Solidity - Smart Contract Part 1

ljb630

Apr 27 2024

Laljan Basha Shaik (ljb630)

Technical Writer & Researcher

Contact: ljb630web3@gmail.com

I, Laljan Basha Shaik, am a proficient technical writer and researcher with substantial expertise in the technological domain, focusing particularly on blockchain and Web3 sectors. Since April 2023, I have been actively engaged in freelance projects that emphasize technical documentation and research development. My technical acumen spans a broad array of tools and platforms, including LaTeX, Mendeley, Information Technology, and the generation of technical reports in various formats such as PDF, XML, JSON, and Excel. My ability to demystify complex technical concepts and translate them into clear and accessible language is complemented by a meticulous approach to writing, editing, and proofreading. I collaborate closely with development teams to create technical guides, documentation, blog posts, and additional content, adhering to stringent data protection standards including DPA, SCC, DPIA, GPRI, SEPA, G2N, and TOM.

Currently, I lead a team of nine as a Technical Support Engineer at Mercans, where I am responsible for the design and development of a Software as a Service (SaaS) platform aimed at digitalizing payroll processes. My responsibilities include crafting user interface screens and engineering a feature that enables the generation of customizable reports. These contributions significantly improve data accessibility and elevate the user experience, underscoring my impact in the field of digital solutions.

Research Paper: Solidity and Smart Contracts

Here In this paper, we will explore the critical role of smart contracts within the blockchain ecosystem, particularly focusing on Solidity[1] as the primary language used for their major development. It discusses the distinctions between **blockchain core developers**, who construct the blockchain infrastructure from scratch, and **blockchain software developers**, who build decentralized applications (DApps) utilizing these infrastructures. The latter category, which is currently experiencing high demand in the Web3 space, involves integrating smart contracts with front-end applications using libraries such as Web3.js and Ether.js. This paper delves into the use of development frameworks like Hardhat and Truffle that facilitate such integrations, providing a technical overview and practical insights into the tools and methodologies employed by developers in this rapidly evolving field. The focus will be on public blockchain technologies, with a brief mention of private blockchain networks like Hyperledger, which will not be covered in detail in this work.

Further Exploration

For scholars and practitioners seeking to expand their understanding of smart contracts and the Solidity programming language, it is advisable to consult several key resources. These materials are curated to facilitate a deeper comprehension of both fundamental concepts and advanced implementations within the domain of blockchain technology.

1. Solidity Documentation: An essential resource for developers, the official Solidity documentation provides exhaustive details on the language's syntax, capabilities, and optimal practices for crafting robust smart contracts. [Access Solidity Documentation](https://soliditylang.org/).

2. Introduction to Smart Contracts: Chainlink offers an insightful blog that delineates the basic principles of smart contracts. This resource is ideal for individuals new to the blockchain field, elucidating how smart contracts function and their significance within digital transactions. [Explore Chainlink’s Educational Blog](https://chain.link/education/smart-contracts).

3. Hybrid Smart Contracts: To understand the innovative integration of on-chain and off-chain data within smart contracts, refer to Chainlink's educational segment on hybrid smart contracts. This guide sheds light on how external data sources and real-world events can be assimilated into blockchain applications, enhancing their functionality and applicability. [Learn More at Chainlink](https://chain.link/education-hub/hybrid-smart-contracts).

