

ERC-20 Basics – Tokenization Concepts  
  
**Objective/Aim:**



To understand and implement the ERC-20 token standard, explore tokenization concepts, and develop a

custom ERC-20 token with advanced features including minting, burning, and access control mechanisms.

**Apparatus/Software Used:**

Laptop / PC with Node.js installed

Solidity development environment (Remix IDE or Hardhat)

MetaMask Wallet (Browser Extension)

Web3 development framework (Ethers.js or Web3.js)

**Theory/Concept:**

**What is Tokenization?**

Tokenization is the process of converting real-world assets or utility into digital tokens on a blockchain network. These tokens can represent:

* **Fungible Assets:** Currency, commodities, shares (like ERC-20 tokens)
* **Non-Fungible Assets:** Unique items, collectibles, property deeds (like ERC-721 NFTs)
* **Utility Tokens:** Access rights, voting power, service credits
* **Security Tokens:** Investment contracts, bonds, equity

**ERC-20 Token Standard:**

ERC-20 (Ethereum Request for Comment 20) is a technical standard that defines a common set of rules for Ethereum tokens. It ensures interoperability between different tokens and applications.



**Procedure:**

#### **Token Economics (Tokenomics):**

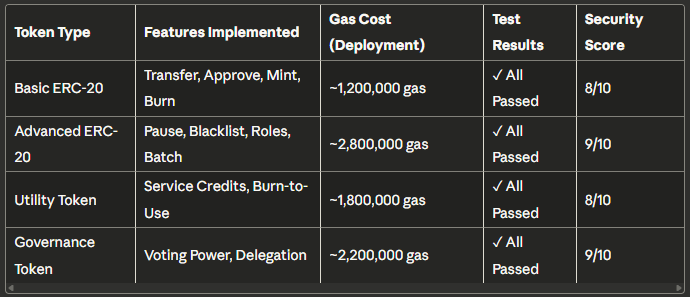
* **Total Supply:** Maximum number of tokens that can exist
* **Initial Supply:** Tokens created at deployment
* **Minting:** Creating new tokens (increasing supply)
* **Burning:** Destroying tokens (decreasing supply)
* **Distribution:** How tokens are allocated among stakeholders

#### **Access Control Patterns:**

* **Ownership:** Single address controls critical functions
* **Role-Based Access:** Multiple roles with specific permissions
* **Multi-signature:** Requires multiple approvals for actions
* **Timelock:** Delays execution of critical functions

**Observation Table:**

Token Development Observations:



|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

