



CPSC 3640/5400: Decentralized Payments, Contracts, and Finance for Humans and AI

Lecture 8: Tokens and Tokenization

Instructor: Prof. Fan Zhang

Yale University

Fall 2026

News in crypto



The cryptographic world computer

2026 Sep 27

[See all posts](#)

https://vitalik.eth.limo/general/2026/09/27/the_cryptographic_world_computer.html

- Vitalik's new article on Ethereum by 2030
- He talks about how Ethereum will be fundamentally different from Bitcoin, or a blockchain in general.
- “a hybrid architecture that combines together blockchains and modern cryptography, to enable much more powerful properties.”

Section II: DeFi as the new fintech frontier

- Organized into a few categories
- **DeFi Primitives:** What they are, how they are implemented
 - Apps: Tokens, Tokenization, Decentralized Exchanges, Oracles, (re-)staking
 - Infra for fast transactions
- **DeFi advantages:** things not possible in TradFi
 - Privacy, Interoperability, censorship resistance,
- **DeFi risks:** new problems unique to DeFi & mitigation
 - order manipulation, human factors, bugs

Tokens

- Physical tokens. Representation of assets, currency, or access rights.



Game tokens



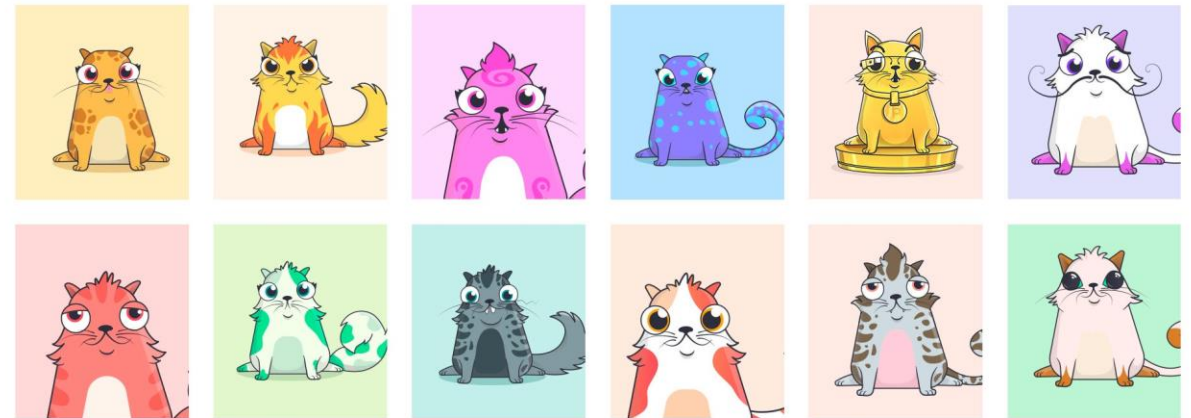
Laundry tokens



NYC subway tokens
(replaced by MetroCard in 2003)

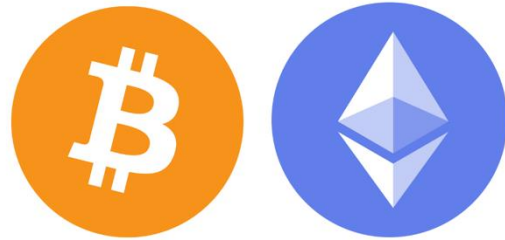
Blockchain-managed tokens

- **Digital representation of assets, currency, or access rights**
- Many examples:
- Digital currencies (e.g., USDC, USDT)
- Assets (e.g., gold, real estate, gaming items)
- Equity (e.g., DAO members hold equity tokens)
- Voting rights
- Collectibles (e.g., CryptoKitties)
- Identities (e.g., domain names)
- ...



Fungibility

- “Fungibility is the property of a good or a commodity whose individual units are essentially interchangeable.” Wikipedia

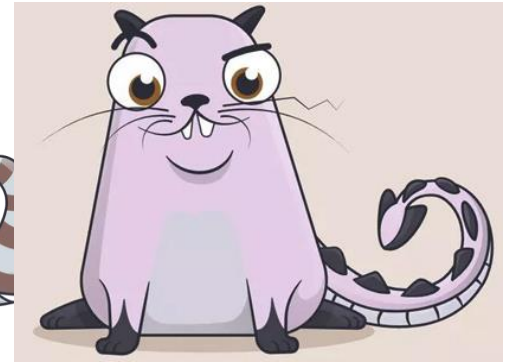


Bank notes are essentially* fungible.



