994889.9499 ThC

Portfel odbiorcy

0xed2bb4Ebb79630608f8E0521d277422067dFb6B8

Liczba tokenów do wysłania

10000000

Wyślij

Po kliknięciu w przycisk Wyślij rozszerzenie Metamask zapyta o zgodę na dokonanie transferu

MetaMask Notification - Opera

CONFIRM TRANSACTION

Przemek

26c46D...37Df

14.740 ETH

3506.10 USD

84E3eE...9dB9

Amount

0.00 ETH

0.00 USD

Cas Limit

54619

UNITS

Cas Price

1

GWEI

Max Transaction Fee

0.000054 ETH

0.01 USD

Max Total

0.000054 ETH

0.01 USD


Data included: 68 bytes



RESET


SUBMIT

REJECT

Zatwierdzenie akcji przez naciśnięcie przycisku Submit powinno przelać tokeny na wybrane przez nas konto.

 Rinkeby
Test Net



Drugie konto

Oxed2bb...

1.999 ETH
475.64 USD

BUY


SEND


SENT


TOKENS

You own 3 tokens

ADD TOKEN

 0 ThC

 0 ThC

 21005003.050 ThC