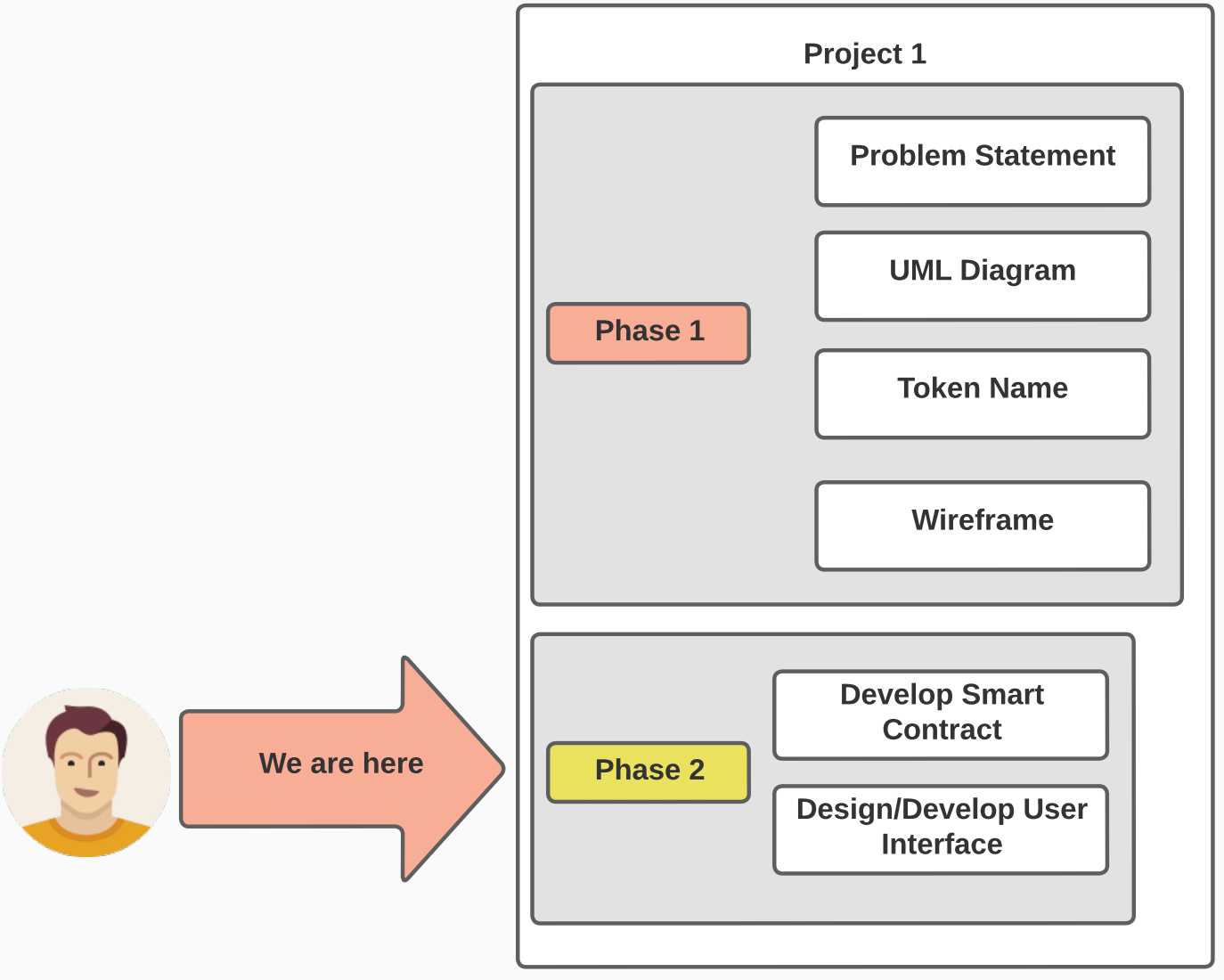
Today's Discussion

Topics covered today

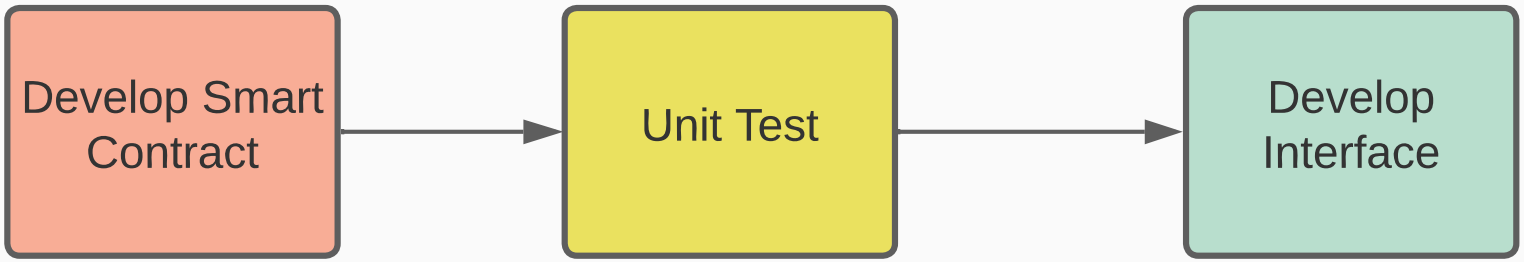
- Unit testing
