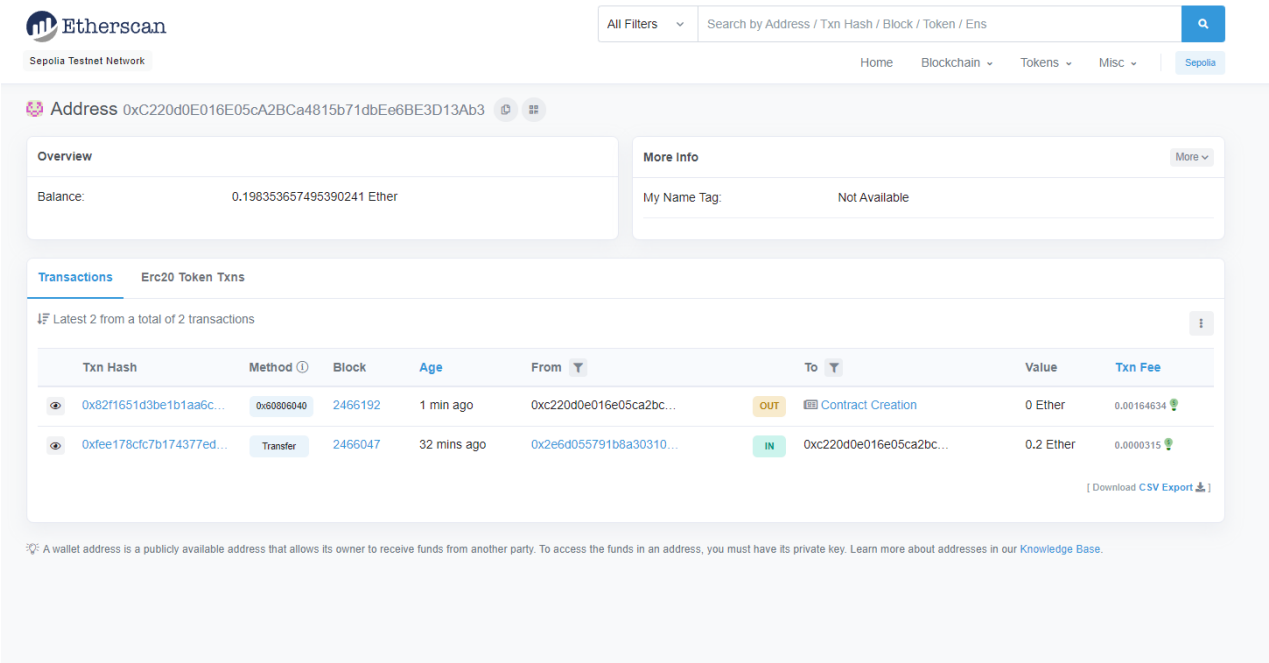
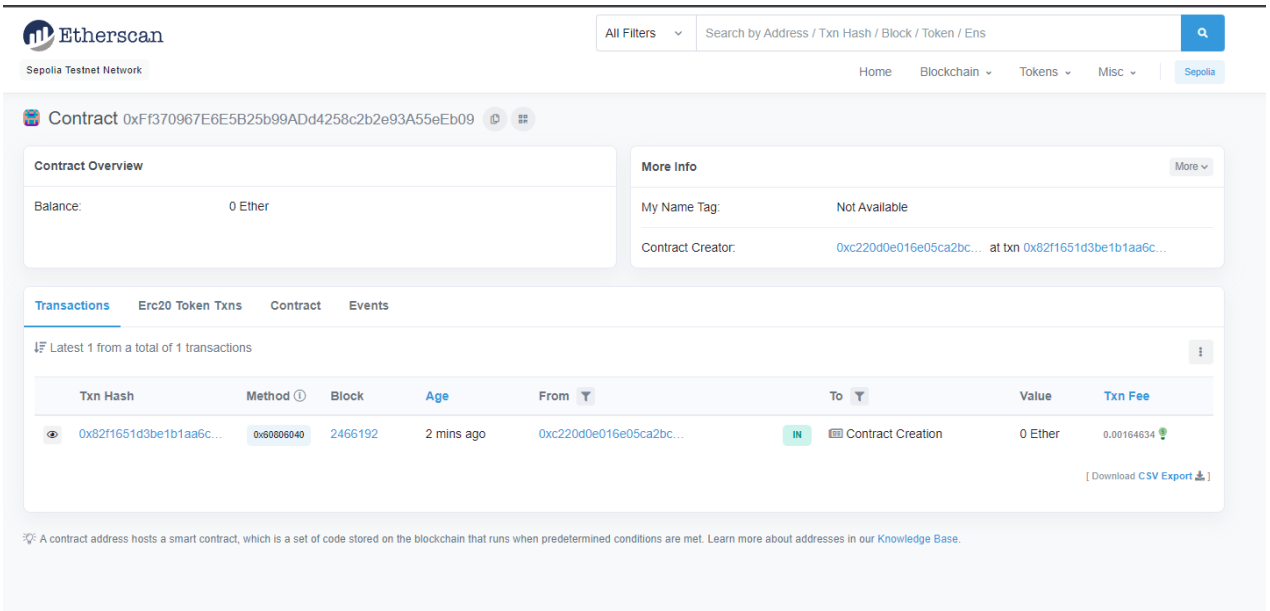
Blockchain Smart Contract Exercise Boch Lukas & Cetinkaya Yusuf