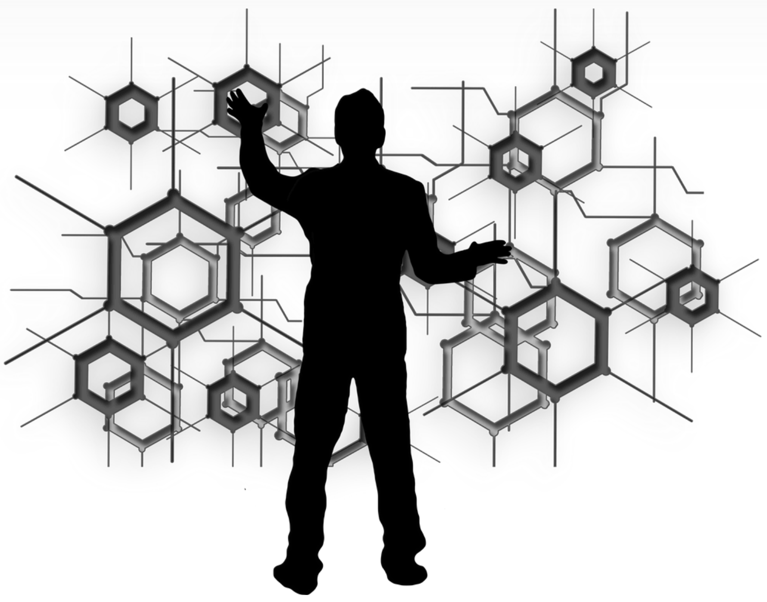
Career Transition to Web3

For students and working professionals aspiring to pivot their careers towards the Web3 sector, specifically blockchain development, a practical approach is essential. It is recommended to engage in hands-on learning experiences that offer a deep dive into the technologies and frameworks underpinning blockchain and smart contracts.

Blockchain Developer Training

Solidity Course by Cyfrin: A comprehensive course designed for those beginning their journey in blockchain development. This curriculum covers a range of topics from the basics of Solidity to advanced interactions with smart contracts. Delivered through engaging video lessons, the course makes complex concepts accessible and manageable, providing a solid foundation for anyone looking to enter the field of blockchain technology. [Explore Blockchain Development Courses on Cyfrin](https://updraft.cyfrin.io/courses).

Note: All underlined items are clickable and serve as redirected URLs to external websites.



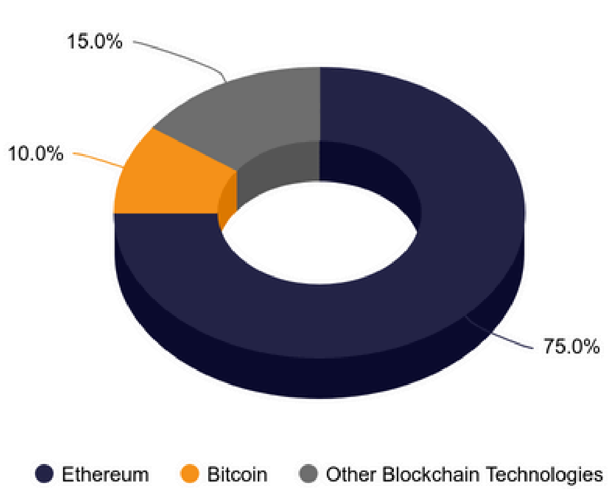
What is a Blockchain?

Continuing from the foundational understanding of blockchain as a distributed and immutable ledger, we delve deeper into its operational essence. At its core, blockchain functions as a decentralized database that securely records and stores every transaction across multiple nodes or computers within the network, ensuring that each entry is both visible and verifiable by all participants. This ledger resembles a traditional ledger in that it records debits and credits between parties; however, unlike traditional systems, once a transaction is entered into a blockchain, it cannot be altered or tampered with. This immutability is secured through the use of cryptographic hashing, which links each block to its predecessor in a chronological chain.

Each block in the chain contains a number of transactions, and once filled, is timestamped and linked to the previous block, thus forming a chain of blocks with information that is chained together. This chaining of blocks ensures that the blockchain is tamper-evident, adding a layer of security and trust that is crucial for many applications. The decentralized nature of blockchain not only enhances security but also democratizes the data, making it accessible to anyone within the network while removing the need for a central authority.

This design makes blockchain an ideal platform for recording not just cryptocurrency transactions, as is commonly associated with the technology, but also a wide variety of data types such as contracts, records, and other information that benefit from secure, permanent recording. Understanding these basic principles is essential for anyone looking to understand or work with blockchain technology, providing a sturdy foundation for exploring more complex applications and implications of this transformative platform.

