Build a Shared Record-Keeping System

FinTech

Lesson 18.2

Class Objectives

By the end of this lesson, you will be able to:

- Describe what cryptography is and how it’s used to secure a blockchain.
- Explain how cryptography is used to convert data into a hash code.
- Use Python to define a `Block` class data structure that includes attributes and methods.
- Explain the role of hash codes in securing the integrity of record data on the blockchain.
- Use hash codes to link blocks together in order to create a blockchain.

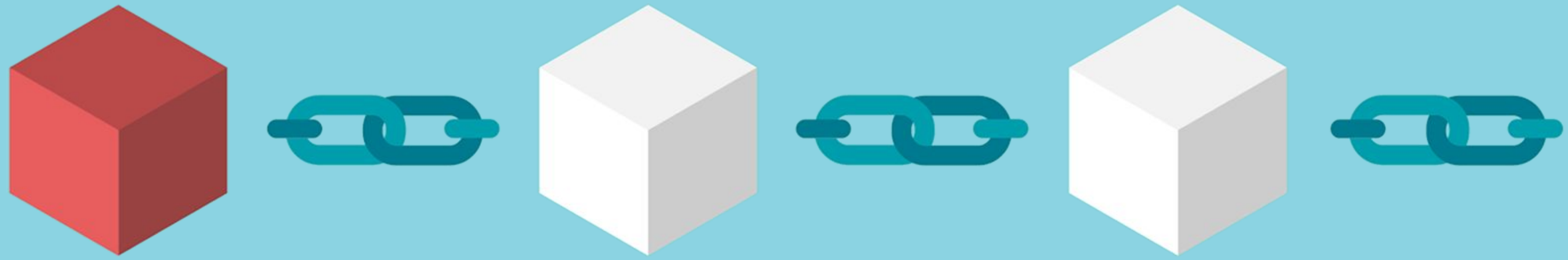
Data Breaches

70% According to a recent study by Verizon, 70% of all the data breaches in the world in 2020 were financially motivated.

43% And 43% of the breaches targeted vulnerabilities in web applications, which is more than double the reported amount from the previous year. This highlights the problem of securing online financial systems.

Data Breaches

To combat malicious hackers and keep data secure, blockchain systems combine clever mathematical concepts, called **cryptography**, with a collection of formalized data that's known as a **block.**