Currency Exchange

Exchange Contract:

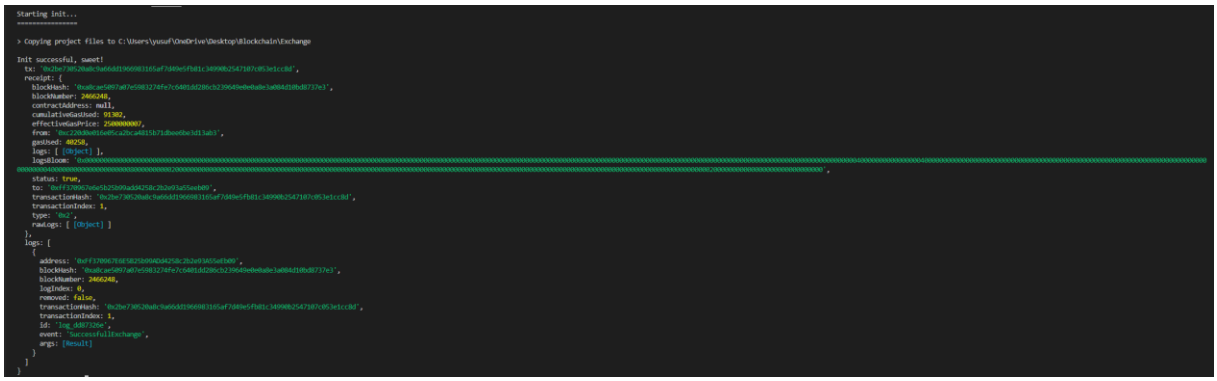
The contract is deployed on the Sepolia test network and is accessible under the following address:
0xFf370967E6E5B25b99ADd4258c2b2e93A55eEb09





Access Contract Exchange

- Create a new project, Create a folder Exchange
- RUN: truffle init
- Copy build folder and the truffle config file
- Replace the dummy in “sepolia section” with your mnemonic and the infura key
- Run: truffle console --network sepolia
- Run: let instance = await Exchange.deployed()



```
Starting init...
================

> Copying project files to C:\Users\yusuf\OneDrive\Desktop\Blockchain\Exchange

Init successful, sweet!
  tx: '0x2be730520a8c9a66dd1966983165af7d49e5fb81c34990b2547107c053e1cc8d',
  receipt: {
    blockHash: '0xa8cae5097a07e5983274fe7c6401dd286cb239649e0e0a8e3a084d10bd8737e3',
    blockNumber: 2466248,
    contractAddress: null,
    cumulativeGasUsed: 91302,
    effectiveGasPrice: 2500000007,
    from: '0xc220d0e016e05ca2bca4815b71dbee6be3d13ab3',
    gasUsed: 40258,
    logs: [ [Object] ],
    logsBloom: '0x[illegible]',
    status: true,
    to: '0xff370967e6e5b25b99add4258c2b2e93a55eeb09',
    transactionHash: '0x2be730520a8c9a66dd1966983165af7d49e5fb81c34990b2547107c053e1cc8d',
    transactionIndex: 1,
    type: '0x2',
    rawLogs: [ [Object] ]
  },
  logs: [
    {
      address: '0xFf370967E6E5B25b99ADd4258c2b2e93A55eEb09',
      blockHash: '0xa8cae5097a07e5983274fe7c6401dd286cb239649e0e0a8e3a084d10bd8737e3',
      blockNumber: 2466248,
      logIndex: 0,
      removed: false,
      transactionHash: '0x2be730520a8c9a66dd1966983165af7d49e5fb81c34990b2547107c053e1cc8d',
      transactionIndex: 1,
      id: 'log_dd87326e',
      event: 'SuccessfullExchange',
      args: [Result]
    }
  ]
}
```