- Connecting JavaScript to web3
- Work process flows through GitHub
- Tokens Introduction

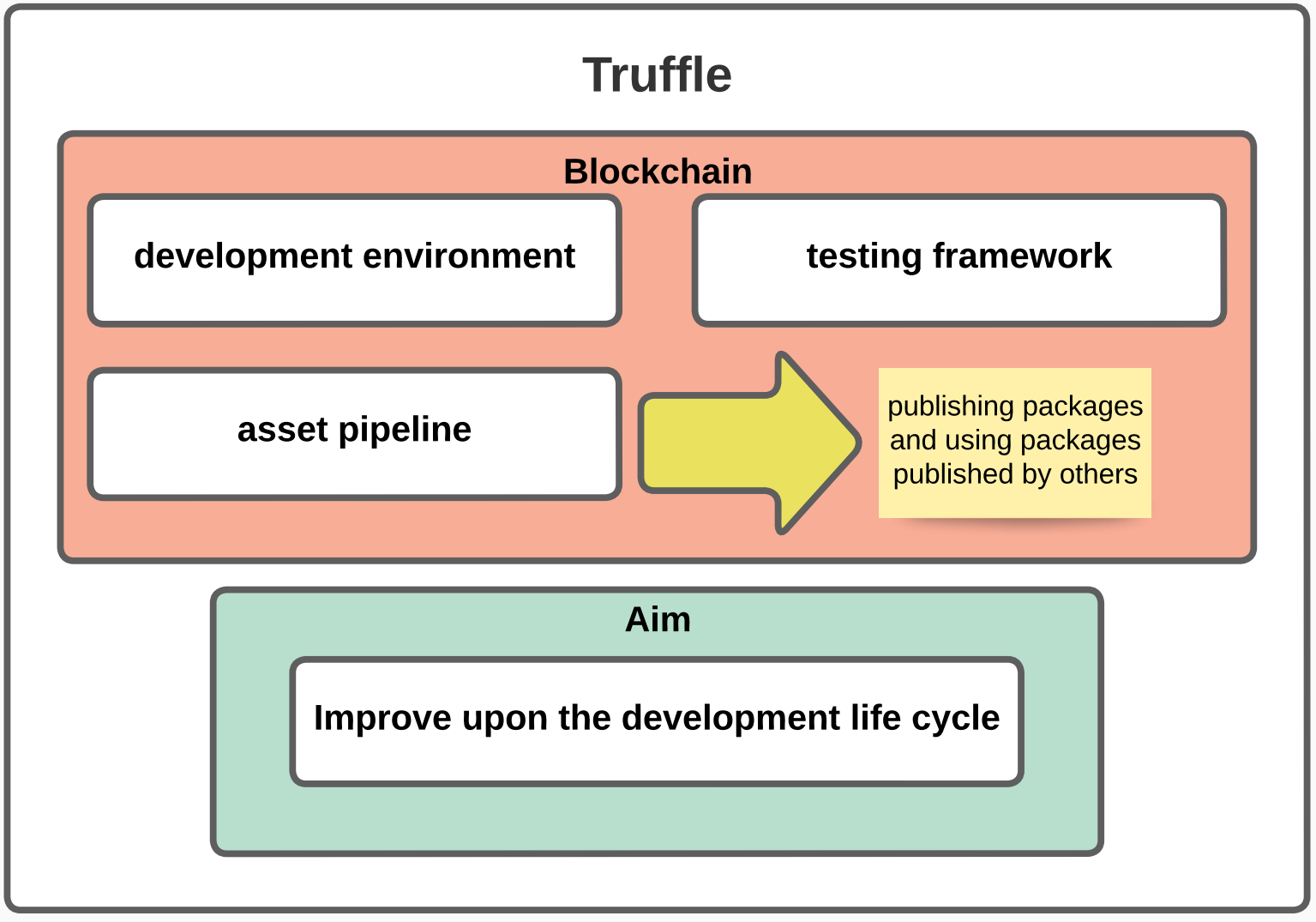
Why?