First Block

Hash: `12f13q`

Previous Hash: `000000`

Second Block

Hash: `d340kc`

Previous Hash: `12f13q`

Third Block

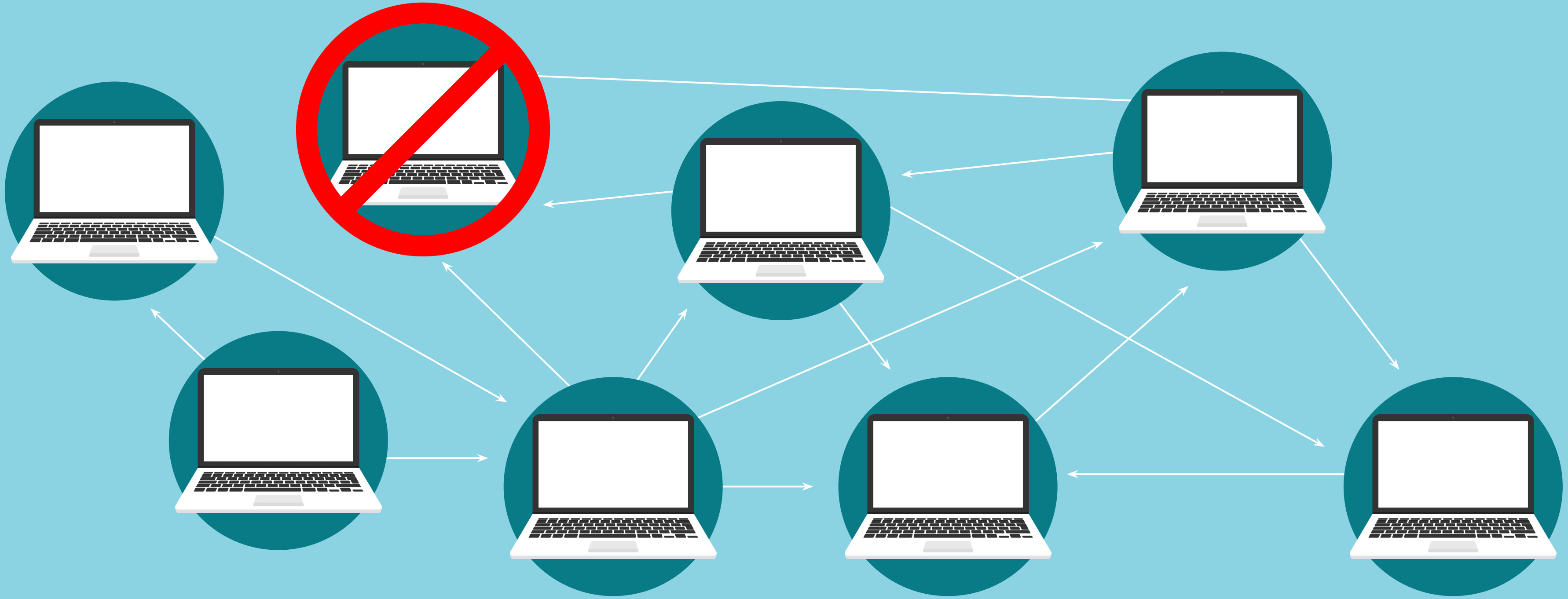
Hash: `1m12ep`

Previous Hash: `d340kc`

The blocks link together to form a chain of trust, or integrity, which is known as a blockchain.

Data Breaches

This blockchain data structure is unique, because the data that's written to it becomes instantly verifiable by design. This means that if someone changes or tampers with the data in the ledger, the users of the ledger will know about it.



Blockchain Recap

What are the five key features of a blockchain?

Five Key Features of a Blockchain

Decentralization	Users have direct access to the blockchain and can edit it simultaneously. Transactions are not monitored by a central authority.
A distributed architecture	Many computers in various locations store identical copies of the same ledger. These computers communicate with each other to arrive at particular decisions, like the validity of a new block in the chain.
Trust	Blockchain technology is designed so that users can trust that the blockchain accurately records all its data and prevents tampering with that data.
Record keeping	Each block represents a transaction. The chain links these transactions over time.
Transparency	Anyone can review the history of the transactions in a blockchain, such as who added the data and when.

What is the difference between a permissioned and a permissionless blockchain?

Permissioned vs. Permissionless Blockchains

Permissioned

A permissioned blockchain has a trusted, third-party arbiter—for example, a government, corporate CEO or Board of Directors, or another well-respected institution—acting as the central decision-making authority.

Permissionless

A permissionless blockchain does not have a central authority to provide trust. Instead, people place their trust in the pre-specified rules of the blockchain, which are the incentives that keep the users acting appropriately.


Questions?

Shared Accounting with Digital Ledgers

Distributed Ledger

A blockchain is a type of shared record-keeping system that seeks to maintain trust through the use of a **distributed ledger**.

First Block
Distributed Ledger

	A	B	C	D
1	XC14ADF	T45W46	A563537	D546563

Second Block
Distributed Ledger

	A	B	C	D
1	XC14ADF	T45W46	A563537	D546563
2	N579689	J788567	Q4W4W6	6785795

Third Block
Distributed Ledger

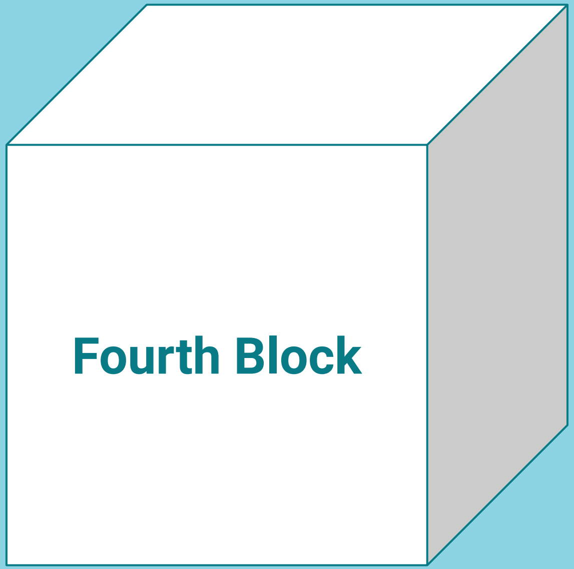
	A	B	C	D
1	XC14ADF	T45W46	A563537	D546563
2	N579689	J788567	Q4W4W6	6785795
3	SFG5354	7578576	GHD784	L787656

