Smart contract and Ethereum:
Overview of Ethereum, Writing Smart Contract in Solidity, Remix IDE , Different networks of ethereum, understanding blocks in blockchain, compilation and deployment of smart contracts in Remix

---

Overview of Ethereum

Introduction to Ethereum

- World's first programmable blockchain platform
- Created by Vitalik Buterin in 2015
- Native cryptocurrency: Ether (ETH)
- Key features:
    - Decentralized platform
    - Turing-complete programming
    - Smart contract functionality
    - Decentralized applications (DApps)

All About Ethereum

Ethereum revolutionized blockchain technology by introducing programmability beyond simple transactions. Unlike Bitcoin, which primarily focuses on currency transactions, Ethereum created a complete computing platform that can:

- Execute complex programmatic instructions
- Store and process data
- Support multiple types of digital assets
- Enable autonomous programs (smart contracts)
- Process conditional transactions
- Handle complex business logic

The platform provides a global, decentralized computer (the Ethereum Virtual Machine or EVM) that developers can use to deploy and run applications. This was groundbreaking because it transformed blockchain from a simple ledger into a computational platform.

Created by Vitalik Buterin in 2015

The development of Ethereum has an interesting history:

- Conceived in 2013 when Vitalik Buterin, then 19, observed Bitcoin's limitations
- Formally proposed in late 2013 through the Ethereum White Paper
- Development funded through a public crowdsale in 2014
- Officially launched on July 30, 2015
- Initial team included:
    - Vitalik Buterin (Founder)
    - Gavin Wood (Technical co-founder)
    - Charles Hoskinson
    - Joseph Lubin

The platform has undergone several major upgrades since launch:

- Homestead (2016)
- Metropolis (2017)
- Constantinople (2019)
- London (2021)
- The Merge (2022) - Transition to Proof of Stake

ETHER- The Native Cryptocurrency of Ethereum

Ether serves multiple purposes in the Ethereum ecosystem:

Currency Functions

- Medium of exchange within the network
- Store of value
- Trading asset on cryptocurrency markets
- Collateral in DeFi applications

Network Utility

- Gas payments for transaction processing
- Staking in Proof of Stake consensus
- Network security through economic incentives
- Governance participation

Technical Characteristics

- Divisible to 18 decimal places (smallest unit called "wei")
- Dynamic supply model
- Used for transaction fees (gas)
- Required for smart contract deployment and execution

Key Features

Decentralized Platform

- Network Architecture:
    - Distributed node network
    - No central authority
    - Peer-to-peer connections
    - Global distribution
- Decentralization Aspects:
    - Infrastructure (nodes)
    - Development
    - Governance
    - Access rights
    - Data storage

Turing-Complete Programming

Ethereum's programming capability is Turing-complete, meaning it can:

- Perform any computational operation
- Execute loops and conditional statements
- Handle complex data structures
- Implement sophisticated algorithms
- Support recursive functions

This is achieved through:

- The Ethereum Virtual Machine (EVM)
- Gas mechanism to prevent infinite loops
- Solidity programming language
- Other EVM-compatible languages

Smart Contract Functionality

Smart contracts are self-executing programs that run on the Ethereum blockchain:

Features:

- Autonomous execution
- Immutable once deployed
- Transparent rules and conditions
- Deterministic outcomes
- Trustless operations

Capabilities:

- Token creation
- Automated transactions
- Complex business logic
- Multi-signature wallets
- Escrow services
- Automated market makers

Decentralized Applications (DApps)

DApps are applications built on Ethereum that combine:

- Smart contracts (backend)
- Web interface (frontend)
- Decentralized storage
- Blockchain integration

Ethereum Architecture

- Account types:
  - Externally Owned Accounts (EOAs)
  - Contract Accounts
- Gas system:
  - Transaction fee mechanism
  - Resource allocation
  - Gas price and gas limit
- State transitions and transactions

- Ethereum Virtual Machine (EVM)

Ethereum Architecture Deep Dive

Account Types in Ethereum

Externally Owned Accounts (EOAs)

- Controlled by private keys
- Characteristics:
    - Has an Ethereum address
    - Can hold ETH and tokens
    - Can initiate transactions
    - No associated code
    - Controlled by private/public key pair

- Capabilities:
    1. Send ETH and tokens
    2. Create smart contracts
    3. Trigger smart contract functions
    4. Sign messages

```
Account Properties:
- Address (20 bytes)
- Nonce (transaction count)
- Balance (in Wei)
- No code
- No storage
```