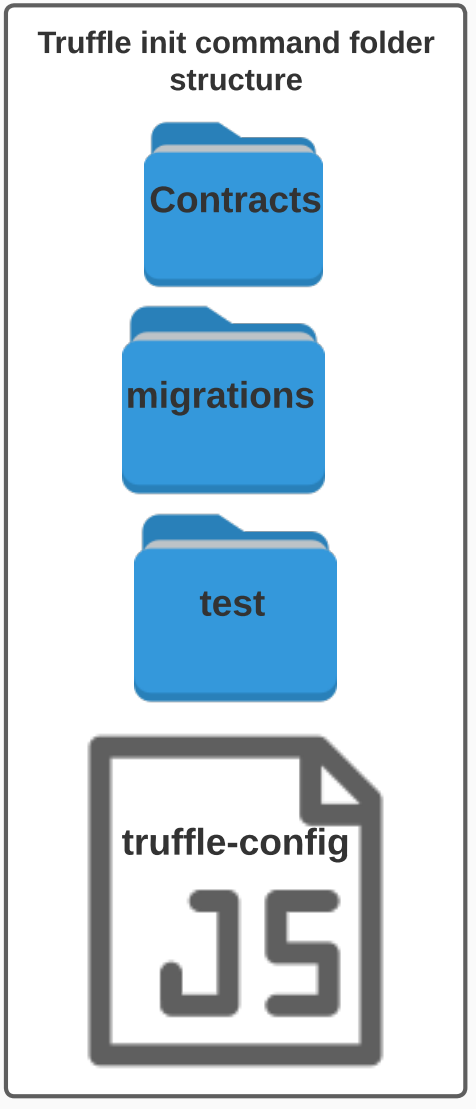
- Ensures that smart contracts don’t have bugs
- Ensuring that your smart contracts can be used by JavaScript to add data to the blockchain.
- Faster and more robust development

