Dynamo Smart Contract Code Explanation

Status: Final Business Plan
Professor: Siddhartha R Dalal
Author:
Hongyun Wei, Elias George Chahine, Zhanpeng Xu, Romauli Butarbutar, Jonathan Foy

Part I: Smart Contracts and Features

Dynamo marketplace enables the sell and purchase feature of gaming item NFT using smart contracts. It illustrates two fundamental functionalities that Dynamo provides: 1) Mint the NFT and put the NFT for sale, 2) Purchase of the NFT.

<u>Contract A: Dynamo NFT.sol</u>

Our ultimate vision is to have the Dynamo NFT contract mint in-game items into an NFT, and list these NFTs in a marketplace for sale. As of now, features that are included are:

- Generate new ID when minting an NFT
- Set token URI (a unique identifier of the in-game item NFT)
- NFT (contains information such as the id, price, sale availability)
- Get the price
- Custodian: We only want to transfer the NFT from one gamer to another when the gamer actually purchases the NFT. To conduct in a secure way, we want to approve a custodian, a person in charge of transferring ownership on a sale, to transact on behalf of the buyer and seller.
- List Token for sale

This part of the code has three written functions and five functions that come from the ERC721 standard which is used to represent ownership of non-fungible tokens. Following the flow chart (please refer to Part V), Dynamo will first mint the gaming items into the NFT and list it on sale using the 'mintNft' function. The rest of the code includes the major functionalities such as getting the price of a particular NFT, transferring the NFT, re-writing the sales status of the NFT, and checking ownership of the NFT etc. The ownership of the first contract is Dynamo, acting on behalf of the seller.

<u>Contract B: Dynamo Transaction.sol</u>

This contract transfers the NFT - the buyer will receive the NFT and the seller will receive the money (this function is called by the custodian). The ownership of the second contract is the buyer. A main requirement for this smart contract to execute is that the money being sent is greater than the sales price of the NFT. Features included:

- Buyer can purchase the NFT
- Send funds through Metamask
- Ownership of an NFT can be transferred from seller to buyer

Part II: Cost/Benefit Analysis: Reason for Having Two Contracts

Cost:

- Sending information through two contracts increases the chances of a security breach

Benefit:

- Two contracts allow us to add a custodian, introducing the opportunity to include a third-party working on behalf of Dynamo. This brings a number of benefits:
    - The custodian can enforce constraints, most notably the constraint function *require(amountPaid >= salePrice)*. This function ensures that the amount paid is greater than the item price. If the condition is not met, the transaction will not be triggered.
    - Process can be automated to include verifying the transaction, transfering the NFT, transfering the ETH, and marking the NFT sold, all at once
- Two contracts decrease the gas fee, compared to one extremely long contract.

Looking at the conclusions from the cost/benefit analysis, we ultimately decided to design two smart contracts for this project.

Part III: Prerequisites

1. Compile the 'Dynamo NFT.sol' and 'Dynamo Transaction.sol'
    a. In this report we will refer to 'Dynamo NFT.sol' as "the first contract" for simplicity.
    b. In this report we will refer to 'Dynamo Transaction.sol' as "the second contract" for simplicity.
2. Set 'ENVIRONMENT' to 'JavaScript VM (London)'
3. The first address we will be using is: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
    a. We will use it to deploy the smart contract
4. The second address we will be using is: 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2
    a. We will use it to pretend to be the buyer
5. Select and deploy two solidity files under 'CONTRACT'

Part IV: Smart Contract Workflow

1. Enter the address of the first contract into the 'setDynamoNftAddress' function in the second contract. Doing this allows us to reference the first contract in the second contract so that we can later call the price of specific items minted in the first contract, and conduct the transaction verification as mentioned in the '**Benefits**' section.
   - **Example:**
     - Enter the address of the first contract into the *'setDynamoNftAddress'* function in the second contract
   - **Output:** 'true Transaction mined and execution succeed'
   - Make sure to use the correct address of the first contract. At the moment, the transaction will run but we will face problems later down the line as we move through the different features of the smart contracts.

2. Use the 'mintNft' function in the first contract to mint an NFT for a specific item
   - *'tokenURI'* identifies the identity of the item
   - *'price'* represents the sale price that seller set
   - *'custodian'* is required to enter the address of the second contract to enable the auto process demonstrated in the '**Benefits**' section
     - **Example:**

'tokenURI'	- enter 'NFT0'
'Price'	- enter '100'
'custodian'	- enter 'the address of second contract'

     - **Output:** 'true Transaction mined and execution succeed'
     - Make sure to use the correct address of the second contract. At the moment, the transaction will run but we will face problems later down the line as we move through the different features of the smart contracts.

3. Alternative functions:
   - *'_listed'*: shows the information of a specific NFT
     - **Example:**

'_listed'	- enter '0' (the NFT is in the array at index 0)

     - **Output:**

       ```
       Id        0
       price     100
       ForSale   true
       ```

     - Failing to enter the existing NFT id will cause the transaction to be reverted

   - *'getSalePrice'*: retrieves sale price of an NFT
     - **Example:**

        ‘getSalePrice’   - enter ‘0’ (get the sale price of NFT indexing 0)

    - **Output:** 100 (the sale price we enter before)
    - Failing to enter the existing NFT id will cause the transaction to be reverted.
- *‘ownerOf’*: shows the address of current owner of a specific NFT
    - **Example:**

        ‘ownerOf’   - enter ‘0’

    - **Output:** ‘the address of the first account’
    - Failing to enter the existing NFT id will cause the transaction to be reverted

4. To buy the NFT, switch to the second account and enter the token id (the id will follow the sequence of NFT creation) after the ‘buyNft’ function in the second contract.
    - *‘VALUE’* column represents the amount the buyer wants to pay - it must be greater than the sale price of the specific NFT, otherwise the transaction will be reverted
        - **Example:**