The Major Blockchain Technologies In The Web3



What is Ethereum?

Ethereum, founded by Vitalik Buterin in 2013, represents a significant evolution in blockchain technology. Unlike Bitcoin, which is primarily designed as a digital currency, Ethereum is a programmable blockchain that allows developers to build and deploy decentralized applications (dApps). This capability has catalyzed a diverse range of applications beyond mere financial transactions.

- **Programmable**: Ethereum's most distinguishing feature is its Turing-complete programming language, Solidity, which enables the creation of smart contracts. These contracts automatically execute, control, and document legally relevant events according to the terms of a contract or an agreement.
- **Versatile Applications**: The programmable nature of Ethereum allows for applications that go beyond currency and financial transactions, supporting a wide variety of applications including voting systems, decentralized governance, and more.

Challenges with Real-Time Data:

A fundamental limitation of Ethereum, and indeed all blockchains, is their inability to access or interact with real-time data due to their isolated nature. This restriction hinders applications that require real-time information, such as price feeds, weather updates, or stock market data.

The Role of Oracles:

To bridge the gap between external data sources and the blockchain, Ethereum utilizes oracles. An oracle is a third-party service that fetches data from the outside world and feeds it into the blockchain. This enables smart contracts on Ethereum to react to external events and execute based on real-time data.

Further Exploration

Ethereum Foundation Official Website : (https://ethereum.org/)

Chainlink Official Documentation : (https://docs.chain.link/)

Solidity Language Documentation : (https://docs.soliditylang.org/)

Etherscan Ethereum Block Explorer : (https://etherscan.io/)

Blockchain - Protocols, and Tokens

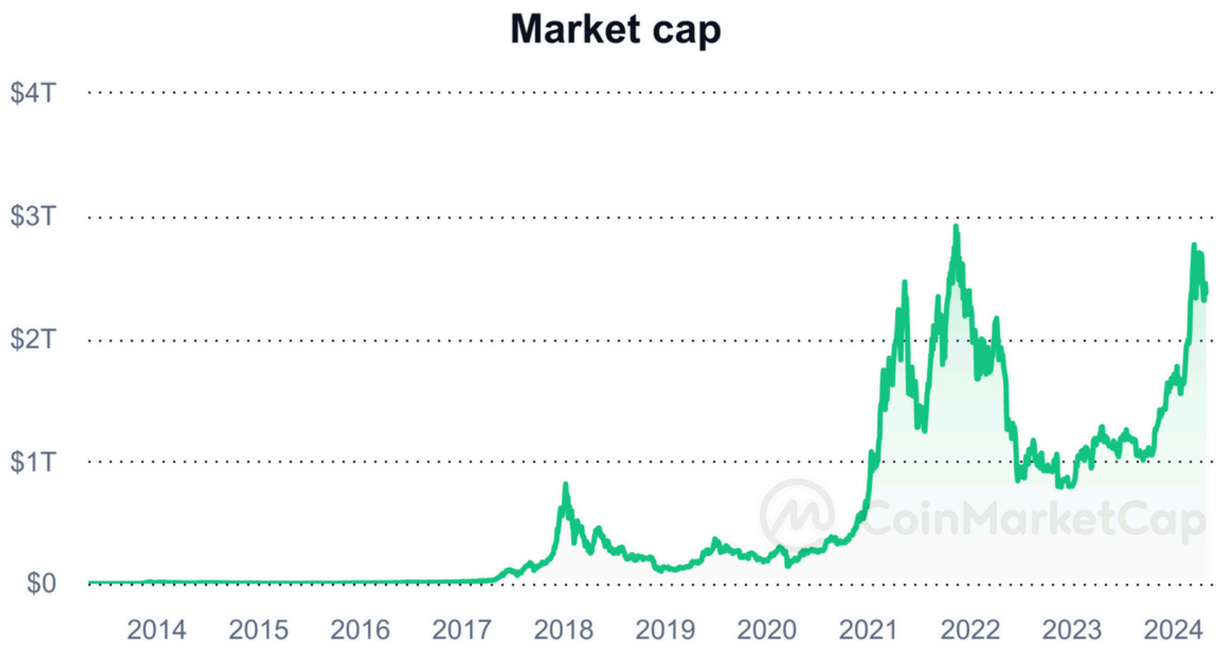
Blockchain technology stands as a paradigm of decentralization, reshaping how data is managed and transactions are processed across numerous industries. Unlike centralized systems, where a single entity has control, a decentralized blockchain distributes data across a network of computers, ensuring transparency, security, and resilience against attacks or failures.