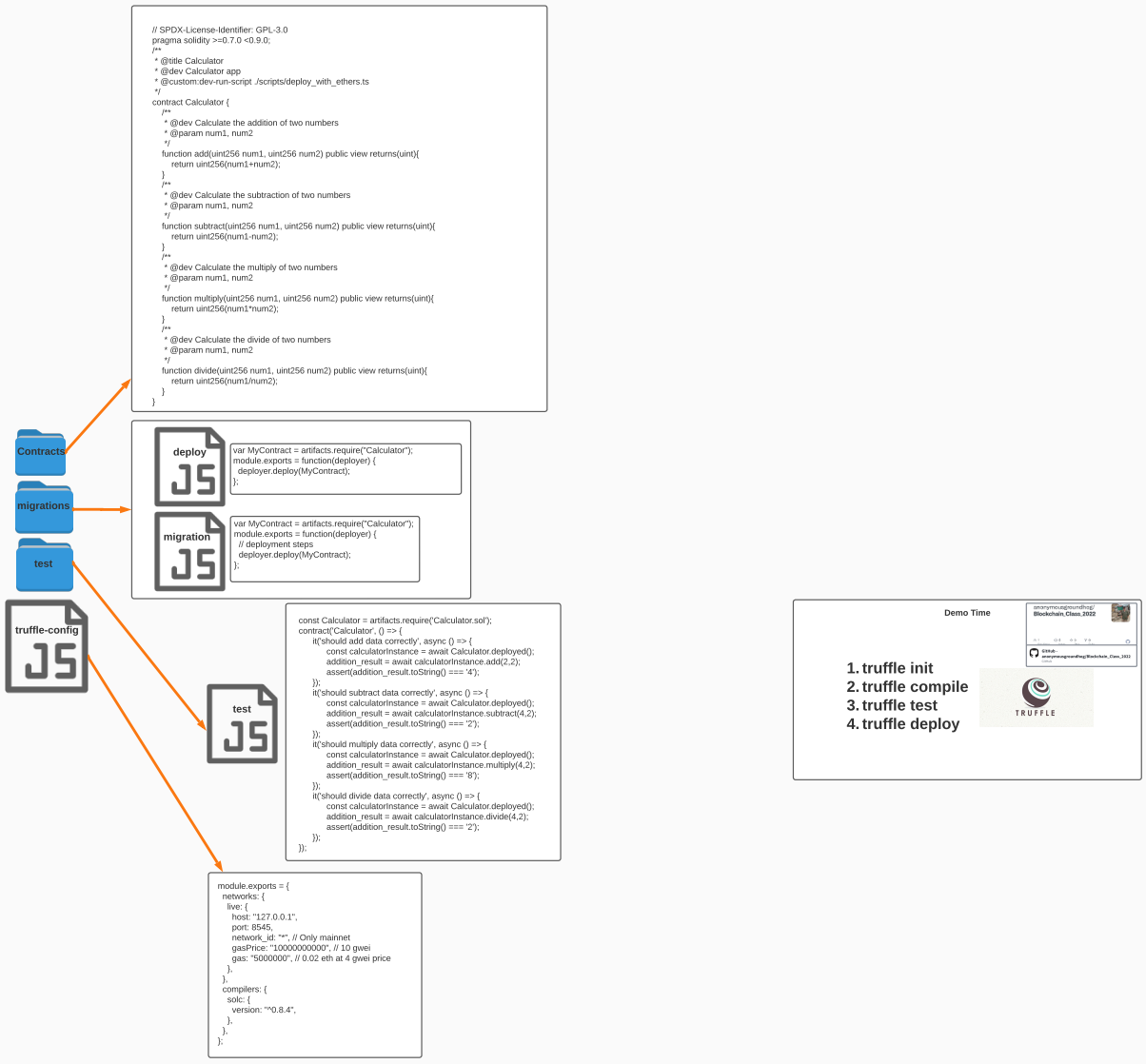

Project 1
Problem Statement
UML Diagram
Phase 1
Token Name
Wireframe
We are here
Develop Smart Contract
Phase 2
Design/Develop User Interface


Main Idea


Develop Smart Contract
Unit Test
Develop Interface


Truffle
Blockchain
development environment
testing framework
asset pipeline
publishing packages
and using packages
published by others
Aim
Improve upon the development life cycle


Truffle init command folder structure
Contracts
migrations
test
truffle-config
JS


// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.9.0;
/**
* @title Calculator
* @dev Calculator app
* @custom:dev-run-script ./scripts/deploy_with_ethers.ts
*/
contract Calculator {
/**
* @dev Calculate the addition of two numbers
* @param num1, num2
*/
function add(uint256 num1, uint256 num2) public view returns(uint){
return uint256(num1+num2);
}
/**
* @dev Calculate the subtraction of two numbers
* @param num1, num2
*/
function subtract(uint256 num1, uint256 num2) public view returns(uint){
return uint256(num1-num2);
}
/**
* @dev Calculate the multiply of two numbers
* @param num1, num2
*/
function multiply(uint256 num1, uint256 num2) public view returns(uint){
return uint256(num1*num2);
}
/**
* @dev Calculate the divide of two numbers
* @param num1, num2
*/
function divide(uint256 num1, uint256 num2) public view returns(uint){
return uint256(num1/num2);
}
}
Contracts
migrations
test
truffle-config
deploy
var MyContract = artifacts.require("Calculator");
module.exports = function(deployer) {
deployer.deploy(MyContract);
};
migration
var MyContract = artifacts.require("Calculator");
module.exports = function(deployer) {
// deployment steps
deployer.deploy(MyContract);
};
test
const Calculator = artifacts.require('Calculator.sol');
contract('Calculator', () => {
it('should add data correctly', async () => {
const calculatorInstance = await Calculator.deployed();
addition_result = await calculatorInstance.add(2,2);
assert(addition_result.toString() === '4');
});
it('should subtract data correctly', async () => {
const calculatorInstance = await Calculator.deployed();
addition_result = await calculatorInstance.subtract(4,2);
assert(addition_result.toString() === '2');
});
it('should multiply data correctly', async () => {
const calculatorInstance = await Calculator.deployed();
addition_result = await calculatorInstance.multiply(4,2);
assert(addition_result.toString() === '8');
});
it('should divide data correctly', async () => {
const calculatorInstance = await Calculator.deployed();
addition_result = await calculatorInstance.divide(4,2);
assert(addition_result.toString() === '2');
});
});
module.exports = {
networks: {
live: {
host: "127.0.0.1",
port: 8545,
network_id: "*", // Only mainnet
gasPrice: "10000000000", // 10 gwei
gas: "5000000", // 0.02 eth at 4 gwei price
},
},
compilers: {
solc: {
version: "^0.8.4",
},
},
};

Demo Time
anonymous/groundhog/
Blockchain_Class_2022
github-
anonymousgroundhog/Blockchain_Class_2022
1. truffle init
2. truffle compile
3. truffle test
4. truffle deploy
TRUFFLE

Demo Time

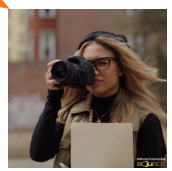
1. truffle init
2. truffle compile
3. truffle test
4. truffle deploy

Tokenization Smart Contract (chapter 9)

Tokenize → representing the asset with a digital unit that can be transferred, traded, exchanged, regulated, and managed like fiat currency or cryptocurrency