237711

@ 0.002 ETH



Dragon (valued at 600 ETH)

Collectibles are non-fungible.

Decentralized Implementation of Tokens

- Tokens exist before Ethereum
 - NameCoin uses Bitcoin fork to track domain names (a form of NFTs)
- Ethereum community pioneered *token standards*
 - Fungible tokens: ERC 20
 - Non-fungible tokens (NFT): ERC 721
- *Why standards*: enable interoperability

ERC20 tokens

- “20” is the original GitHub issue number for this Ethereum Request for Comment (ERC)
- ERC20 is a **standard**
 - defines a common interface for contracts implementing a token, such that any compatible token can be accessed and used in the same way.
- 1000+ ERC-20 tokens have been created
 - E.g., <https://etherscan.io/tokens>

A total of 2,303,351 Token Contracts found
Showing 3562 Tokens with OK or Neutral Reputation

#	Token
1	 Tether USD (USDT)
2	 BNB (BNB)
3	 USDC (USDC)
4	 Wrapped ZEC (WZEC)
5	 stETH (stETH)
6	 Wrapped liquid staked ... (wstETH)
7	 ChainLink Token (LINK)
8	 Wrapped Binance Beac... (wBETH)
9	 USDS Stablecoin (USDS)
10	 Multichain BTC (multiBTC)

**Just enough Solidity to
understand ERC-20 and
ERC-721**

“Hello World”

```
contract HelloWorld {  
    string public greet = "Hello World!";  
}
```

A smart contract with a single *state variable* and no functions.

[Use Remix to deploy a counter smart contract & checkout state changes.]

Solidity

```
contract HelloWorld {  
    string public greet = "Hello World!";  
}
```

- We will highlight some important syntax
- Each contract can define **State Variables, Functions, Function Modifiers, Events, Errors**
- Contracts can *inherit* from other contracts.

State Variables

- State variables are stored in the storage.
- Each state variable has
 - a name
 - a type
 - a visibility modifier
- (Sidebar: It can be expensive to store many states. “gas optimization” is an important practical skill.)

State Variables: types

- Common types include:
- `bool`, `uint8`, `uint16`, ... `uint256`, `int8`, ... `int256`
- `address` (`bytes20`)
- `string`
- `byte[]`
- `mapping(keyType ==> valueType)`
- One can create their own type using `struct`

State Variables: Visibility

- Public - can be read by other contracts
 - a getting function will be auto generate
- Internal - can only be read by this contract and derived contracts
- Private - can only be read by this contract

Note: state variables are **not** writable by other contracts.

ⓘ Warning

Making something `private` or `internal` only prevents other contracts from reading or modifying the information, but it will still be visible to the whole world outside of the blockchain.

Functions

- Functions are blocks of statements
- Visibility (common ones in red)
 - `public` – can be called by any contract and account
 - `private` – can be called by this contract
 - `internal` – can be called by this contract or its derivation
 - `external` – can not be called by this contract
- State mutability modifier (optional)
 - `view` – compile will guarantee that this function does NOT modify state.
 - `pure` – promise not to read from or modify the state.
- Payable – by default functions cannot receive ether (called with `<value>`)
 - `payable` changes that (more on this later)

ERC20 standard

- An **ERC20-compliant** token contract must provide at least the following functions and events:

```
interface IERC20 {  
    function totalSupply() external view returns (uint256);  
    function balanceOf(address account) external view returns (uint256);  
    function allowance(address owner, address spender) external view returns (uint256);  
    function transfer(address recipient, uint256 amount) external returns (bool);  
    function approve(address spender, uint256 amount) external returns (bool);  
    function transferFrom(address sender, address recipient, uint256 amount) external returns (bool);  
    event Transfer(address indexed from, address indexed to, uint256 value);  
    event Approval(address indexed owner, address indexed spender, uint256 value);  
}
```

- Internal implementation, including how tokens are minted, may vary.
- Check out <https://github.com/openzeppelin/openzeppelin-contracts/blob/master/contracts/token/ERC20/IERC20.sol> and <https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/token/ERC20/ERC20.sol>

Typical implementation

Key state variables

- `mapping (address => uint256) private _balances;`
- `mapping (address => mapping (address => uint256)) private _allowances; // ignore this for now`

Metadata...