Ethereum: As one of the most influential blockchain protocols, Ethereum not only supports transactions but also facilitates the execution of complex contracts and the creation of decentralized applications through its own native programming language, Solidity. Tokens generated on the Ethereum platform include TRX (Tron), SNT (Status), and REP (Augur).

Bitcoin: The pioneer of blockchain technology, Bitcoin remains primarily a digital currency system. Unlike Ethereum, Bitcoin does not support the creation of tokens within its protocol, focusing instead on enhancing its efficiency and security as a cryptocurrency.

NEO: Often referred to as the "Ethereum of China," NEO is designed to be a more scalable and government-friendly version of Ethereum. It supports the creation of digital assets and smart contracts, with tokens such as DBC (DeepBrain Chain), TKY (THEKEY), and TNC (Trinity Network Credit) generated on its platform.

Waves: Known for its unique approach to custom token creation, the Waves platform allows users to launch, distribute, and trade their own crypto tokens. Tokens associated with Waves include WGT (WavesGo), WCT (Waves Community Token), and INTL (Incent).



Initial Coin Offerings (ICOs):

ICOs represent a novel fundraising mechanism where new projects sell their underlying crypto tokens in exchange for bitcoin or ether. This method has become a popular means of capital raising for startups in the blockchain space. Each of the aforementioned protocols has hosted numerous ICOs, allowing investors to purchase tokens as a form of investment in the project. These tokens can serve various purposes, from granting holders access to certain features of a project to acting as a stake within the decentralized application.

