Introduction to Smart Contracts

Pre-Class Discovery Handout

Lesson 14 · Complete before class · 25–30 minutes

Activity 1: Explore a Smart Contract on Etherscan

10 min · Individual

Visit etherscan.io and navigate to a well-known smart contract such as the USDC token contract (address: `0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48`). Explore the contract page and answer:

1. How many transactions has this contract processed? What is the total number of token holders?
2. Click the "Contract" tab and then "Read Contract." Find the `name()`, `symbol()`, and `decimals()` functions. What values do they return?
3. Click "Write Contract." What functions are available? Why would you need to connect a wallet to use them?
4. Look at the "Events" tab. What event is emitted most frequently? What information does a `Transfer` event contain (from, to, value)?

Bonus: Find another smart contract (e.g., Uniswap Router) and compare its complexity (number of functions, transaction volume) to the USDC contract.

Activity 2: The Vending Machine Analogy

5 min · Pairs

Nick Szabo described a vending machine as the simplest form of a smart contract: you insert money, the machine checks the amount, and it dispenses the product—no negotiation, no trust, no intermediary.

1. List three real-world processes (besides vending machines) that work like a smart contract: deterministic input, automatic verification, guaranteed output.
2. For each process, identify: (a) what is the "input," (b) what condition is checked, and (c) what is the automatic output.
3. Which of your examples could actually be implemented as a blockchain smart contract? What would be gained or lost?

Process	Input	Condition Checked	Automatic Output

Activity 3: Compare Smart Contract Platforms

10 min · Small Groups

Research and compare three smart contract platforms: Ethereum, Solana, and one platform of your choice (e.g., Cardano, Polkadot, Avalanche, or Arbitrum). Answer:

1. What programming language does each platform use for smart contracts?
2. What is the approximate transaction speed (TPS) and average transaction cost for each?
3. What consensus mechanism does each use? How does this affect security and decentralization?
4. Which platform would you choose to deploy a high-frequency trading DApp? A community governance DAO? Justify each choice.

Platform	Language	TPS / Cost	Consensus	Best For
Ethereum				
Solana				

Activity 4: Smart Contract Use Case Design

5 min · Individual

Design a simple smart contract for one of the following scenarios (or invent your own):

- A crowdfunding contract that refunds donors if a goal is not met by a deadline
- An escrow contract for freelance work that releases payment upon delivery confirmation
- A lottery contract where participants buy tickets and a winner is selected

1. What **state variables** would your contract need? (e.g., owner address, balance, deadline)
2. What **functions** would users call? (e.g., `contribute()`, `withdraw()`, `refund()`)
3. What **conditions** must be checked before executing each function? (e.g., "deadline has not passed," "caller is the owner")
4. What could go wrong? Identify at least one security risk and how you would mitigate it.

Contract name: _____ **Number of functions:** _____ **Main security risk:** _____

Key Terms

Term	Definition
Smart Contract	A self-executing program stored on a blockchain that automatically enforces the terms of an agreement when predetermined conditions are met, without requiring intermediaries.
Solidity	The most widely used programming language for writing smart contracts on Ethereum and EVM-compatible blockchains. Statically typed, influenced by JavaScript and C++.
EVM	Ethereum Virtual Machine. The runtime environment that executes smart contract bytecode on every node in the Ethereum network, ensuring deterministic computation.
Gas	A unit measuring the computational effort required to execute operations on Ethereum. Users pay gas fees (in ETH) to compensate validators for processing transactions.
Gwei	A denomination of ETH equal to 10^{-9} ETH (one billionth). Gas prices are typically quoted in Gwei (e.g., 20 Gwei per unit of gas).
Transaction	A signed message sent from an externally owned account to the blockchain. Transactions can transfer ETH, deploy contracts, or call contract functions.
Block	A batch of validated transactions bundled together and appended to the blockchain. Each block references the previous block's hash, forming an immutable chain.
Wallet	Software (e.g., MetaMask) or hardware (e.g., Ledger) that manages a user's private keys and enables signing transactions to interact with the blockchain.
Private Key	A secret 256-bit number that proves ownership of a blockchain address and authorizes transactions. Must never be shared; loss means permanent loss of funds.
Address	A 20-byte (160-bit) identifier derived from a public key, used to send and receive ETH and interact with smart contracts. Displayed as a 42-character hexadecimal string (0x...).
Deploy	The process of sending a transaction containing compiled smart contract bytecode to the blockchain, creating a new contract instance at a unique address.
ABI	Application Binary Interface. A JSON specification describing a smart contract's functions, inputs, outputs, and events, enabling external applications to interact with it.
DApp	Decentralized Application. A front-end application (web or mobile) that interacts with one or more smart contracts on a blockchain instead of a centralized server.
Oracle	A service (e.g., Chainlink) that feeds external, off-chain data (prices, weather, sports scores) into smart contracts, which cannot natively access data outside the blockchain.
Immutable	The property that once a smart contract is deployed to the blockchain, its code cannot be modified or deleted, ensuring permanence but requiring careful pre-deployment testing.