

 ERC-20 Basics – Tokenization Concepts

**Objective/Aim:**

To study the concept of tokenization on the Ethereum blockchain and understand the

implementation and working of ERC-20 tokens, including their basic functions and standards.

**Apparatus/Software Used:**

1. Remix IDE
2. MetaMask
3. Etherscan
4. OpenZeppelin Contracts
5. Brave Web Browser

**Theory/Concept:**

**Tokenization** refers to converting real-world or digital assets into blockchain-based tokens.

These tokens can represent currencies, shares, loyalty points, or any digital value that can be

transferred or traded securely.

# Key Features of ERC-20 Tokens:

The ERC-20 (Ethereum Request for Comment-20) standard defines a common set of rules that all

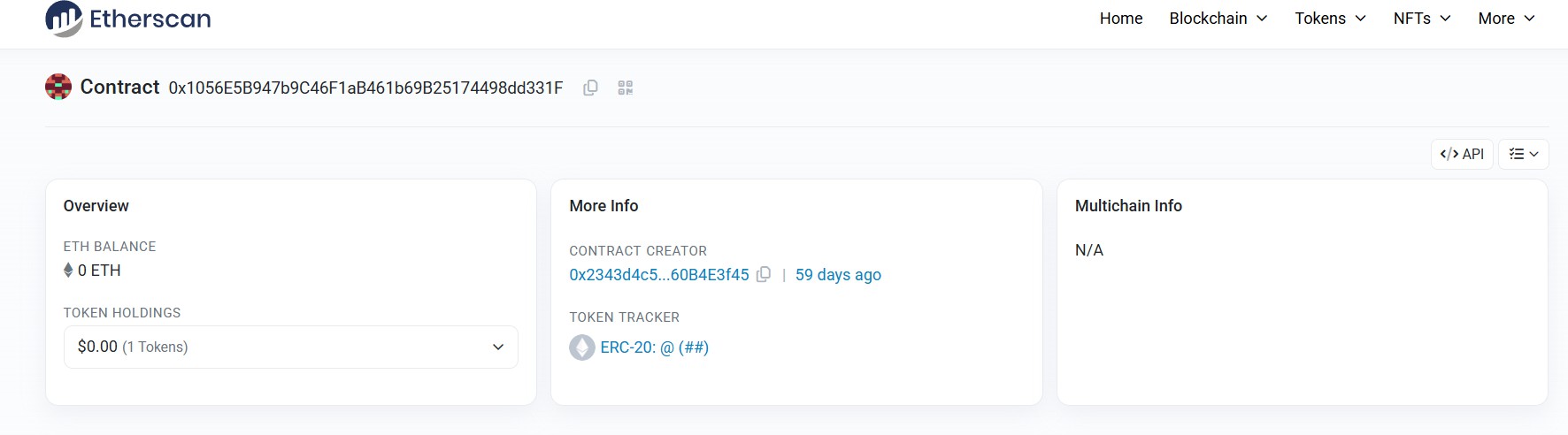
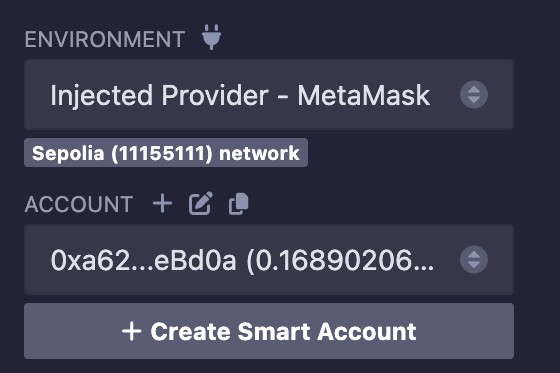
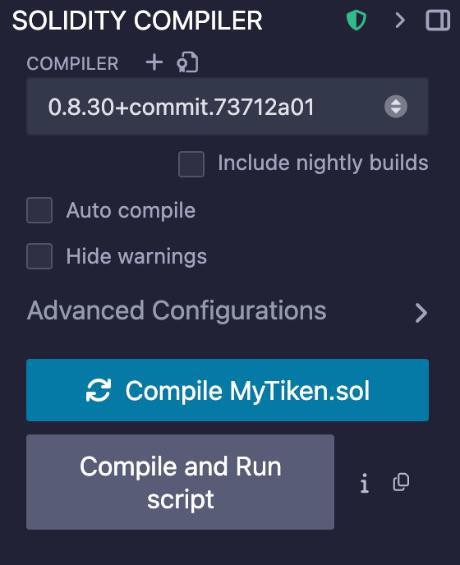
fungible tokens on the Ethereum blockchain must follow. It ensures that tokens can interact seamlessly with wallets, smart contracts, and decentralized applications (DApps).

1. **Fungibility:** Every token is identical and interchangeable.
2. **Interoperability:** Standard functions make tokens compatible with multiple platforms.
3. **Transparency:** All transactions are visible on the blockchain.
4. **Smart Contract-Based:** Token logic is programmed in Solidity.

# Common ERC-20 Functions:

* + totalSupply() – Returns total number of tokens.
  + balanceOf(address) – Shows balance of a specific address.
  + transfer(address, uint256) – Sends tokens to another address.
  + approve(address, uint256) – Allows another address to spend tokens.
  + transferFrom(address, address, uint256) – Transfers tokens using allowance.
  + allowance(address, address) – Shows approved spending limit.





**Procedure:**

**Step 1:** Open **Remix IDE**.

**Step 2: Create** a new **Solidity (.sol)** file.

**Step 3:** Write the **ERC-20 token smart contract** code.

**Step 4:** Compile the contract.

**Step 5:** Choose **Injected Provider – MetaMask** as the deployment environment.

**Step 6:** Deploy the contract through **MetaMask**.

**Step 7:** Approve and confirm the transaction in MetaMask.

**Step 8:** Copy the **deployed contract address**.

**Step 9:** Check and explore your token on **Etherscan**.

**Step 10:** Add the token to **MetaMask** using its contract address.

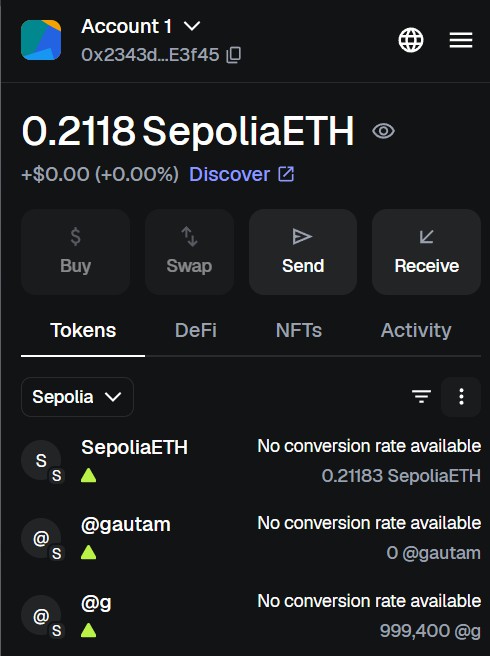
**Step 11:** In Remix, open the contract under **Deployed Contracts**.

**Step 12:** Use the **transfer function** to send tokens to another wallet.

**Step:1**

**Step:2 Step:3**

**Step:4**



**Step:5**

**Observation Table:**

1.The ERC-20 token contract was successfully compiled and deployed using Remix and MetaMask.

2.The token appeared in MetaMask after importing the contract address, confirming successful minting.

3.Token transfer to another wallet was executed and verified on Etherscan, confirming proper contract functionality.



****



