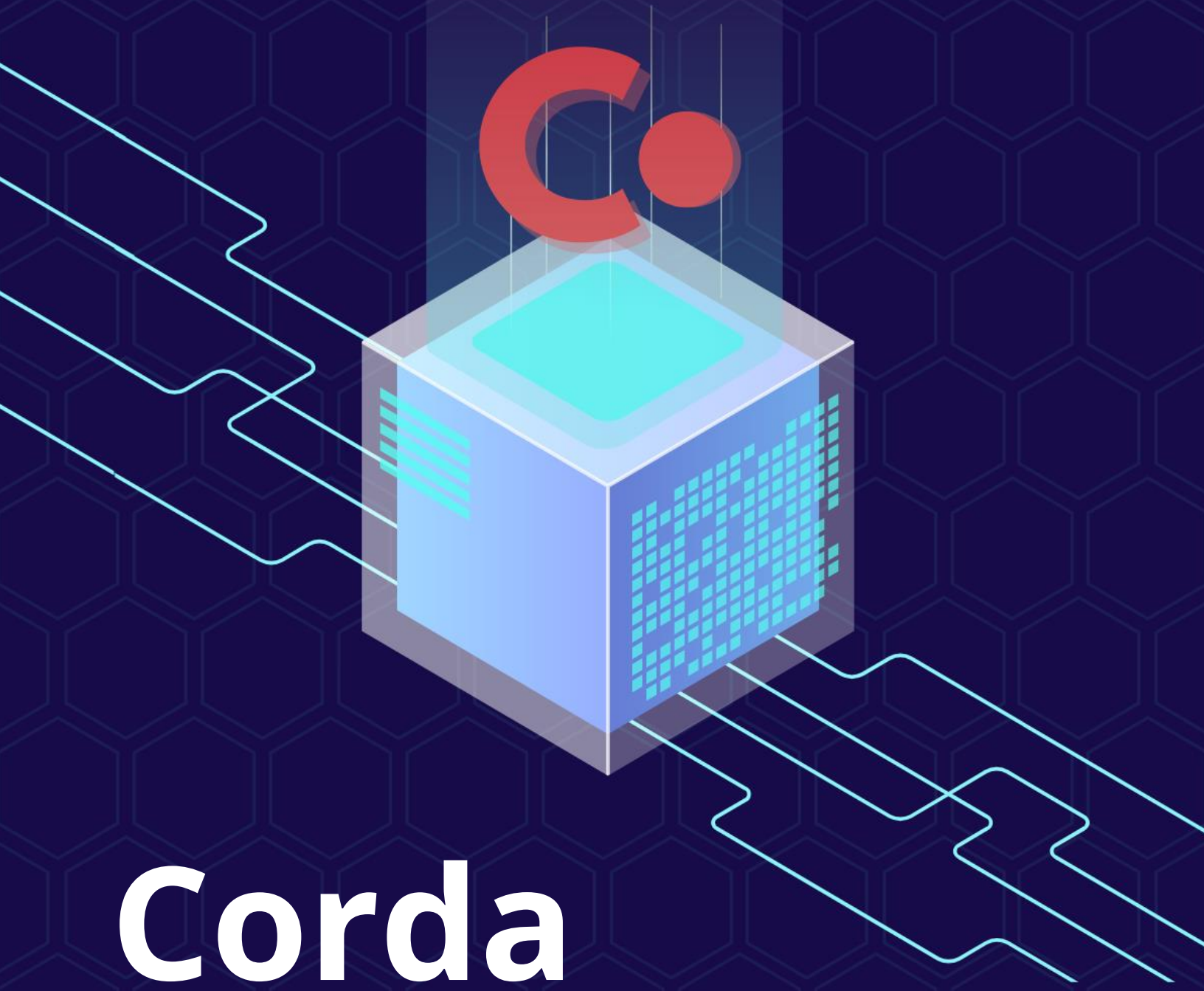
101 Blockchains



Corda

101 Blockchains Flashcards

Learn the concepts of Corda blockchain with the Corda flashcards, and get ready to start your career as a Corda professional.

1. Corda

Corda is basically a distributed ledger technology that is permissioned and equipped with a workflow messaging network. The primary objective of Corda was aimed at helping regulated financial institutions with interoperability in enterprise settings. Corda can ensure reliable levels of scalability, state consistency, transaction privacy, as well as workflow flexibility for different enterprise use cases such as trade finance, healthcare, capital markets, supply chain, insurance, telecommunications, and governance.

2. Permissioned

Corda network is permissioned in nature as it follows a precise network admission process. A node serves as the representation of any actor on the Corda network, with nodes having familiarity among them. The permissioned nature of Corda also ensures system design for rejecting any unauthorized nodes, thereby leading to more focus on accountability. The permissioned trait in Corda is helpful for regulated enterprises in following their strict security requirements.