Fourth Block
Distributed Ledger

	A	B	C	D
1	XC14ADF	T45W46	A563537	D546563
2	N579689	J788567	Q4W4W6	6785795
3	SFG5354	7578576	GHD784	L787656
4	SHG8757	GHDG746	5664QR6	ADFAFDA

Block

A **block** is the most fundamental data structure of the blockchain.
It's essentially a container that holds data.



Fourth Block
Distributed Ledger

	A	B	C	D
1	XC14ADF	T45W46	A563537	D546563
2	N579689	J788567	Q4W4W6	6785795
3	SFG5354	7578576	GHD7845	L7876561
4	SHG8757	GHDG746	5664QR6	ADFAFDA

Distributed Ledger

In a blockchain, new entries can be added only to the end of the ledger.
That is, a new block can be added only to the end of the chain.

First Block
Distributed Ledger

	A	B	C	D
1	XC14ADF	T45W46	A563537	D546563

Second Block
Distributed Ledger

	A	B	C	D
1	XC14ADF	T45W46	A563537	D546563
2	N579689	J788567	Q4W4W6	6785795

Third Block
Distributed Ledger

	A	B	C	D
1	XC14ADF	T45W46	A563537	D546563
2	N579689	J788567	Q4W4W6	6785795
3	SFG5354	7578576	GHD784	L787656

Fourth Block
Distributed Ledger

	A	B	C	D
1	XC14ADF	T45W46	A563537	D546563
2	N579689	J788567	Q4W4W6	6785795
3	SFG5354	7578576	GHD784	L787656
4	SHG8757	GHDG746	5664QR6	ADFAFDA

Different blockchains store different types of data records inside their blocks. However, almost every chain can store transaction data.

Defining a Blockchain Block with a Python Data Class

A **class** offers a way to define a custom data structure in Python.

The class can store multiple variables, known as **attributes** when they exist in a class.

Python Data Class

Python has a special type of class, called a `dataclass`, that can be used to define and store data.

```
# Imports
 from dataclasses import dataclass

 # Creating the Block data class
 @dataclass
 class Block:
     data: str
```

A Python data class is like a blueprint for a data container. It defines the structure and provides hints about the types of data that belong in it. This is called **typing**.

Instructor Demonstration

Python Data Class

Activity: Build a Blockchain Block

In this activity, you will build a Streamlit application that accepts user input and then stores that input in a `Block` data class.

Suggested Time:

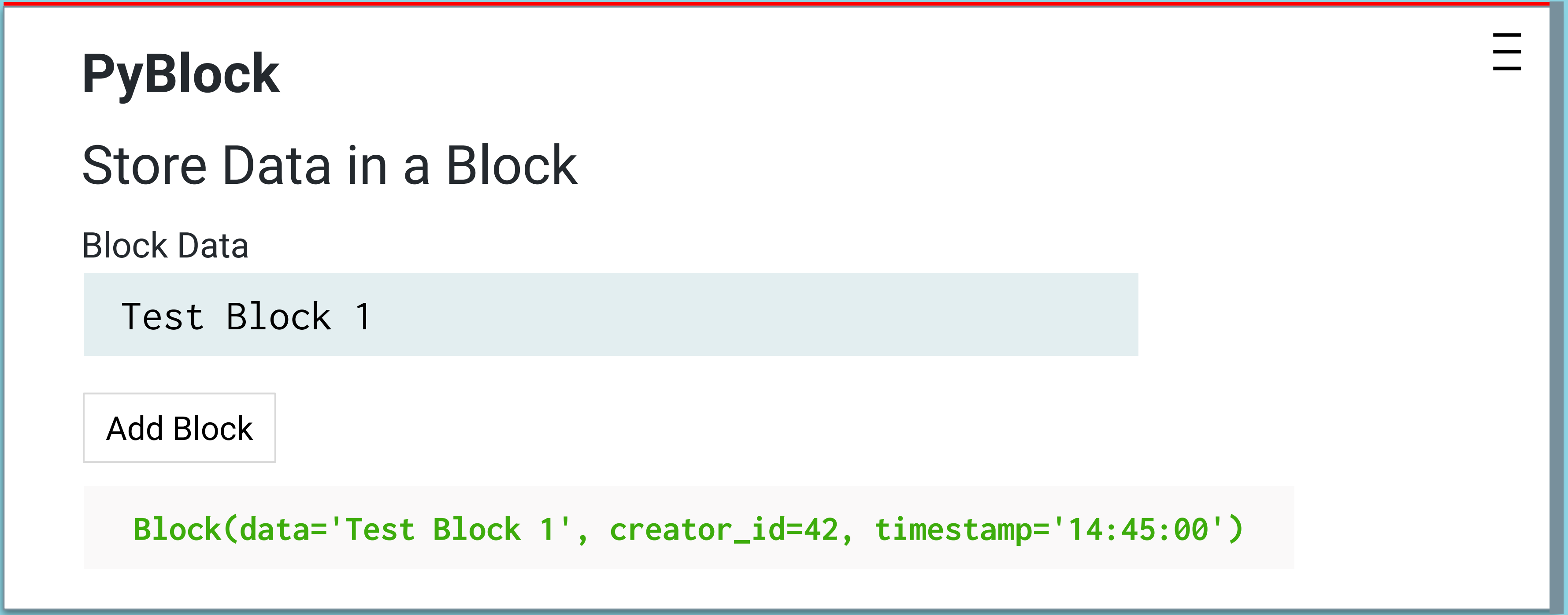
15 minutes

Time's Up! Let's Review.