Examples of assets

Files

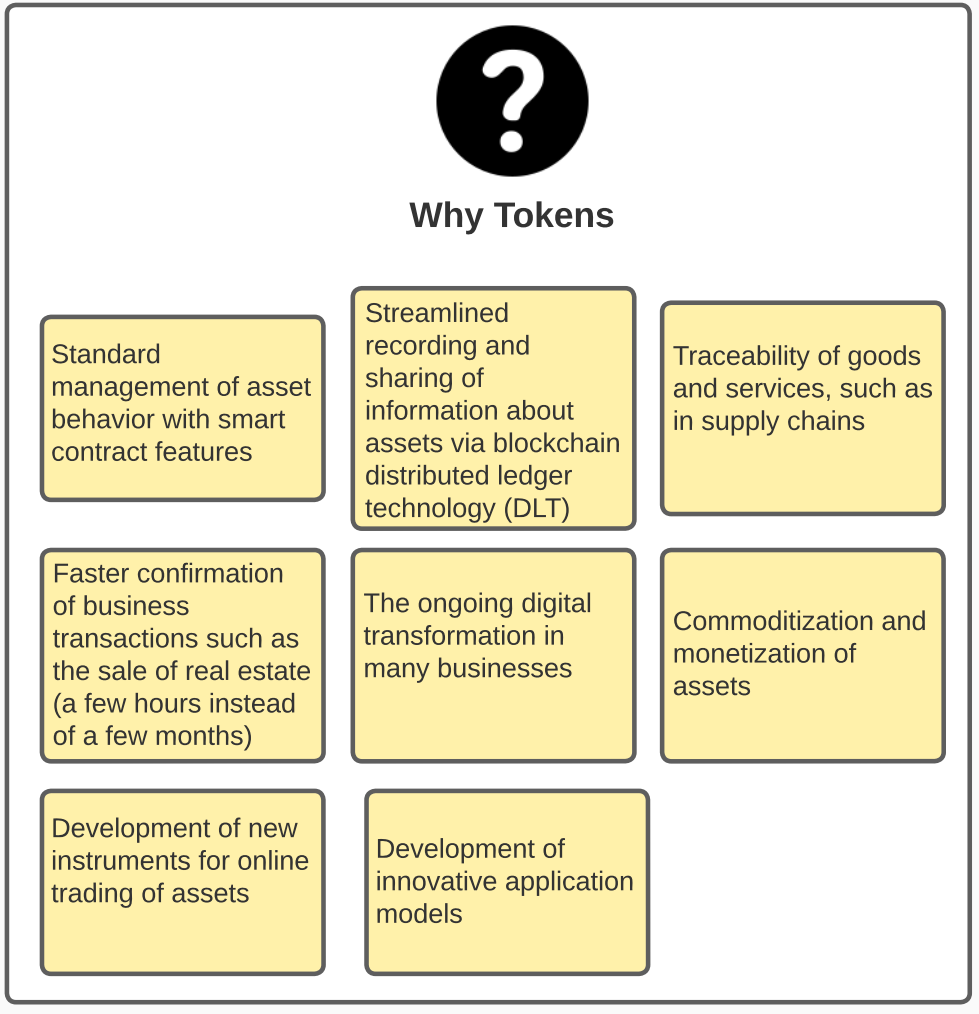


Photos

Real Estate

Tokens

Tokens need to comply with standards to facilitate seamless interaction among different token applications


Why Tokens
Standard management of asset behavior with smart contract features
Streamlined recording and sharing of information about assets via blockchain distributed ledger technology (DLT)
Traceability of goods and services, such as in supply chains
Faster confirmation of business transactions such as the sale of real estate (a few hours instead of a few months)
The ongoing digital transformation in many businesses
Commoditization and monetization of assets
Development of new instruments for online trading of assets
Development of innovative application models

Sign in: https://forms.gle/nG3XMy5MNZyf6FQh9

ERC20
Ethereum Request for Comments

Developed by Fabian Vogelsteller

ERC721
Ethereum Request for Comments

a template or format that other developers agree to follow

has launched a thousand non-fungible tokens

What are NFT's?

Fungible means interchangeable and replaceable

One NFT cannot replace another

See

Beginner's Guide to NFTs: What Are Non-Fungible Tokens? - Decrypt
Decrypt

easy to create tokens that were verifiably, cryptographically unique

could be linked to unique content

Artwork

Music

Sneakers

More Details

Developed by Dieter Shirley

developed

CryptoKitties | Collect and breed digital cats!
CryptoKitties

What is ERC-721? The Ethereum NFT Token Standard - Decrypt
Decrypt

How to create and deploy an ERC-721 (NFT)
Quicknode

uses

nibbstack/**erc721**

The reference implementation of the ERC-721 non-fungible token standard.

11 Contributors · 17 Used by · 999 Stars · 380 Forks

GitHub - nibbstack/erc721: The reference implementation of the ERC-721 non-fungible token standard.
GitHub

Smart Contract: token.sol

```
//SPDX-License-Identifier: UNLICENSED
// SPDX-License-Identifier: MIT
pragma solidity 0.8.0;

import "https://github.com/0xcert/ethereum-erc721/src/contracts/tokens/nf-token-metadata.sol";
import "https://github.com/0xcert/ethereum-erc721/src/contracts/ownership/ownable.sol";

contract newNFT is NFTokenMetadata, Ownable {

  constructor() {
    nftName = "Synth NFT";
    nftSymbol = "SYN";
  }

  function mint(address _to, uint256 _tokenId, string calldata _uri) external onlyOwner {
    super._mint(_to, _tokenId);
    super._setTokenUri(_tokenId, _uri);
  }
}
```