Explanation Contract Exchange

This small implementation of a smart contract is about being able to exchange the currencies Euro, Dollar and Pound. For this, 2% is deducted per transaction as a processing fee and the exchange rate is that of 12.12.2022. First, a struct "Wallet" was created. This contains the data: address owner, uint centEU, uint centUS, uint pennyUK. Then a wallet testWallet is created with the addresse: 0xc0ffee254729296a45a3885639AC7E10F9d54979 and 10000 of each currency. Since solidity does not support decimal numbers, the solution was to calculate with cent/penny and powers of ten in order to be able to perform precise percentage calculations. The address has in this simple example no purpose, but could be used by extension to distinguish between several wallets.

walletInfo()

The first function walletInfo only returns the address of the wallet and the respective amount of currencies. In the following pictures you can also see that the transactions have changed the value of the amounts of the currencies of the wallet.

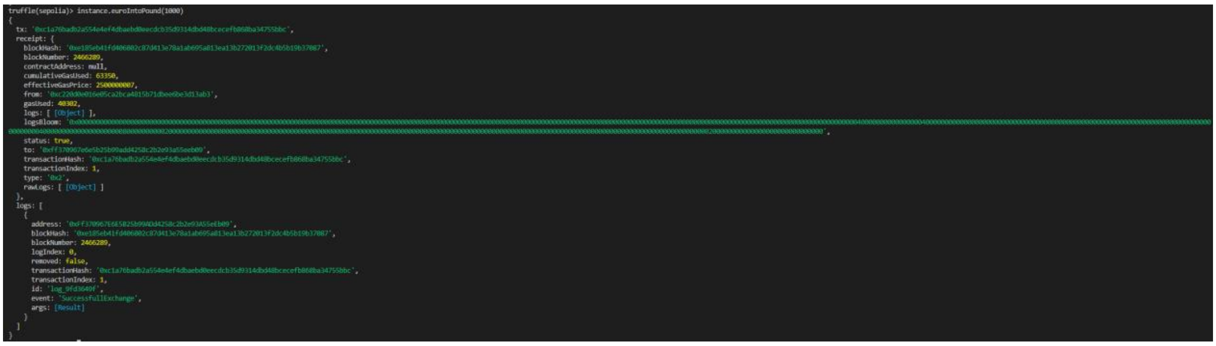
```
truffle(sepolia)> instance.walletInfo()
Result {
  '0': '0xc0ffee254729296a45a3885639AC7E10F9d54979',
  '1': BN {
    negative: 0,
    words: [ 9000, <1 empty item> ],
    length: 1,
    red: null
  },
  '2': BN {
    negative: 0,
    words: [ 11038, <1 empty item> ],
    length: 1,
    red: null
  },
  '3': BN {
    negative: 0,
    words: [ 10000, <1 empty item> ],
    length: 1,
    red: null
  },
  _owner: '0xc0ffee254729296a45a3885639AC7E10F9d54979',
  _centEU: BN {
    negative: 0,
    words: [ 9000, <1 empty item> ],
    length: 1,
    red: null
  },
  _centUS: BN {
    negative: 0,
    words: [ 11038, <1 empty item> ],
    length: 1,
    red: null
  },
  _pennyUK: BN {
    negative: 0,
    words: [ 10000, <1 empty item> ],
    length: 1,
    red: null
  }
}
```

```
truffle(sepolia)> instance.walletInfo()
Result {
  '0': '0xc0ffee254729296a45a3885639AC7E10F9d54979',
  '1': BN {
    negative: 0,
    words: [ 8931, <1 empty item> ],
    length: 1,
    red: null
  },
  '2': BN {
    negative: 0,
    words: [ 9038, <1 empty item> ],
    length: 1,
    red: null
  },
  '3': BN {
    negative: 0,
    words: [ 11635, <1 empty item> ],
    length: 1,
    red: null
  },
  _owner: '0xc0ffee254729296a45a3885639AC7E10F9d54979',
  _centEU: BN {
    negative: 0,
    words: [ 8931, <1 empty item> ],
    length: 1,
    red: null
  },
  _centUS: BN {
    negative: 0,
    words: [ 9038, <1 empty item> ],
    length: 1,
    red: null
  },
  _pennyUK: BN {
    negative: 0,
    words: [ 11635, <1 empty item> ],
    length: 1,
    red: null
  }
}
```