Improving Data Integrity with Hashing

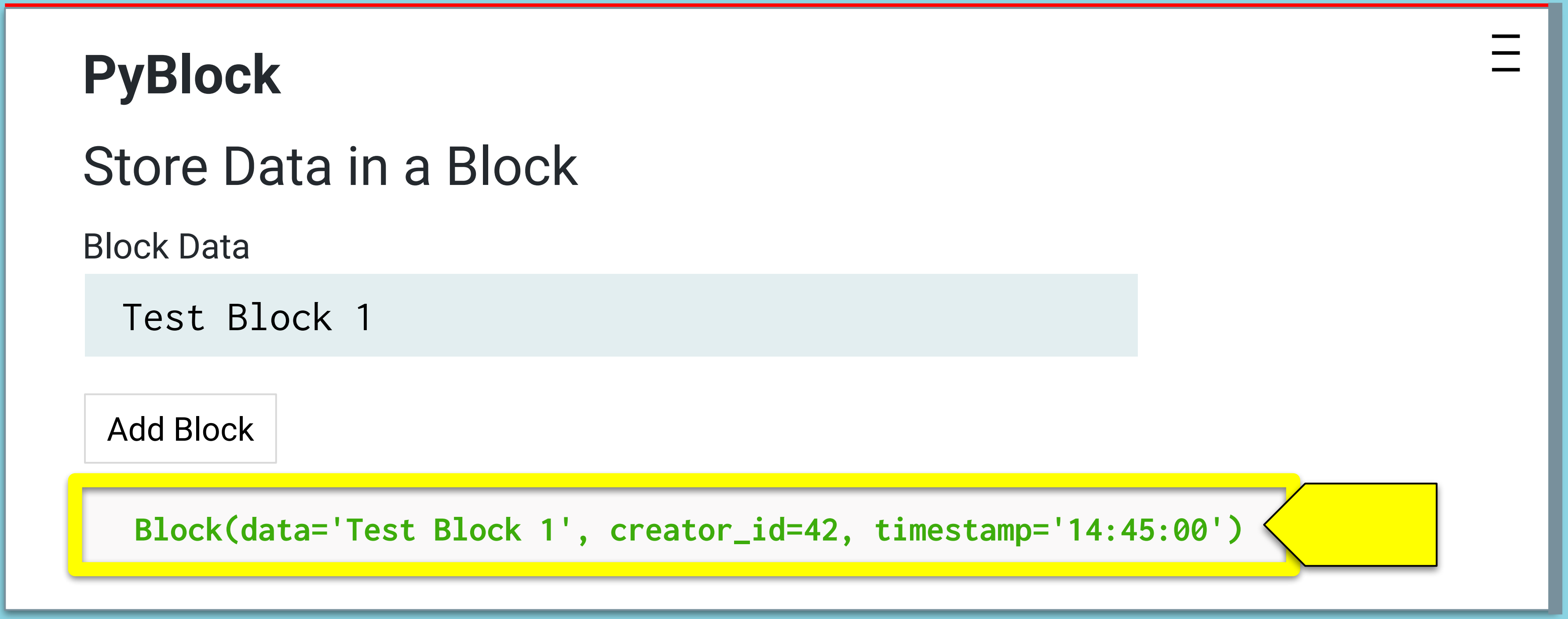
Python Data Class

For the instance of the `Block` class we just created, it's easy to verify that no one else has changed the data—we simply check the information in the `data` attribute.



Python Data Class

We can determine if someone has changed the string simply by looking at it.
But imagine a block with a data attribute that contains an entire novel.



Manually validating all of that data would take a lot of time and energy!

In these cases, we need a better approach. This is where hashing comes in.

Hashing

Hashing is a cryptography technique that takes a piece of input data and then outputs a fixed-length, mathematical representation of that data—the data hash.

A **hash** hides the input and gives you an output that represents the data.

First Block

Hash: `12f13q`

Second Block

Hash: `d340kc`

Third Block

Hash: `1m12ep`

Hashing

Hashing functions have three key properties:

01

When given a piece of unique data, a hashing function **always returns the same unique hash** for that piece of data. In other words, the same input will always return the same output.

02

Hashing functions return **fixed-length** hashes. That is, all hashes returned by the same function have the same length.

03

Hashing functions are always **one-way** functions—you can’t determine the input (the original piece of data) by analyzing the output (the hash).

One-Way Hashing Function

Hashing functions are one-way functions. The data is translated via the hashing function into the hash code. It's not possible to translate a hash code back into the original pieces of data. This makes hashing especially useful for securing private data, such as financial or medical records.

One-Way Hashing Function

Input	→	3672854bsb238cbl9932b.....
???	~~←~~	3672854bsb238cbl9932b.....

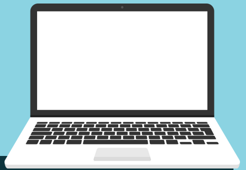
Hashing is a fundamental component of a blockchain.

One-Way Hashing Function

We can verify whether the data in a block has changed by checking the hash for the block. This is because the same input data always returns the same hash. Even slightly different input data returns a completely different hash.

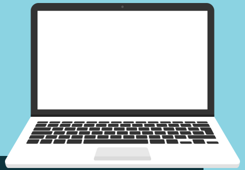
Input	→	3672854bsb238cbl9932b.....
???	~~←~~	3672854bsb238cbl9932b.....

First Block



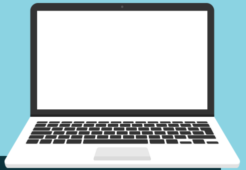
Hash: `12f13q`

Second Block



Hash: `d340kc`

Third Block



Hash: `1m12ep`

Instructor Demonstration

Use hashlib

Activity: Hashing with Hashlib

In this activity, you will use the hashlib library and Streamlit to build an application that can hash any text input.

Suggested Time:

15 minutes

Time’s Up! Let’s Review.


Break

Instructor Demonstration

Add a Method to a Data Class

Activity: Hashing a Block

In this activity, you will create a Streamlit application that can create a new block of data and display the hash for that block.