- `uint256 private _totalSupply;`
- `string private _name;`
- `string private _symbol;`
- `uint8 private _decimals;`

Typical implementation

```
/**
 * @dev Returns the amount of tokens in existence.
 */
function totalSupply() external view returns (uint256);
```

```
/**
 * @dev Returns the amount of tokens owned by `account`.
 */
function balanceOf(address account) external view returns (uint256);
```

```
/**
 * @dev Moves `amount` tokens from the caller's account to `recipient`.
 *
 * Returns a boolean value indicating whether the operation succeeded.
 *
 * Emits a {Transfer} event.
 */
function transfer(address recipient, uint256 amount) external returns (bool);
```

- Externally visible
- **View**: does not change state
- Send msg.sender's tokens to recipient
- Throw error if $\text{balanceOf}(\text{msg.sender}) < \text{amount}$
- Need Safe math: E.g., `"_balances[msg.sender] -= amount"` might cause overflow

approval + transferFrom flow

```
function approve(address spender, uint256 amount)
```

- **Token owner approves spender to spend her tokens up to amount**

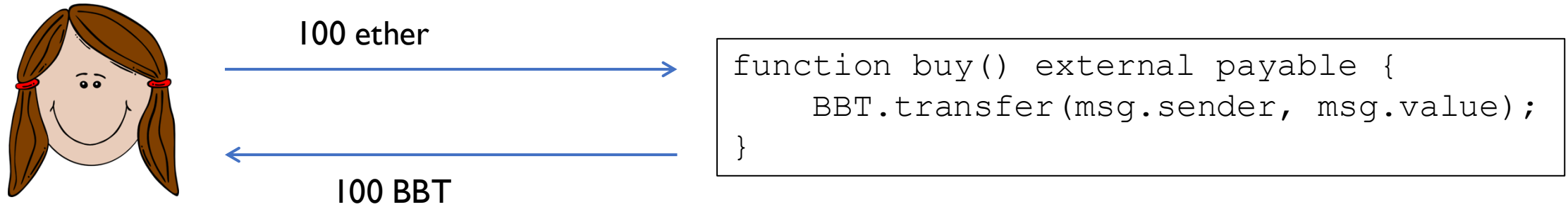
```
function allowance(address owner, address spender) external  
view returns (uint256);
```

- **Check how much has been approved**

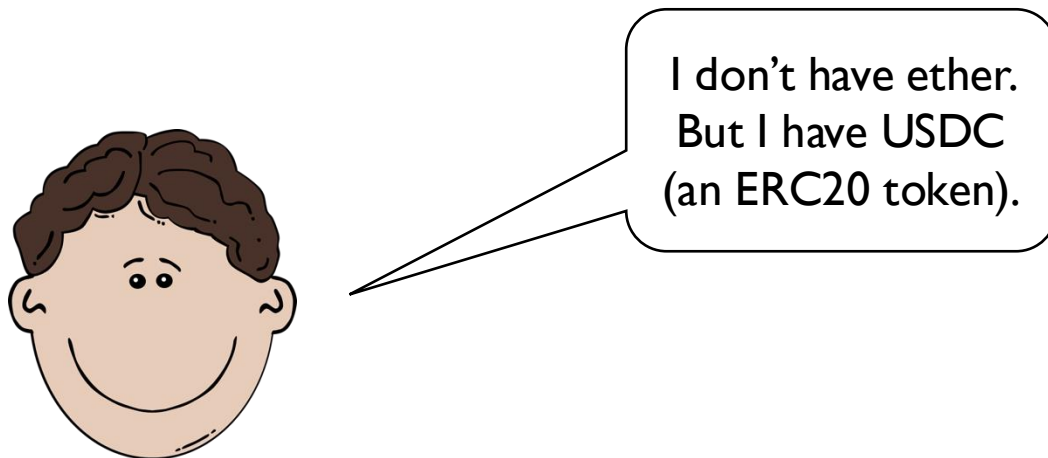
```
function transferFrom(address owner, address recipient,  
uint256 amount)
```

- **Transfer amount tokens from owner to recipient.**
- **Error if allowance(owner, msg.sender) < amount OR balanceOf(owner) < amount**

