

Starting with Ethereum Blockchain Development

Solidity contracts and a simple decentralised application

FA-0697 · 3 days

Software Development

Introduction

This three-day course introduces blockchain concepts through Ethereum and a small Solidity project. Participants compile, test, deploy and interact with selected contracts using a prepared local development network and a simple web interface.

The course uses current development tooling and distinguishes local/test-network practice from production deployment. It introduces contract-security review without claiming a short workshop produces audited or production-ready contracts.

Learning outcomes

- Explain blockchain transactions, validation and public/private or permissioned/permissionless distinctions.
- Describe Ether, gas, the EVM and Ethereum proof-of-stake validators.
- Write, compile and test a small Solidity contract.
- Deploy and interact with a sample contract on a local or suitable test network.
- Use Hardhat, a supported client library and a test wallet to connect a simple interface.
- Identify selected contract vulnerabilities and the need for further testing and review.

Prerequisites

- Basic understanding of programming concepts (variables, functions, control flow).

- Familiarity with JavaScript or similar programming languages.
- Basic understanding of web development (HTML, CSS).
- Knowledge of asynchronous programming concepts (promises, async/await).

A supported Node.js/npm version and toolchain matching the selected Hardhat release.

Training outline

1. Day 1 — Blockchain and Ethereum foundations

What is Blockchain?

- Definition and key characteristics.
- History and evolution of blockchain.

How Blockchain Works

- Concept of decentralized networks.
- Blocks, transactions and consensus: mining in proof-of-work networks versus validators in Ethereum proof of stake

Types of Blockchains

- Public vs. private blockchains.
- Permissioned vs. permissionless blockchains.

Ethereum concepts

Introduction to Ethereum

- What makes Ethereum unique.
- Ether, gas, and Ethereum Virtual Machine (EVM).

Smart Contracts Basics

- Definition and how they work.
- Use cases of smart contracts.

Development setup

- Solidity compiler, Hardhat project and local network, and MetaMask/test-wallet integration.
- Use prepared tool versions and test accounts; compare historical Truffle/Ganache workflows with maintained alternatives.

2. Day 2 – Solidity and contract deployment

Solidity Basics

- Syntax and structure of Solidity code.
- Data types, functions, and control structures.

Developing a Smart Contract

- Writing a basic smart contract.
- Compiling and deploying smart contracts.

Deployment and transaction inspection

- Deploy through Hardhat to the local development network; inspect receipts, reverted transactions and gas.
- Exercise selected contract functions and basic tests before any test-network deployment.

3. Day 3 – Client integration, security and a simple DApp

- Use a supported ethers.js or viem integration to call contracts and submit test transactions.
- Build a small web interface and distinguish wallet signing from ordinary read-only calls.
- Review common vulnerability patterns, access controls and secure development practices.
- Integrate and test the bounded DApp; identify limitations and further review required before production.