Further Exploration

Protocol: https://www.coinbase.com/learn/crypto-basics/what-is-a-protocol

Token: https://www.coinbase.com/learn/crypto-basics/what-is-a-token

NEO: https://neo.org/

Waves: https://waves.tech/

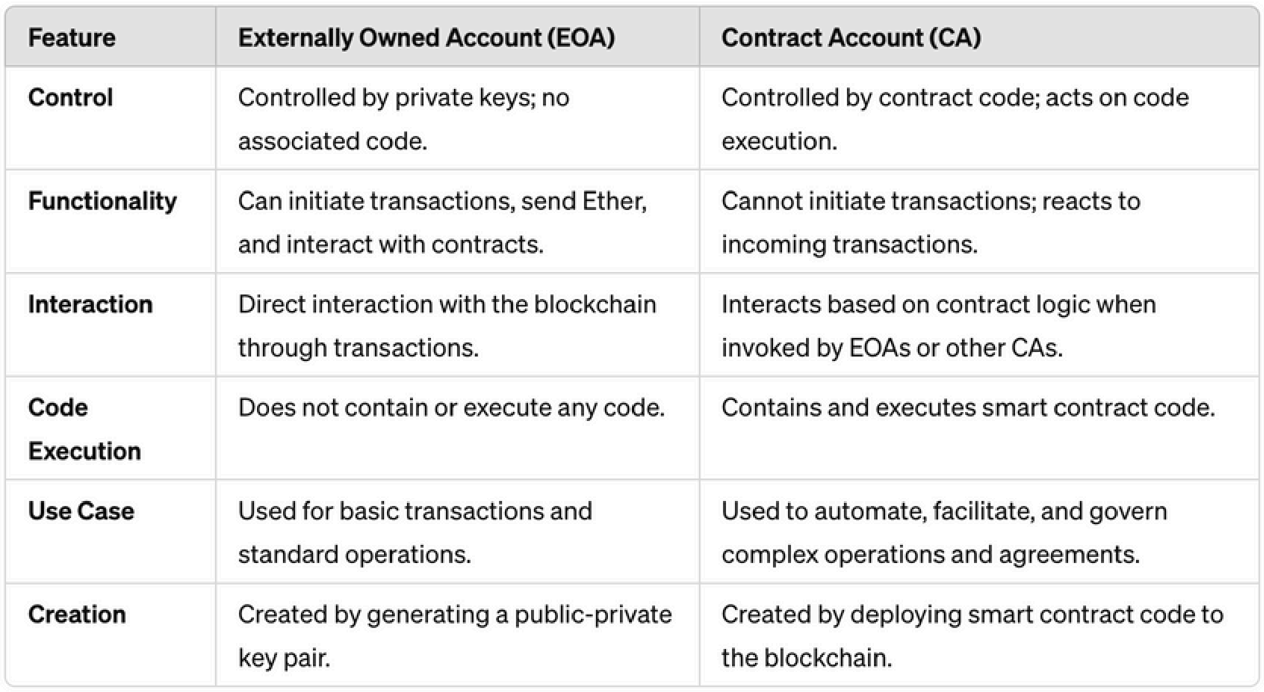
Understanding Ethereum Accounts

Ethereum accounts are fundamental components of Ethereum's architecture, facilitating the execution and recording of transactions on the blockchain. Each account is identified by an address and has an Ethereum balance. Accounts on Ethereum can be categorized into two types:

Externally Owned Accounts (EOA): These are controlled by private keys and have no associated code. EOAs can send transactions (transferring ether and interacting with smart contracts) and are operated by human users or an agent acting on their behalf.

Contract Accounts (CA): These are controlled by their contract code and can only act when triggered by a transaction. Unlike EOAs, contract accounts hold and execute smart contract code and are governed by that code rather than by users.

Overview:



Feature	Externally Owned Account (EOA)	Contract Account (CA)
Control	Controlled by private keys; no associated code.	Controlled by contract code; acts on code execution.
Functionality	Can initiate transactions, send Ether, and interact with contracts.	Cannot initiate transactions; reacts to incoming transactions.
Interaction	Direct interaction with the blockchain through transactions.	Interacts based on contract logic when invoked by EOAs or other CAs.
Code Execution	Does not contain or execute any code.	Contains and executes smart contract code.
Use Case	Used for basic transactions and standard operations.	Used to automate, facilitate, and govern complex operations and agreements.
Creation	Created by generating a public-private key pair.	Created by deploying smart contract code to the blockchain.

Wallet Options:

