ETHEREUM DEVELOPMENT DOCUMENTATION

Last edit: @wackerow, August 16, 2022

See contributors

This documentation is designed to help you build with Ethereum. It covers Ethereum as a concept, explains the Ethereum tech stack, and documents advanced topics for more complex applications and use cases.

This is an open-source community effort, so feel free to suggest new topics, add new content, and provide examples wherever you think it might be helpful. All documentation can be edited via GitHub – if you're unsure how, follow these instructions.

DEVELOPMENT MODULES

If this is your first attempt at Ethereum development, we recommend starting at the beginning and working your way through like a book.

Foundational topics

- Intro to Ethereum – *A quick overview of Ethereum*
- Intro to Ether – *A quick overview of Ether*
- Intro to dapps – *An introduction to decentralized applications*
- Web2 vs Web3 – *The fundamental differences that blockchain-based applications provide*
- Accounts – *Entities in the network that can hold a balance and send transactions*
- Transactions – *Transfers and other actions that cause Ethereum's state to change*
- Blocks – *The way transactions are batched to ensure state is synchronised across all actors*
- Ethereum virtual machine (EVM) – *The EVM handles all the computation on the Ethereum network*
    - Opcodes
- Gas – *Computational power required to process transactions, paid for in ETH by transaction senders*
- Nodes and clients – *The individuals participating in the network and the software they run to verify transactions*
    - Run a node
    - Client diversity
    - Nodes as a service

    - Node architecture
    - Light clients
    - Archive nodes
    - Bootnodes
- Networks – *Implementations of Ethereum including test networks*
- Consensus mechanisms – *How the individual nodes of a distributed network agree on the current state of the system*
    - Proof-of-work
    - Proof-of-stake

Ethereum stack

- Intro to the stack – *An overview of the Ethereum/web3 stack*
- Smart contracts – *Programs that reside at an Ethereum address and run functions when triggered by transactions*
    - Smart contract languages
    - Smart contract anatomy
    - Smart contracts libraries
    - Testing smart contracts
    - Compiling smart contracts
    - Deploying smart contracts
    - Verifying smart contracts
    - Upgrading smart contracts
    - Smart contract security
    - Smart contract formal verification
    - Composability
- Development networks – *Local blockchain environments*

used to test dapps before deployment

- Development frameworks – *Tools that make developing with Ethereum easier*
- Ethereum client APIs – *Convenience libraries that allow your web app to interact with Ethereum and smart contracts*
  - JavaScript APIs
  - Backend APIs
  - JSON-RPC
- Data and analytics – *How blockchain data is aggregated, organized and implemented into dapps*
  - Block explorers
- Storage – *Decentralized storage structures and mechanism*
- Integrated Development Environments (IDEs) – *The best environments to write dapp code*
- Programming languages – *How to get started with Ethereum using languages you may already know*
  - Dart
  - Delphi
  - .NET
  - Golang
  - Java
  - JavaScript
  - Python
  - Ruby
  - Rust

Advanced

- Bridges – *An overview of bridging for developers*
- Standards – *Agreed upon protocols for maintaining efficiency and accessibility of projects to the community*
  - Token standards
- Maximal extractable value (MEV) – *How value is extracted from the Ethereum blockchain beyond the block reward*
- Oracles – *How information is injected into the Ethereum blockchain*
- Scaling – *Methods for preserving decentralization and security as Ethereum grows*
  - Optimistic rollups
  - Zero-knowledge rollups
  - State channels
  - Sidechains
  - Plasma
  - Validium
- Data availability – *docs-nav-data-availability-description*
- Networking layer – *Explanation of Ethereum's networking layer*
  - Network addresses
  - Portal Network
- Data structures and encoding – *Explanation of the data structures and encoding schema used across the Ethereum stack*
  - Patricia Merkle Trie
  - Recursive-length prefix (RLP)
  - Simple serialize (SSZ)
  - Web3 secret storage definition

Back to top ↑

Was this article helpful?

Yes No

Website last updated: November 16, 2023

NEXT
Intro to Ethereum

Use Ethereum

Find wallet

Get ETH

Decentralized applications (dapps)

Layer 2

Run a node

Stablecoins

Stake ETH

Learn

Learn Hub

What is Ethereum?

What is ether (ETH)?

Ethereum wallets

Gas fees

Ethereum security and scam prevention

What is Web3?

Smart contracts

Ethereum energy consumption

Ethereum roadmap

Ethereum Improvement Proposals

History of Ethereum

Ethereum Whitepaper

Ethereum glossary

Ethereum governance

Blockchain bridges

Zero-knowledge proofs

Quiz Hub

Developers

Get started

Documentation

Tutorials

Learn by coding

Set up local environment

Ecosystem

Community hub

Ethereum Foundation

Ethereum Foundation Blog

Ecosystem Support Program

Ethereum bug bounty program

Ecosystem Grant Programs

Ethereum brand assets

Devcon

Enterprise

Mainnet Ethereum

Private Ethereum

Enterprise

About ethereum.org

About us

Jobs

Contributing

Language support

Privacy policy

Terms of use

Cookie policy

Contact