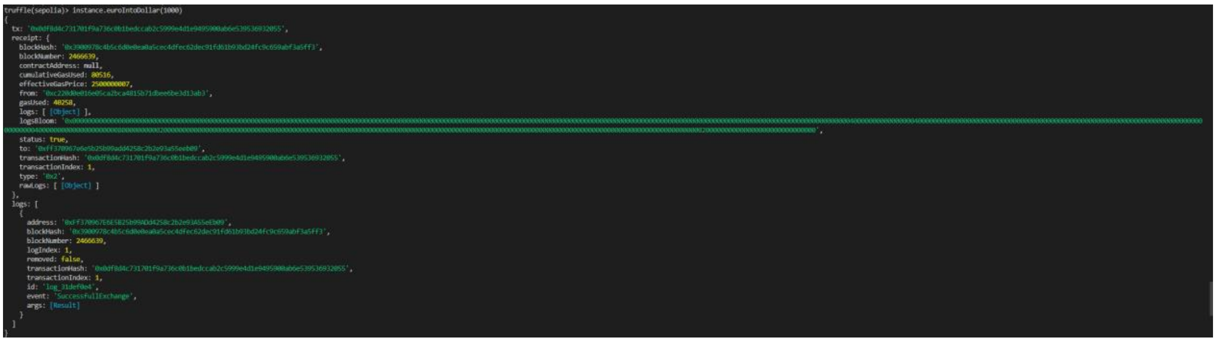
```
truffle(sepolia)> instance.walletInfo()
Result {
  '0': '0xc0ffee254729296a45a3885639AC7E10F9d54979',
  '1': BN {
    negative: 0,
    words: [ 10077, <1 empty item> ],
    length: 1,
    red: null
  },
  '2': BN {
    negative: 0,
    words: [ 10243, <1 empty item> ],
    length: 1,
    red: null
  },
  '3': BN {
    negative: 0,
    words: [ 9635, <1 empty item> ],
    length: 1,
    red: null
  },
  _owner: '0xc0ffee254729296a45a3885639AC7E10F9d54979',
  _centEU: BN {
    negative: 0,
    words: [ 10077, <1 empty item> ],
    length: 1,
    red: null
  },
  _centUS: BN {
    negative: 0,
    words: [ 10243, <1 empty item> ],
    length: 1,
    red: null
  },
  _pennyUK: BN {
    negative: 0,
    words: [ 9635, <1 empty item> ],
    length: 1,
    red: null
  }
}
```

euroIntoDollar(uint _centEU) & euroIntoPound(uint _centEU)

With these functions, you can have euros exchanged for either dollars or pounds. The input parameter is the amount to be exchanged as uint in cent (EU). You don't get a return value except a success message and some log data or an error message if there are not enough of the currency on the wallet. With the function walletInfo() you can see the change afterwards. In addition, an event is fired after each executed transaction. In this implementation, this is still irrelevant, but could be of great use in an extension.