Contract Accounts

- Controlled by code
- Characteristics:
    - Has an Ethereum address
    - Contains smart contract code
    - Activated only by EOA transactions
    - Autonomous execution

- Capabilities:
    1. Store code and data
    2. Execute contract logic
    3. Call other contracts
    4. Hold ETH and tokens

```
Contract Account Properties:
- Address (20 bytes)
- Nonce (contract creation count)
- Balance (in Wei)
- Contract code
- Storage (key-value store)
```

Gas System

Transaction Fee Mechanism

- Purpose:
    - Prevent network spam
    - Compensate validators
    - Resource management

- Components

```
Transaction Fee = Gas Used × Gas Price
Base Fee (determined by network)
Priority Fee (tip to validators)
```

Resource Allocation

- Operation costs:
    - Simple transfer: 21,000 gas
    - Contract deployment: variable (usually >200,000)
    - Contract function calls: varies by complexity

- Cost factors:
    1. Computation complexity
    2. Storage usage
    3. Memory usage
    4. Stack operations

Gas Price and Gas Limit

Gas Price:

- Measured in Wei per gas unit
- Dynamic based on network demand
- EIP-1559 pricing model:
    - Base fee (burned)
    - Priority fee (to validators)

Gas Limit:

- Maximum gas allowed per transaction
- Set by transaction sender
- Block gas limit (~15 million)
- Protection against infinite loops

State Transitions and Transactions

State in Ethereum

- World State
    - Collection of accounts
    - Account balances
    - Contract storage
    - Nonces

- State Storage

```
State Structure:
address → Account {
    nonce: 0,
    balance: 100,
    storageRoot: hash,
    codeHash: hash
}
```

Transactions

- Types:
    1. Regular ETH transfers
    2. Contract deployments
    3. Contract function calls
- Transaction Structure:

```
Transaction {
    nonce: uint,
    gasPrice: uint,
    gasLimit: uint,
    to: address,
    value: uint,
    data: bytes,
    v, r, s: bytes32 // signature components
}
```

Ethereum Virtual Machine (EVM)

EVM Architecture

Components:

1. Stack
   - Main working memory

2. Memory
   - Volatile storage

3. Storage
   - Persistent
   - Key-value store
   - Contract state

4. Call Data
   - Read-only
   - Function calls
   - Transaction input

Execution Model

- Bytecode Operations:

```
PUSH1 0x60    // Push value onto stack
MSTORE        // Store in memory
SLOAD         // Load from storage
CALL          // External call
```

- Execution Cycle:
  1. Fetch instruction
  2. Decode operation
  3. Execute operation
  4. Update state
  5. Move to next instruction

EVM Instructions

- Categories:
  1. Stack Operations
     - PUSH, POP, DUP, SWAP
  2. Arithmetic Operations
     - ADD, SUB, MUL, DIV
  3. Memory Operations
     - MLOAD, MSTORE, MSTORE8
  4. Storage Operations
     - SLOAD, SSTORE
  5. Control Flow
     - JUMP, JUMPI, STOP, RETURN
  6. Environmental
     - ADDRESS, BALANCE, CALLER

Smart Contract Deployment

Process:

1. Compile to bytecode
2. Send creation transaction
3. EVM executes constructor
4. Store contract code
5. Return contract address

Security

- Security Features:
  1. Sandboxed execution
  2. Gas limits
  3. Static typing
  4. Deterministic execution

- Common Considerations:
  1. Reentrancy protection

2. Integer overflow checks
3. Access control
4. Gas optimization

Articles:

1. https://medium.com/@filbuilders/filecoin-101-a-guide-to-blockchain-and-filecoin-essentials-a9ddb42de111
2. https://medium.com/@filbuilders/how-to-configure-metamask-to-work-with-the-filecoin-7184c437d1f1
3. https://medium.com/@filbuilders/beyond-storage-how-fvm-is-revolutionizing-the-filecoin-ecosystem-cd6fe102340d
4. Smart Contract Deployment on Remix IDE: https://docs.filecoin.io/smart-contracts/fundamentals/erc-20-quickstart
5. https://medium.com/@filbuilders/building-your-first-decentralized-application-on-filecoin-7c787659c5cc
6. https://medium.com/@filbuilders/creating-a-user-friendly-dapp-for-file-storage-on-filecoin-f51a48a77dd8