Function can react to Ether payments

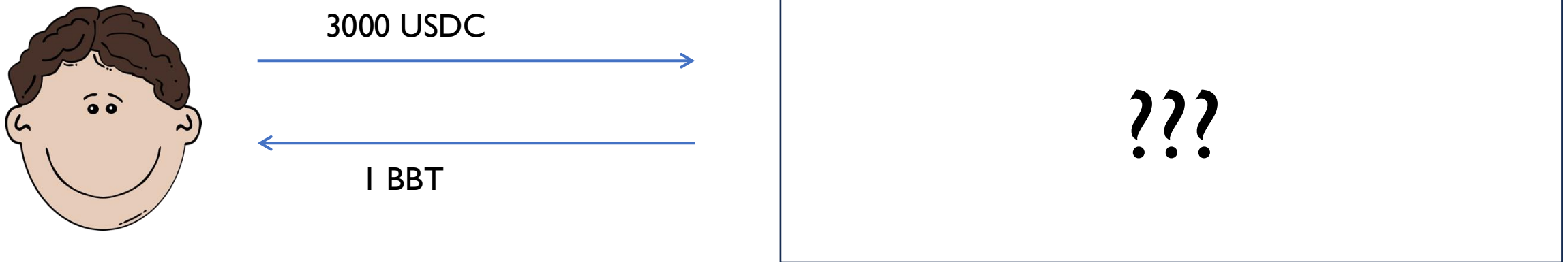


An exchange selling Big Bubble Token (BBT) at 1 ether



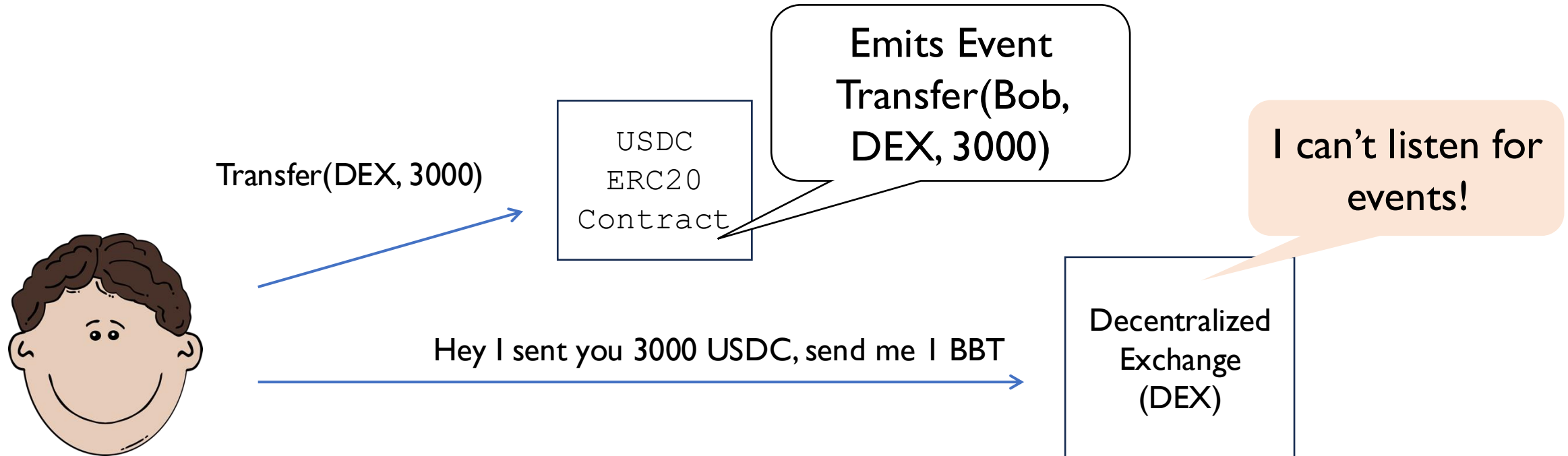
Can functions react to token payments?

- Task: Write a smart contract to sell BBT tokens at 3000 USDC each?