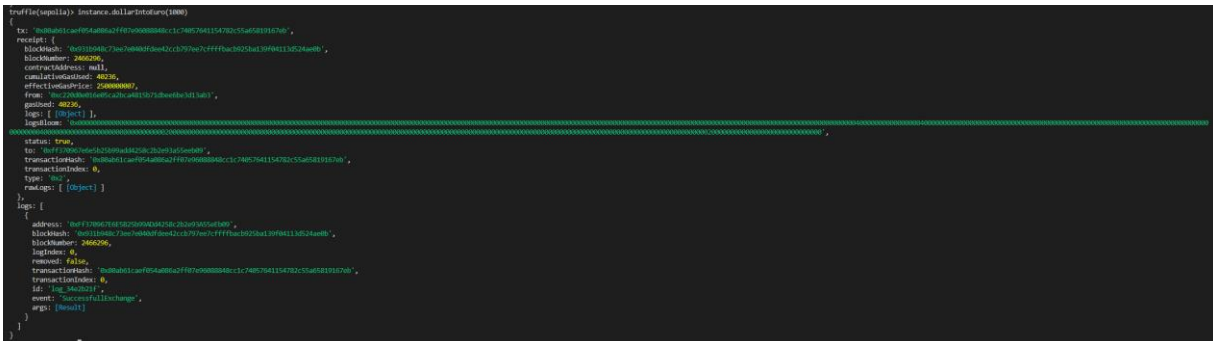
```
truffle(sepolia)> instance.euroIntoPound(1000)
{
  tx: '0xc1a76badb2a554e4ef4dbaebd0eecdcb35d9314dbd48bcecefb868ba34755bbc',
  receipt: {
    blockHash: '0xe185eb41fd406802c87d413e78a1ab695a813ea13b272013f2dc4b5b19b37087',
    blockNumber: 2466289,
    contractAddress: null,
    cumulativeGasUsed: 63350,
    effectiveGasPrice: 2500000007,
    from: '0xc220d0e016e05ca2bca4815b71dbee6be3d13ab3',
    gasUsed: 40382,
    logs: [ [Object] ],
    logsBloom: '0x[illegible]',
    status: true,
    to: '0xff370967e6e5b25b99add4258c2b2e93a55eeb09',
    transactionHash: '0xc1a76badb2a554e4ef4dbaebd0eecdcb35d9314dbd48bcecefb868ba34755bbc',
    transactionIndex: 1,
    type: '0x2',
    rawLogs: [ [Object] ]
  },
  logs: [
    {
      address: '0xFF370967E6E5B25b99ADd4258c2b2e93A55eEb09',
      blockHash: '0xe185eb41fd406802c87d413e78a1ab695a813ea13b272013f2dc4b5b19b37087',
      blockNumber: 2466289,
      logIndex: 0,
      removed: false,
      transactionHash: '0xc1a76badb2a554e4ef4dbaebd0eecdcb35d9314dbd48bcecefb868ba34755bbc',
      transactionIndex: 1,
      id: 'log_9fd3649f',
      event: 'SuccessfullExchange',
      args: [Result]
    }
  ]
}
```



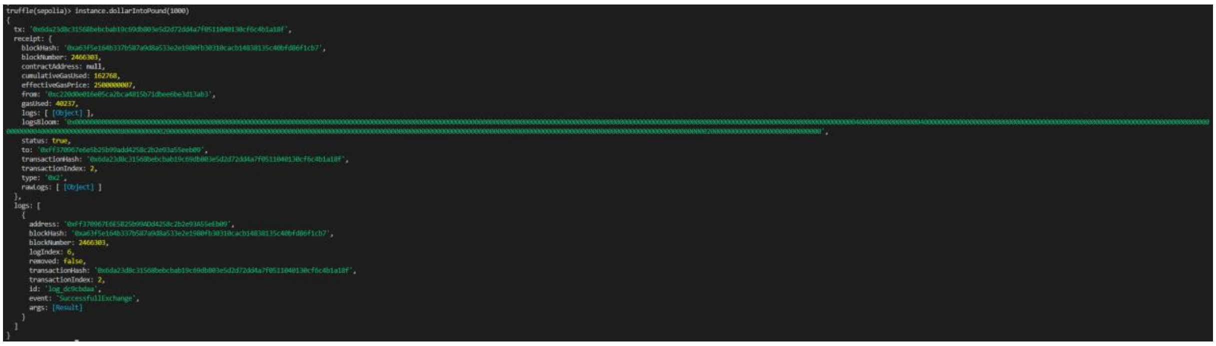
```
truffle(sepolia)> instance.euroIntoDollar(1000)
{
  tx: '0x0df8d4c731701f9a736c0b1bedccab2c5999e4d1e9495900ab6e539536932055',
  receipt: {
    blockHash: '0x3900978c4b5c6d0e0ea0a5cec4dfec62dec91fd61b93bd24fc9c659abf3a5ff3',
    blockNumber: 2466639,
    contractAddress: null,
    cumulativeGasUsed: 80516,
    effectiveGasPrice: 2500000007,
    from: '0xc220d0e016e05ca2bca4815b71dbee6be3d13ab3',
    gasUsed: 40258,
    logs: [ [Object] ],
    logsBloom: '0x[illegible]',
    status: true,
    to: '0xff370967e6e5b25b99add4258c2b2e93a55eeb09',
    transactionHash: '0x0df8d4c731701f9a736c0b1bedccab2c5999e4d1e9495900ab6e539536932055',
    transactionIndex: 1,
    type: '0x2',
    rawLogs: [ [Object] ]
  },
  logs: [
    {
      address: '0xFF370967E6E5B25b99ADd4258c2b2e93A55eEb09',
      blockHash: '0x3900978c4b5c6d0e0ea0a5cec4dfec62dec91fd61b93bd24fc9c659abf3a5ff3',
      blockNumber: 2466639,
      logIndex: 1,
      removed: false,
      transactionHash: '0x0df8d4c731701f9a736c0b1bedccab2c5999e4d1e9495900ab6e539536932055',
      transactionIndex: 1,
      id: 'log_31def0e4',
      event: 'SuccessfullExchange',
      args: [Result]
    }
  ]
}
```

dollarIntoEuro(uint _centUS) & dollarIntoPound(uint _centUS)