            ‘VALUE’   - enter ‘120’

        - **Output:** N/A
        - Failing to enter the amount greater than the sale price id cause the later transaction to be reverted
    - The processes mentioned in the ‘**Benefits**’ section will be triggered
    - Once the transaction is done, the owner of the NFT will be the second account, and the status of that NFT will be marked ‘ForSale – false’

        - *‘_listed’* : shows the information of a specific NFT
            - **Demo example:**

                ‘_listed’   - enter ‘0’

            - **Output:**

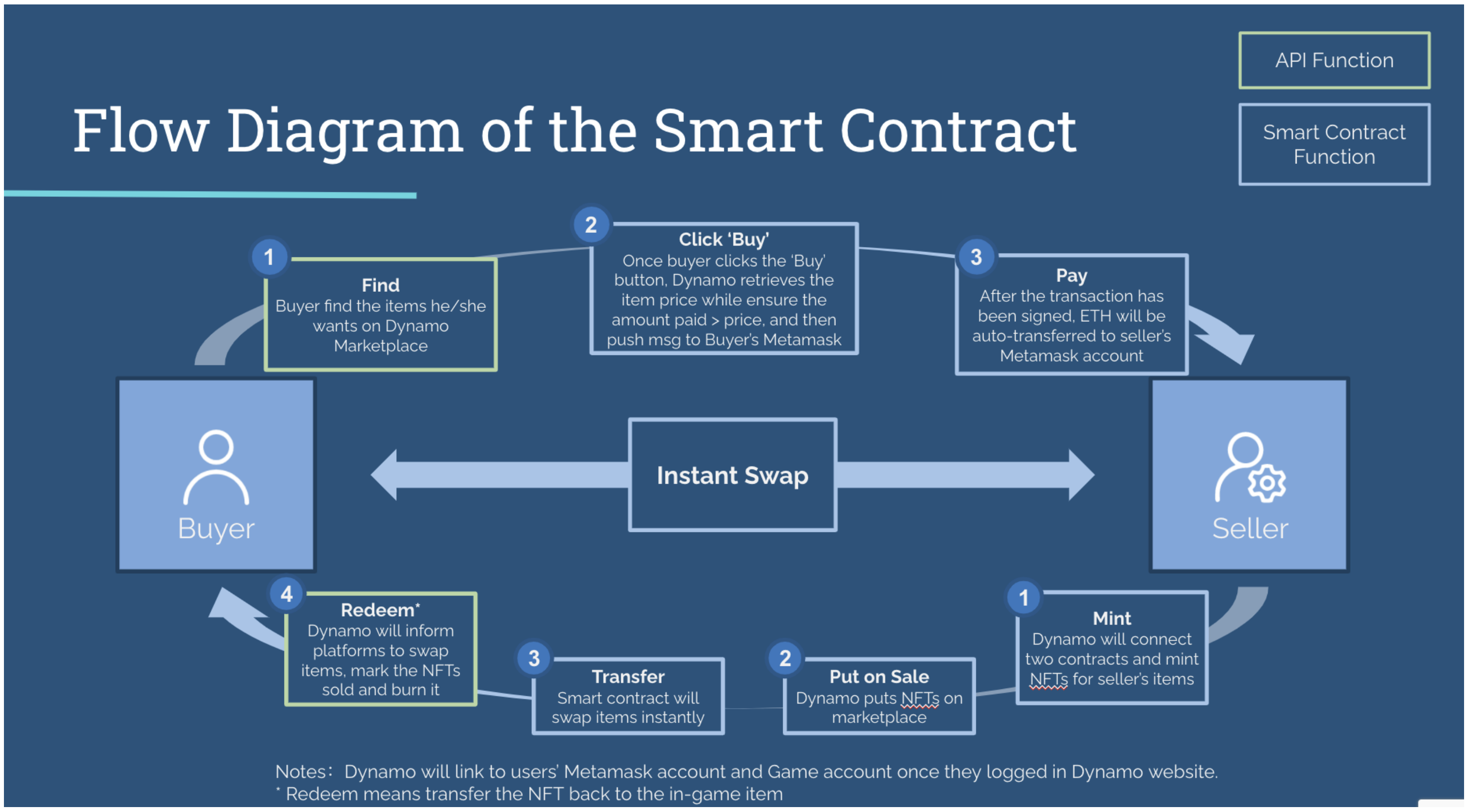
Id	0
price	100
ForSale	false

            - Failing to enter the existing NFT id will cause the transaction to be reverted
        - *‘ownerOf’*: shows the address of current owner of a specific NFT
            - **Example:**

                ‘ownerOf’ - enter ‘0’

            - **Output:** ‘the address of the second account’
            - Failing to enter the existing NFT id will cause the transaction to be reverted

Part V: Flow Diagram


API Function
Smart Contract Function
Flow Diagram of the Smart Contract
1 Find
Buyer find the items he/she wants on Dynamo Marketplace
2 Click 'Buy'
Once buyer clicks the 'Buy' button, Dynamo retrieves the item price while ensure the amount paid > price, and then push msg to Buyer's Metamask
3 Pay
After the transaction has been signed, ETH will be auto-transferred to seller's Metamask account
Buyer
Instant Swap
Seller
4 Redeem*
Dynamo will inform platforms to swap items, mark the NFTs sold and burn it
3 Transfer
Smart contract will swap items instantly
2 Put on Sale
Dynamo puts NFTs on marketplace
1 Mint
Dynamo will connect two contracts and mint NFTs for seller's items
Notes: Dynamo will link to users' Metamask account and Game account once they logged in Dynamo website.
* Redeem means transfer the NFT back to the in-game item