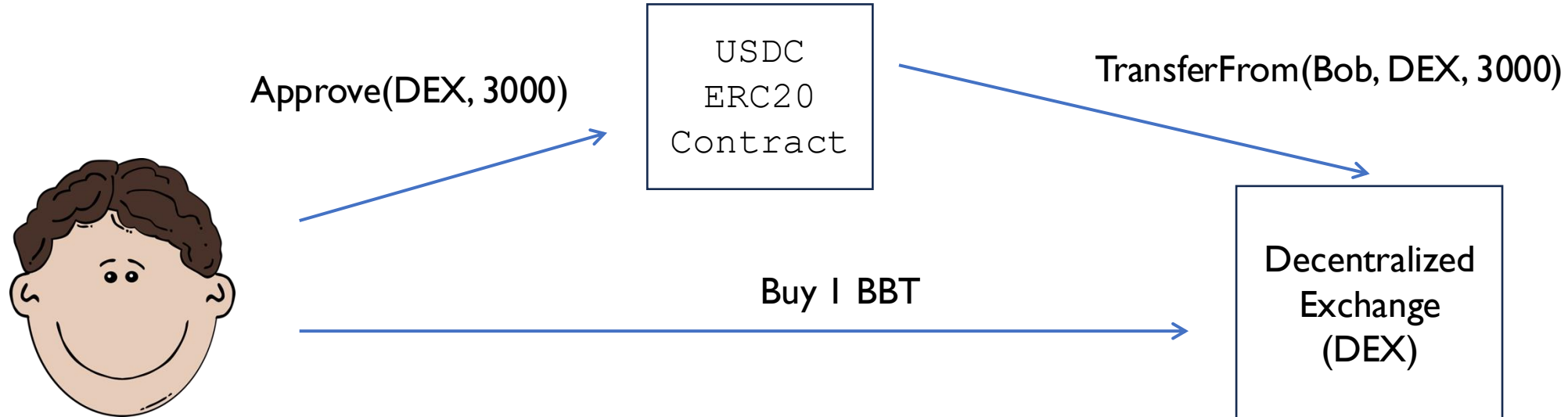
Can functions react to token payments?

- Task: Write a smart contract to sell BBT tokens at 3000 USDC each?



Can functions react to token payments?

- Task: Write a smart contract to sell BBT tokens at 3000 USDC each?



ERC-20 Tokens vs ether

Ether

- Balance is part of **user account state**
- “to” == the **recipient**
- Ether transfers change the state of recipient
- Ether can be sent to any address and trigger payable function execution.

ERC-20 Tokens

- Balance is part of **token contract's state**
- “to” == token contract
 - **Recipient** is specified in calldata given to transfer(From)
- Token transfers changes the state of the token contract.
 - It does **not** change recipient's state
- Tokens can be sent to any address but it **cannot** execute any code of recipient
 - This is a limitation!

ERC-721: NFTs

- Non-fungible Tokens (NFTs)
- Each token represents a **unique** deed to some asset
- Instead of tracking balance, ERC721 tracks owners of each token
 - mapping (uint256 => address) private deedOwner;

```
interface ERC721 /* is ERC165 */ {  
    function balanceOf(address _owner) external view returns (uint256 _balance);  
    function ownerOf(uint256 _deedId) external view returns (address _owner);  
    function transfer(address _to, uint256 _deedId) external payable;  
    function transferFrom(address _from, address _to, uint256 _deedId)  
        external payable;  
    function approve(address _approved, uint256 _deedId) external payable;  
    function setApprovalForAll(address _operator, boolean _approved) payable;  
    function supportsInterface(bytes4 interfaceID) external view returns (bool);  
}
```

Some famous NFTs



Founder Cat #18

18 Gen 0 Brisk Cooldown (1h)

ouou123
Owner

49

Not in any Collection

\$170,000 in 2018

Bored Apes

- **BAYC NFT #8817** has broken records, selling for an astounding \$3.4 million on Sotheby's Metaverse marketplace. (2022-2023?)





- “In May 2007, the digital artist known as Beeple set out to create and post a new work of art online every day. He hasn’t missed a day since, creating a new digital picture every day for 5,000 days straight. Individually known as **EVERYDAYS**, collectively, the pieces form the core of ***EVERYDAYS: THE FIRST 5000 DAYS***, one of the most unique bodies of work to emerge in the history of digital art.” ---
Description on Christies
- Digital art is converted to an NFT using a smart contract platform at [0x2a46f2ffd99e19a89476e2f62270e0a35bbf0756](https://opensea.io/address/0x2a46f2ffd99e19a89476e2f62270e0a35bbf0756)
- NFT sold for \$69m at Christie’s

Token standards evolve

- ERC-20 and ERC-721 are basic but widely used.
- ERC-223 (Token Standard)
 - Aimed at fixing a problem with ERC-20's design where tokens could be lost when transferred to a contract that is not designed to handle incoming tokens.
- ERC-777 (Advanced Token Standard)
 - An improvement over the ERC-20 standard, introducing features like more advanced handling of token transactions and hooks for smart contracts to react to token movements.
- Next: how do people exchange their tokens?

HoC: Our own NFTs

- <https://cpsc3640.netlify.app/hands-on/course-nft-erc721/>

Hands-On Component: Everyone gets a Handsome Dan

Lecture 8 · Monday, September 28, 2026

Follow the [Colab notebook](#) to learn why it is easy to get the rarest course NFT: Handsome Dan.

