Ethereum
Blockchain Development
Week 20

Introduction

The intention of this lab is to look at functions and how variables are stored. All the exercises are completed in Remix IDE or the VM provided.

Code Completion

Writing code in a new language can be a steep learning curve. The approach here is to provide some code with underscores (_) that you are required to complete. These underscores are there to help you. By completing these exercises you will be improving your skills and knowledge of Solidity.

Each exercise starts on a new page. The red numbers in the right-hand margin are estimated minutes you should spend on each exercise.

1 IERC20, Token

The aim of this exercise is to implement a smart contract that implements a fungible token. The purpose of this exercise is to use the ERC20 standard. Complete the following instructions:

1.1 Setup

10–15

- Open VM and Remix-ide desktop and create a new workspace, **W20**.
- Further information is available from the Ethereum Improvement Proposals EIP ERC20 website.
- Upload the file to the contracts folder in the workspace.
- This code has been adapted from the recommended text, [**?**] in Chapter 5, starting on page 196.
- When implementing and testing skip to chapter 6 in [**?**] on page 234 and follow the instructions.
- Download `MyERC20Token.sol` and upload to the contract folder.
- Take some time to read the function declarations.
- Compile and Deploy the `MyERC20Token` contract on the first address, with the following parameters:
  1. Name: `MiddlesexCoin`
  2. Symbol: `MDXC`
  3. Initial Account: `ENTER FIRST ADDRESS HERE`
  4. Initial Balance: `1000000000000000000`

1.2 Using the contract

20–25

Complete the following tasks to use and understand the contract.

1. Approve Second and Third addresses with an allowance of 100
2. Add the Second and Third addresses to the allow list
3. Confirm each address by looking at the results of their balance (0), allowance (100), and allow list (true).
4. Transfer from the owner 10000 to the First and Second addresses above.
5. Confirm this transaction is complete by inspecting the balanceOf the, first, second and third addresses.
6. Transfer 25 MDXC between the second and third address.
7. Check and confirm these balances have been updated.
8. Pause the owner account (first address).
9. Repeat the above steps for the Fourth address, what is the problem?
10. Unpause the account and repeat the steps above for the Fourth address.

2 Membership

Create a new file named, `membership.sol`, in the contract folder.

This simple contract renews expired memberships. If the membership has expired the member is required to pay the `MEMBERSHIP_FEE` of 1 Ether. If the membership has not expired the member is not required to pay anything. Finally, there is a collect function, that the owner can collect the subscriptions from the contract address.

2.1 Set up

15–20

Complete the following:

1. Download the file membership-incomplete.sol.
2. Complete the code and compile.
3. Please note: the addresses supplied in the `init` function may not match your 2^{nd} – 4^{th} addresses, replace these addresses.

2.2 Deployment

15–20

Complete the following:

1. Deploy this contract at the 1^{st} address, this will be the owner of the contract.
2. Run the `init` function to create three accounts.
3. Check the membership and expiry dates of each membership.
4. Pay the membership:
   (a) Select Address 2 and pay the membership
   (b) Select Address 3 and pay the membership
   (c) Select Address 4 and pay the membership
5. Which accounts accepted the payment?
6. Check all the accounts have been updated.
7. Collect 1.5 Ether from the contract Address.
8. Can we still pay the membership?

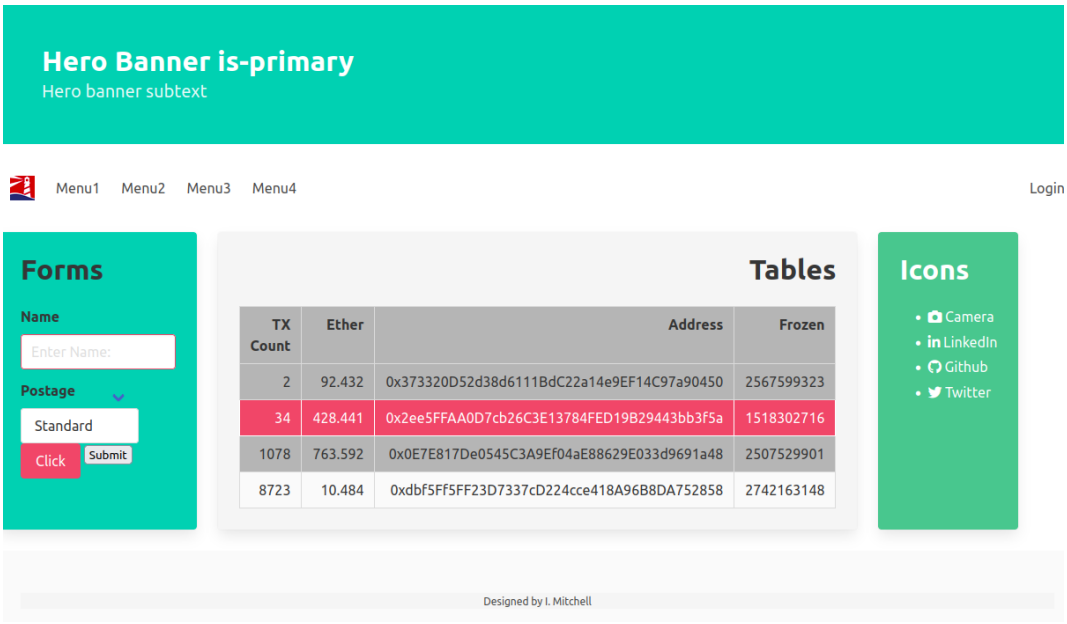

Figure 3.1: User Interface for you to design using Bulma

3 Bulma

20–25

Using Bulma create a website using HTML that resembles the interface displayed in fig. 3.1

References

[1] X. Wu, Z. Zhihong, and D. Song. *Learn Ethereum*. Packt, 1st edition, 2019.