Suggested Time:

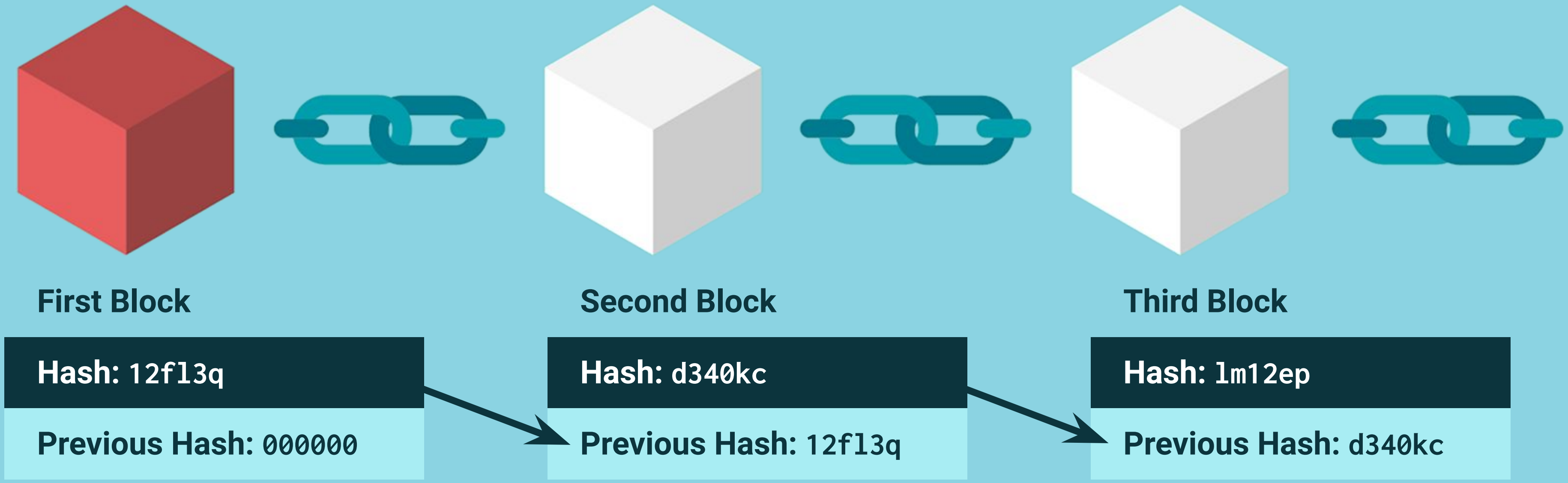
15 minutes

Time's Up! Let's Review.

Chaining Blocks

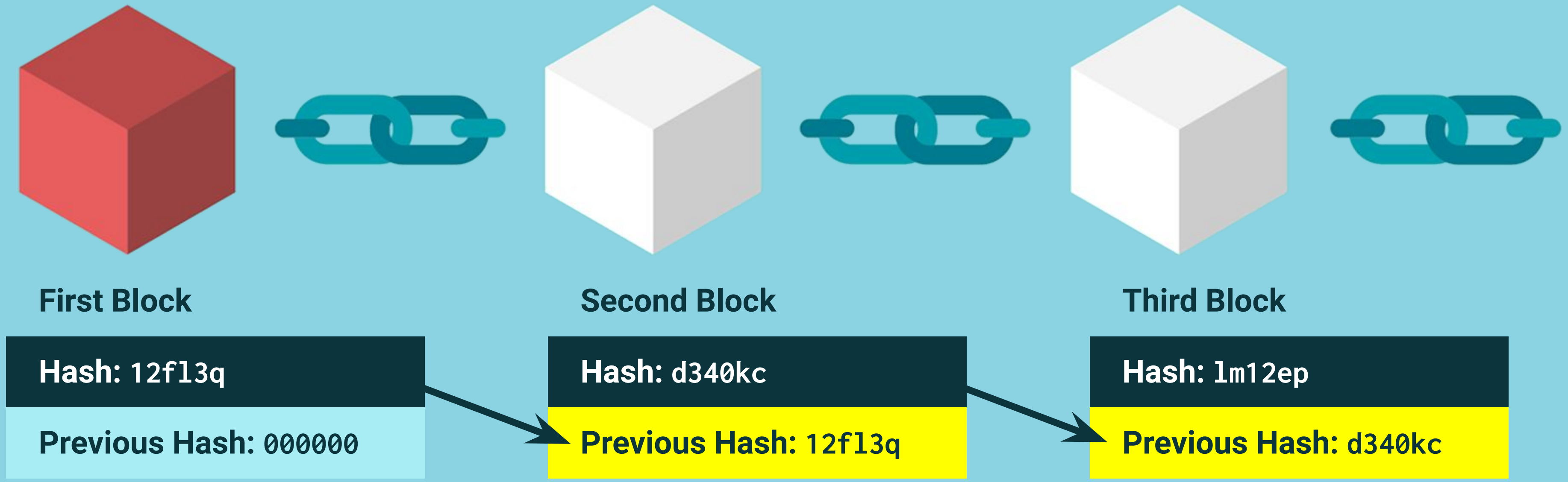
Chaining Blocks

The software program that underlies the blockchain hashes the data when creating the block, as well as links the blocks to each other. When the software creates a new block, it links the new block to the previous block.