Do you have a Handsome Dan?

To succeed, make sure the address you submitted in Canvas last week owns a Handsome Dan. You can check it below.

Enter your Ethereum address to see which course NFTs you currently own on Sepolia. No wallet connection is needed.

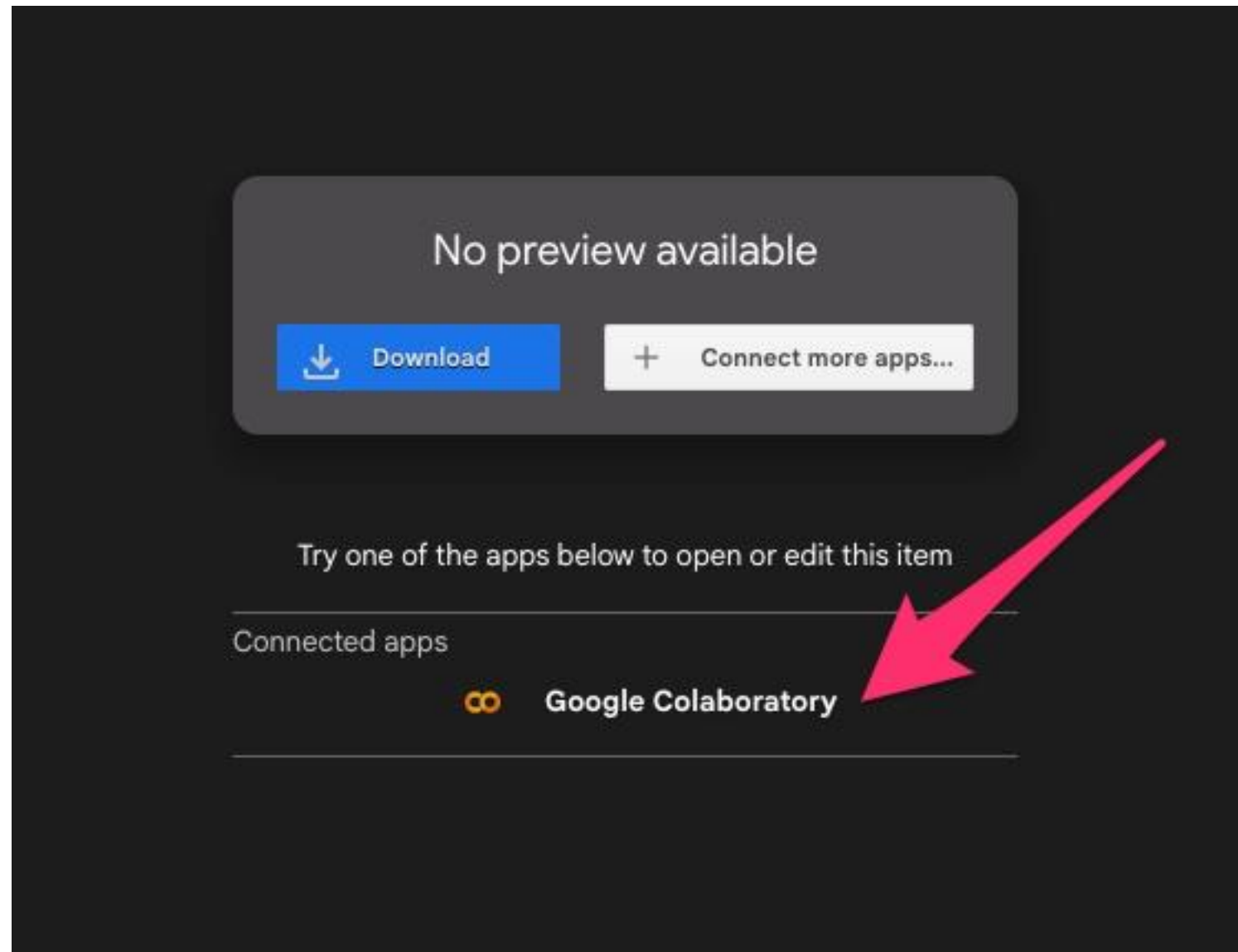
Your Ethereum address

Sepolia

0x...

Check NFTs →

Read-only. Your address is sent to PublicNode's Sepolia RPC service. Never enter a private key.



Login with your Yale account and open the notebook in Google Colab.