3. Notaries

Corda networks feature nodes that are associated with specific roles, and notaries are one of the prominent examples of such nodes. Notaries are quite significant requirements for addressing a uniquely distinct function in the process of finalizing transactions on Corda. Corda network topologies generally include one node at least for every participating organization alongside one Notary Service node. The Notary works as a fault-tolerant service and a trustworthy neutral party.

4. Execution Logic

Corda does not function as a virtual machine and is tailored for recording, management, and synchronization of facts shared by participants. CorDapps, the decentralized applications built on Corda, mirror business processes accurately by starting with binding agreements. Contracts could implement legal agreements in the case of Corda, and it helps in executing contracts through easier implementation of business processes such as a collection of necessary signatures and approvals for the agreements.

5. CorDapps

CorDapps, or Corda Distributed Applications, are clearly self-explanatory as they are simply the distributed applications running on the Corda platform in a set of JAR files. The primary objective of CorDapp points out to enabling nodes to reach an agreement regarding updates in the ledger. CorDapps are able to achieve this objective with a clear definition of flows which the Corda node owners could choose to invoke over RPC.

6. Network Structure

The network structure provides the definition for organizing a Corda network. Inherently, the Corda network is basically an authenticated peer-to-peer network of nodes. Each node in the network is actually a JVM run-time environment. The JVM run-time environment is responsible for hosting Corda services alongside ensuring the execution of CorDapps. The Corda network follows direct communication among nodes with the transmission of TLS-encrypted messages without the need for any global broadcasts.

7. Doorman

Corda networks are semi-private in nature, and every Corda network features a doorman service for enforcing certain rules. The doorman service in the Corda network helps in enforcing the rules about information that nodes should provide. The doorman service also determines the know-your-customer processes required for admission to the Corda network. Nodes should contact the doorman and enter the required information. Eligible nodes can get a root-authority-signed TLS certificate for identity.

8. Network Services

Nodes in the Corda network could offer different types of services. The first type of network service refers to notary services. Corda network can offer one or multiple pluggable notary services. Notaries could ensure the uniqueness of ledger updates alongside checking the possibility of their validity. Every notary service could run on a particular node or throughout a node cluster. Corda nodes can also offer zero or multiple oracle services.

9. Corda Ledger

The Corda ledger is unique in the fact that it is subjective from the perspective of each peer. Two peers in the Corda network always have the assurance of witnessing and understanding exactly the same version of on-ledger facts shared by them. Corda does not rely on a single, centralized store of data, and it helps in maintaining a separate database of known facts with a visible subset of facts only.

10. Identities

Identities in the Corda landscape could point towards the representation of the legal identity of an organization or the service identity for a network service. Legal identities are useful for parties in a transaction, and service identities are helpful for offering transaction-related services. Service identities are different from legal identities to ensure the existence of distributed services on nodes under the ownership of different organizations. Distributed service identities rely on CompositeKeys for signer descriptions.

11. Certificates

Nodes in the Corda network could verify the identity of the owner for a public key with the help of X.509 certificates. Upon running for the first time, a node could generate a key pair followed by submission of a certificate for signing request to the doorman service of the network. The doorman service is applicable for relevant identity checks followed by issuing a certificate to the node as a node certificate authority.

12. Confidential Identities

Identities could be found in two categories such as well known and confidential identities. These two categories depend on the publication of their X.509 certificate. Well-known identities are the commonly identifiable public key for a legal entity or service. Confidential identities are restricted only to the participants in transactions related to the concerned entity. Confidential entities could involve exposure of public keys to third parties, albeit with limited certificate distribution.

13. States

The state in the case of Corda refers to an immutable object providing representation for a fact that is known to one or multiple Corda nodes at a specific instance in time. The states in Corda contain arbitrary data, thereby enabling the representation of any types of facts such as KYC data, stocks, identity information, loans, and others. Other than the fact, the state also features a reference to contracts governing states.

14. State Sequences

The states in Corda are immutable, and it is difficult to modify them directly according to a change in the world's state. On the contrary, a state sequence helps in the representation of the lifecycle of a shared fact over the course of time. At the time of updating a state, users can create a new version of the same with the representation of a new world state while marking the previous as historic.

15. Vault

The vault is also a critical component in any Corda implementation. As a matter of fact, every node on the network maintains a vault. It is basically a database for a node to track all the current and historic states it knows about. It also helps in finding out the states that it perceives as relevant to its own cause. The vault can ensure tracking of consumed and unconsumed states.

16. Soft Locking

Soft locking is an important procedure applied in the vault for attempting and preventing a node from developing transactions attempting to use similar inputs at a time. Soft locking can help in preventing loss of work when such transactions are flagged as double-spend attempts. The automatic application of soft locks to coin selection ensures that transactions do not attempt to spend similar fungible states, thereby resulting in an InsufficientBalanceException.