With these functions, you can have dollars exchanged for either euros or pounds. The input parameter is the amount to be exchanged as uint in cent (US). You don't get a return value except a success message and some log data or an error message if there are not enough of the currency on the wallet. With the function walletInfo() you can see the change afterwards. In addition, an event is fired after each executed transaction. In this implementation, this is still irrelevant, but could be of great use in an extension.



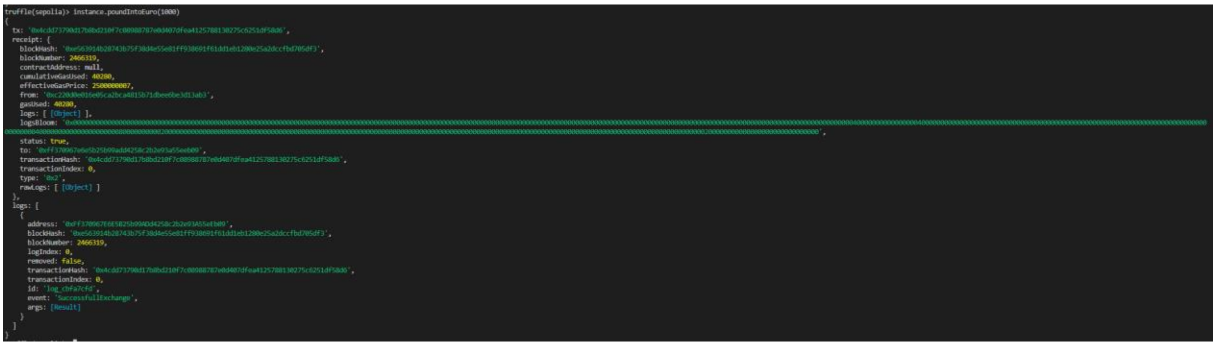
```
truffle(sepolia)> instance.dollarIntoEuro(1000)
{
  tx: '0x88ab61caef054a886a2ff07e96088848cc1c74057641154782c55a65819167eb',
  receipt: {
    blockHash: '0x931b948c73ee7e040dfdee42ccb797ee7cffffbacb925ba139f04113d524ae0b',
    blockNumber: 2466296,
    contractAddress: null,
    cumulativeGasUsed: 40236,
    effectiveGasPrice: 2500000007,
    from: '0xc220d0e016e05ca2bca4815b71dbee6be3d13ab3',
    gasUsed: 40236,
    logs: [ [Object] ],
    logsBloom: '0x[illegible]',
    status: true,
    to: '0xff370967e6e5b25b99add4258c2b2e93a55eeb09',
    transactionHash: '0x88ab61caef054a886a2ff07e96088848cc1c74057641154782c55a65819167eb',
    transactionIndex: 0,
    type: '0x2',
    rawLogs: [ [Object] ]
  },
  logs: [
    {
      address: '0xFF370967E6E5B25b99ADd4258c2b2e93A55eEb09',
      blockHash: '0x931b948c73ee7e040dfdee42ccb797ee7cffffbacb925ba139f04113d524ae0b',
      blockNumber: 2466296,
      logIndex: 0,
      removed: false,
      transactionHash: '0x88ab61caef054a886a2ff07e96088848cc1c74057641154782c55a65819167eb',
      transactionIndex: 0,
      id: 'log_34e2b21f',
      event: 'SuccessfullExchange',
      args: [Result]
    }
  ]
}
```