Chaining Blocks

It does this by adding the hash of the previous block to the data of the new block.
The hash of the previous block becomes part of the data in the new block.



The purpose of linking blocks by hash code is that if someone changes the data in the previous block, the new block will know.

This is because the hash of the previous block will change, and the new block already knows what that hash should be.

The Chaining Process

Let's say that we have a blockchain of 10 blocks. Someone changes the data in Block 5, which also changes the hash of Block 5.

Block 1	Block 2	Block 3	Block 4	Block 5	Block 6	Block 7	Block 8	Block 9	Block 10
Hash: 12fl3q	Hash: d340kc	Hash: lm12ep	Hash: 37c44c	Hash: 56h6k8	Hash: 90v22x	Hash: 7h411z	Hash: b2j355	Hash: f25l2b	Hash: 78d23c
Previous Hash: 000000	Previous Hash: 12fl3q	Previous Hash: d340kc	Previous Hash: lm12ep	Previous Hash: 37c44c	Previous Hash: 56h6k8	Previous Hash: 90v22x	Previous Hash: 7h411z	Previous Hash: b2j355	Previous Hash: f25l2b

The Chaining Process

Because the data in each block includes the hash of the previous block, the data in Block 6 includes the original hash of Block 5. When the hash of Block 5 changes, the hash of Block 6 becomes invalid.

Block 1	Block 2	Block 3	Block 4	Block 5	Block 6	Block 7	Block 8	Block 9	Block 10
Hash: 12fl3q	Hash: d340kc	Hash: lm12ep	Hash: 37c44c	Hash: 56h6k8	Hash: [illegible] Invalid	Hash: 7h411z	Hash: b2j355	Hash: f25l2b	Hash: 78d23c
Previous Hash: 000000	Previous Hash: 12fl3q	Previous Hash: d340kc	Previous Hash: lm12ep	Previous Hash: 37c44c	Previous Hash: 56h6k8	Previous Hash: 90v22x	Previous Hash: 7h411z	Previous Hash: b2j355	Previous Hash: f25l2b

The Chaining Process

This, in turn, invalidates the hash of Block 7, and so on throughout the chain. So, if a malicious hacker tries to change either the data in any block or the sequence of the blocks, the rest of the chain becomes invalid.

Block 1	Block 2	Block 3	Block 4	Block 5	Block 6	Block 7	Block 8	Block 9	Block 10
Hash: `12fl3q`	Hash: `d340kc`	Hash: `lm12ep`	Hash: `37c44c`	Hash: `56h6k8`	Hash: [illegible] Invalid	Hash: [illegible] Invalid	Hash: `b2j355`	Hash: `f25l2b`	Hash: `78d23c`
Previous Hash: `000000`	Previous Hash: `12fl3q`	Previous Hash: `d340kc`	Previous Hash: `lm12ep`	Previous Hash: `37c44c`	Previous Hash: `56h6k8`	Previous Hash: [illegible] Invalid	Previous Hash: `7h411z`	Previous Hash: `b2j355`	Previous Hash: `f25l2b`

The Chaining Process

The chaining process in review:

01

The chaining process allows all users of the blockchain to validate the records in the chain over time.

02

Users can thus trust the integrity and order of the data in the chain.

03

This capability of blockchain—maintaining data integrity without a central authority—is one of the most exciting innovations of this technology.

Instructor Demonstration

The Chaining Process

Activity: Blockchain Application

In this activity, you will enhance a Streamlit application that generates new blocks of user data and adds the blocks to a Python blockchain.

Suggested Time:

15 minutes

Time’s Up! Let’s Review.

The End