17. Transactions

Transactions in Corda basically refer to the proposals aimed at updating the ledger. Corda depends on an unspent transaction output (UTXO) model, which implies immutability for every state on the ledger. The ledger goes through evolution over the course of time through the application of transactions. Transactions help in updating the ledger by marking existing ledger states as inputs. Transactions also serve as the representation of a single link in the state sequences.

18. Transaction Chains

The proposer of the transaction must create the output states proposed by the transaction as they do not exist at the time of creating a new transaction. On the other hand, input states are already available as outputs of previous transactions. So, the input states are included in a proposed transaction through reference. The input states references include a hash of transaction, which creates the input and input index in previous transaction outputs.

19. Transaction Validity

Transaction validity is one of the important conditions that every required signer should verify before signing a transaction. Transaction validity is proved only when the proposed transaction, and all transactions in the chain of transactions responsible for creating the inputs of the currently proposed transaction, fulfill certain conditions. One of the conditions implies the contractual validity of a transaction. The other condition for transaction validity is the digital signatures of required parties.

20. Reference States

Contracts related to other input or output states must refer to some states without updating or consuming them. Reference states are evident through the addition of a state to the references list of a transaction rather than in the inputs or outputs list. Reference states are different from regular states as the contracts for reference states are not subject to execution for relevant transactions and are not consumed by committing transactions to the ledger.

21. Commands

Commands are an important part of transactions in Corda for implementing desired actions and rules. They are useful for indicating the intent of the transaction. Commands have a profound influence on the ways for checking the validity of the transaction. Every command features a list of one or multiple signers. The union of all public keys listed in commands can help in obtaining the list of required signers for a transaction.

22. Attachments

Many applications involve massive chunks of data which is reusable throughout various transactions, such as a table of currency codes or a public holiday calendar. Attachments help in reusing a large piece of data. Every transaction could refer to none or multiple attachments through hash. Attachments are basically ZIP or JAR files with arbitrary content. The information highlighted in the attachment files can be highly crucial for verifying transaction validity precisely.

23. Time-window

Corda applications also involve requirements of approval for a proposed transaction in a specific time window. For example, a contract option should be implemented after a specific date. The addition of a time window to the transaction can help in addressing such requirements. Time windows help in specification of the time period in which a particular transaction has to be executed. The notary serves as the timestamping authority and can refuse to commit ineligible transactions.

24. Contract Validity

Digital signature of all required signers is important for validity of a transaction. However, contract validity is also important for establishing transaction validity. Contract validity is applicable when every transaction state offers specifications for a contract type. The contract must take a specific transaction and input followed by stating validity of transaction according to contract rules. Transaction validity is proven only with approval of the contract of input and output state.

25. Contract Sandbox

Contract sandbox principle suggests the verification of transactions should always follow a deterministic approach. In this approach, a contract must always accept or reject a particular transaction. Contract sandbox is essential for ensuring that all network peers reach a consensus regarding transaction validity. The sandbox includes a whitelist that prevents import of libraries by the contract that can lead to non-determinism. Subsequently, the contract could access the transaction information only.

26. Flow Framework

Corda depends on point-to-point messaging rather than choosing a global broadcast. Network participants should specify the information to be sent, the receiving parties, and the order of sending information. Flows help in automating all these processes without specifying them manually. A flow basically refers to the sequence of steps providing instructions to a node about achieving a particular ledger update task, like settling a trade or issuing of an asset.

27. Inter-node Communication

Nodes can communicate with each other through messages between the flows. Every node feature zero or multiple flow classes registered for responding to messages via another single flow. After encapsulation of a business process in a flow, all the communication between the nodes happens according to its rules. Inter-node communications start with a flow in which the user is registered to respond to, and receive messages within the context of flow.

28. Flow Library

It is possible to craft flows by beginning a flow as a subprocess with reference to another flow, thereby creating a sub-flow. Corda presents a distinct library of flows that helps in management of common tasks. Therefore, developers don't have to work on redefining the logic underlying common processes. The common processes for which flows are available include verification of transaction chain, collecting signatures from counterparty nodes, or transaction notarizing.

29. Concurrency

Concurrency is a critical aspect of flows in Corda. The flow framework in Corda enables nodes to have many active flows simultaneously. The flows could last for many dates throughout different node restarts and upgrades. Concurrency is possible through serialization of flows to disks upon entering a blocking state. In this case, the node doesn't wait to unblock the flow. It immediately starts working on other scheduled flows before the rollback.

101 Blockchains

101 Blockchains is the world's leading research-based platform built for Enterprise Blockchain Professionals, with a thriving community of over 25,000 professionals.
101 Blockchains offers world-class training courses and industry-recognized certification programs that are helping professionals all over the world upgrade their skills and accelerate their career growth.

Check out our collection of 101 Blockchains Flashcards. Gain further knowledge about blockchain technology with in-depth guides and blogs. Find world-class professional training courses.