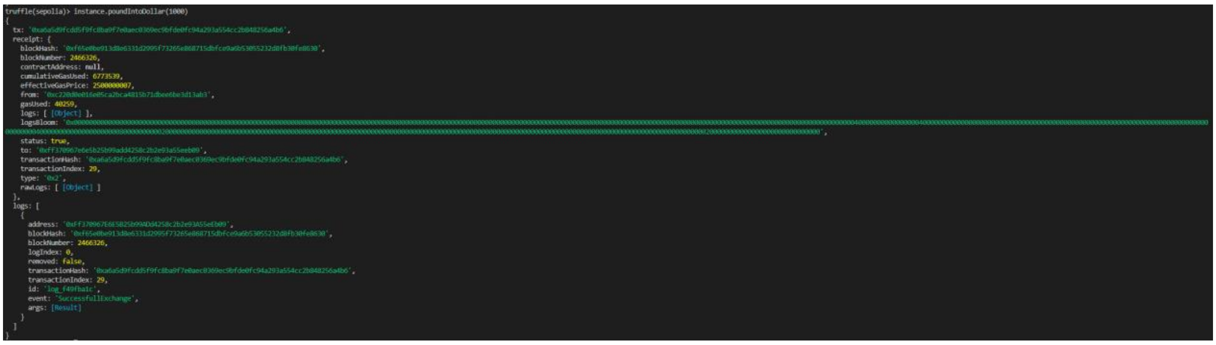
```
truffle(sepolia)> instance.dollarIntoPound(1000)
{
  tx: '0x6da23d8c31568bebcbab19c69db803e5d2d72dd4a7f0511040130cf6c4b1a18f',
  receipt: {
    blockHash: '0xa63f5e164b337b587a9d8a533e2e1980fb30310cacb14838135c40bfd86f1cb7',
    blockNumber: 2466303,
    contractAddress: null,
    cumulativeGasUsed: 162768,
    effectiveGasPrice: 2500000007,
    from: '0xc220d0e016e05ca2bca4815b71dbee6be3d13ab3',
    gasUsed: 40237,
    logs: [ [Object] ],
    logsBloom: '[illegible]',
    status: true,
    to: '0xff370967e6e5b25b90add4258c2b2e93a55eeb09',
    transactionHash: '0x6da23d8c31568bebcbab19c69db803e5d2d72dd4a7f0511040130cf6c4b1a18f',
    transactionIndex: 2,
    type: '0x2',
    rawLogs: [ [Object] ]
  },
  logs: [
    {
      address: '0xFf370967E6E5B25b90ADd4258c2b2e93A55eEb09',
      blockHash: '0xa63f5e164b337b587a9d8a533e2e1980fb30310cacb14838135c40bfd86f1cb7',
      blockNumber: 2466303,
      logIndex: 6,
      removed: false,
      transactionHash: '0x6da23d8c31568bebcbab19c69db803e5d2d72dd4a7f0511040130cf6c4b1a18f',
      transactionIndex: 2,
      id: 'log_dc9cbdaa',
      event: 'SuccessfullExchange',
      args: [Result]
    }
  ]
}
```

poundIntoDollar(uint _pennyUK) & poundIntoEuro(uint _pennyUK)