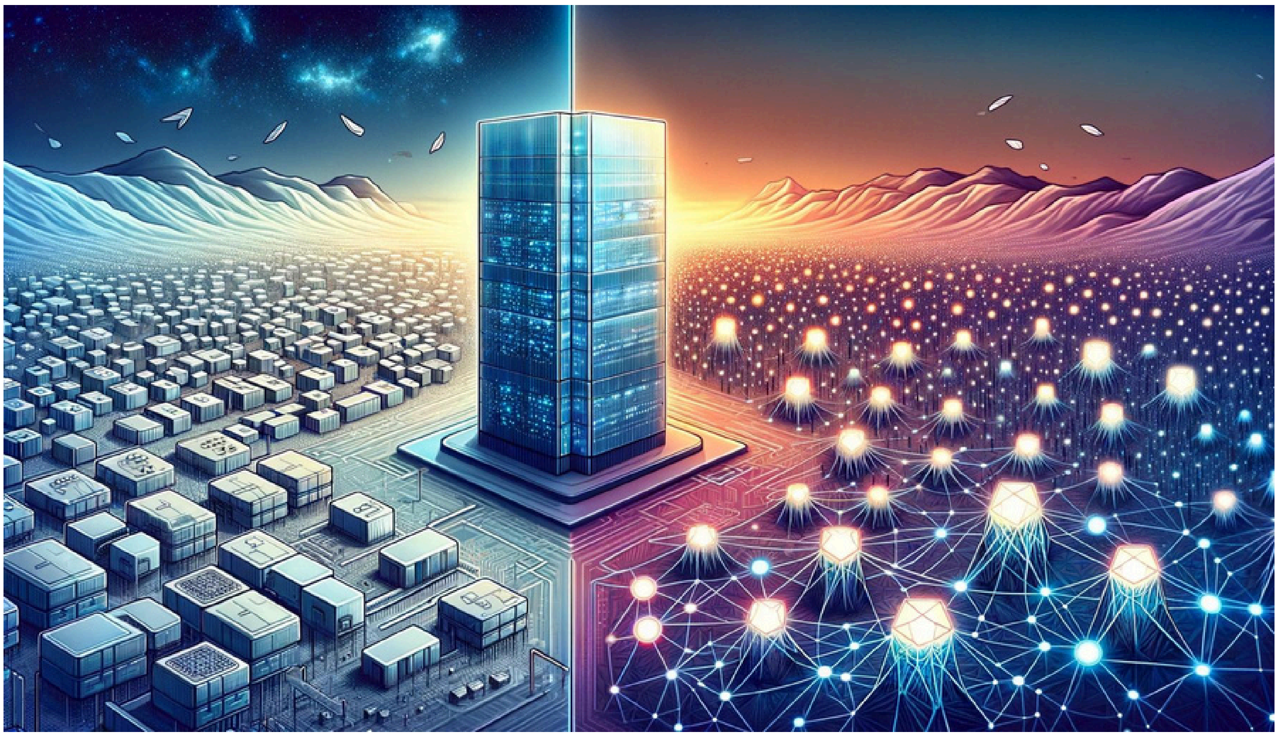
Popular wallet services like **MetaMask**, MyEtherWallet, or Ledger (hardware wallet) provide secure, user-friendly interfaces for managing your EOA.

Centralized Application:

This section of the image shows a centralized application with a single central server connected to multiple user nodes. It highlights how control and data flow are centralized at a single point, typical of traditional web services.

Decentralized Application (DApp):

The other section depicts a decentralized application, where multiple nodes are interconnected without a central server. This illustrates equal participation and data flow among all nodes, characteristic of blockchain-based applications.



The Above image will provide a detailed comparison between **centralized applications** and **decentralized applications (DApps).**

Centralized Applications BANK

Decentralized Applications (DApps) UNISWAP MAKER Rarible

The Ethereum Virtual Machine (EVM)

The Ethereum Virtual Machine (EVM) is a crucial component of the Ethereum ecosystem, acting as the runtime environment for smart contracts. It provides the necessary isolation and security framework that allows decentralized applications (dApps) to execute code safely and effectively on the Ethereum blockchain.

The EVM runs code in a completely isolated environment, ensuring that smart contracts operate without risking the integrity and security of the main network.

It enables developers to write smart contracts in high-level languages such as Solidity, which are then compiled into bytecode that the EVM can execute across any Ethereum node.

Gas: Represents the fundamental unit of computation used in Ethereum. Each operation that a smart contract performs has a fixed gas consumption.

Gas Price: Refers to the amount of Ether a user is willing to pay per unit of gas. Users can set higher gas prices to prioritize their transactions, which is particularly useful during periods of high network congestion.

GWEI: A denomination of Ether used to specify gas prices. 1 GWEI equals 1 billionth of an Ether. Using smaller units helps in fine-tuning transaction costs and makes the calculation of transaction fees more straightforward.

Gas Limit: Specifies the maximum amount of gas that a user is willing to spend on a transaction or smart contract execution. Setting an appropriate gas limit ensures that the user does not spend more on gas than intended.

Further Exploration

ETH Gas Station : (https://ethgasstation.info/)

GasNow: (https://cn.etherscan.com/gastracker)

Gas Fee Calculator : (https://www.cryptoneur.xyz/en/gas-fees-calculator)

CYBER SAVVY

thank you!

Solidity - Smart Contract Part 1

ljb630

June 1 2024