With these functions, you can have pounds exchanged for either dollars or euros. The input parameter is the amount to be exchanged as uint in penny (UK). You don't get a return value except a success message and some log data or an error message if there are not enough of the currency on the wallet. With the function walletInfo() you can see the change afterwards. In addition, an event is fired after each executed transaction. In this implementation, this is still irrelevant, but could be of great use in an extension.



```
truffle(sepolia)> instance.poundIntoEuro(1000)
{
  tx: '0x4cdd73790d17b8bd210f7c08988787e0d407dfea4125788130275c6251df58d6',
  receipt: {
    blockHash: '0xe563914b28743b75f38d4e55e01ff938691f61dd1eb1200e25a2dccfbd705df3',
    blockNumber: 2466319,
    contractAddress: null,
    cumulativeGasUsed: 40280,
    effectiveGasPrice: 2500000007,
    from: '0xc220d0e016e05ca2bca4815b71dbee6be3d13ab3',
    gasUsed: 40280,
    logs: [ [Object] ],
    logsBloom: '[illegible]',
    status: true,
    to: '0xff370967e6e5b25b90add4258c2b2e93a55eeb09',
    transactionHash: '0x4cdd73790d17b8bd210f7c08988787e0d407dfea4125788130275c6251df58d6',
    transactionIndex: 0,
    type: '0x2',
    rawLogs: [ [Object] ]
  },
  logs: [
    {
      address: '0xFf370967E6E5B25b90ADd4258c2b2e93A55eEb09',
      blockHash: '0xe563914b28743b75f38d4e55e01ff938691f61dd1eb1200e25a2dccfbd705df3',
      blockNumber: 2466319,
      logIndex: 0,
      removed: false,
      transactionHash: '0x4cdd73790d17b8bd210f7c08988787e0d407dfea4125788130275c6251df58d6',
      transactionIndex: 0,
      id: 'log_cbfa7cfd',
      event: 'SuccessfullExchange',
      args: [Result]
    }
  ]
}
```



```
truffle(sepolia)> instance.poundIntoDollar(1000)
{
  tx: '0xa6a5d9fcdd5f9fc8ba9f7e0aec0369ec9bfde0fc94a293a554cc2b848256a4b6',
  receipt: {
    blockHash: '0xf65e0be913d8e6331d2995f73265e868715dbfce9a6b53055232d8fb30fe8630',
    blockNumber: 2466326,
    contractAddress: null,
    cumulativeGasUsed: 6773539,
    effectiveGasPrice: 2500000007,
    from: '0xc220d0e016e05ca2bca4815b71dbee6be3d13ab3',
    gasUsed: 40259,
    logs: [ [Object] ],
    logsBloom: '[illegible]',
    status: true,
    to: '0xff370967e6e5b25b90add4258c2b2e93a55eeb09',
    transactionHash: '0xa6a5d9fcdd5f9fc8ba9f7e0aec0369ec9bfde0fc94a293a554cc2b848256a4b6',
    transactionIndex: 29,
    type: '0x2',
    rawLogs: [ [Object] ]
  },
  logs: [
    {
      address: '0xFf370967E6E5B25b90ADd4258c2b2e93A55eEb09',
      blockHash: '0xf65e0be913d8e6331d2995f73265e868715dbfce9a6b53055232d8fb30fe8630',
      blockNumber: 2466326,
      logIndex: 0,
      removed: false,
      transactionHash: '0xa6a5d9fcdd5f9fc8ba9f7e0aec0369ec9bfde0fc94a293a554cc2b848256a4b6',
      transactionIndex: 29,
      id: 'log_f49fba1c',
      event: 'SuccessfullExchange',
      args: [Result]
    }
  ]
}
```

Transactions After Usage

Contract 0xFf370967E6E5B25b99ADd4258c2b2e93A55eEb09

Contract Overview

Balance: 0 Ether

More Info

My Name Tag: Not Available

Contract Creator: 0xc220d0e016e05ca2bc... at txn 0x82f1651d3be1b1aa6c...

Transactions | Erc20 Token Txns | Contract | Events

Latest 7 from a total of 7 transactions

Txn Hash	Method	Block	Age	From		To	Value	Txn Fee
0xa6a5d9fcdd5f9fc8ba9f...	0x3f2a6c1f	2466326	2 mins ago	0xc220d0e016e05ca2bc...	IN	0xff370967e6e5b25b99a...	0 Ether	0.00010064
0x4cdd73790d17b8bd21...	0xe4b779b9	2466319	4 mins ago	0xc220d0e016e05ca2bc...	IN	0xff370967e6e5b25b99a...	0 Ether	0.0001007
0x6da23d8c31568bebcb...	0x3e779162	2466303	7 mins ago	0xc220d0e016e05ca2bc...	IN	0xff370967e6e5b25b99a...	0 Ether	0.00010059
0x80ab61caef054a086a...	0xcb3c2a18	2466296	9 mins ago	0xc220d0e016e05ca2bc...	IN	0xff370967e6e5b25b99a...	0 Ether	0.00010059
0xc1a76badb2a554e4ef...	0xed9a836e	2466289	10 mins ago	0xc220d0e016e05ca2bc...	IN	0xff370967e6e5b25b99a...	0 Ether	0.00010075
0x2be730520a8c9a66dd...	0xd9a28a91	2466248	19 mins ago	0xc220d0e016e05ca2bc...	IN	0xff370967e6e5b25b99a...	0 Ether	0.00010064
0x82f1651d3be1b1aa6c...	0x60806040	2466192	32 mins ago	0xc220d0e016e05ca2bc...	IN	Contract Creation	0 Ether	0.00164634

[Download CSV Export]

truffle-config.js

For security reasons, we deleted our 12-word phrase and sepolia address from the file after it was done and replaced it with the placeholder "dummy", since the project will be on Github and you can never be too careful with something like that; and for good practice reasons ;)

```
sepolia: {
 provider: function (){
   return new HDWalletProvider("dummy", "https://sepolia.infura.io/v3/dummy")
 },
 network_id: "11155111", // Any network (default: none)
 gas: 4000000
}
//
```