Lecture slides - Week 9

OOP - Statics and Associations

Dr. Aamir Akbar

Director of both AWKUM AI Lab and AWKUM Robotics, Final Year Projects (FYPs) coordinator, and lecturer at the department of Computer Science
Abdul Wali Khan University, Mardan (AWKUM)

Contents

Statics

Static Attributes

A static attribute refers to a variable or property that belongs to a class itself rather than to its instances or objects. A static attribute are defined within a class but outside of any methods and it is accessed using the class name.

For Example:

```python
class Customer:
    id = 0   # static attribute

    def __init__(self, name):
        self.name = name
```

Static Methods

A static method belongs to the class itself rather than to instances of the class. A static method can be called on the class without requiring an instance to be created.

Activity: Modify the Customer class so that every customer will be assigned a unique account number.

Static Methods

A static method belongs to the class itself rather than to instances of the class. A static method can be called on the class without requiring an instance to be created.

Activity: Modify the Customer class so that every customer will be assigned a unique account number.

```python
class Customer:

    __id = 0

    def __init__(self, name):
        self.name = name
        self.account_num = Customer.__create_account_num()

    def __create_account_num():
        Customer.__id += 1
        return f"customer-{Customer.__id}"
```

Association

Association or Has-a relationship

In OOP, the `association or has-a` relationship refers to a class having an object of another class as one of its attributes.

- A Car `has an` Engine.
- A University `has a` Department.
- A House `has a` Room.
- A Smartphone `has a` Camera.
- A Library `has a` Book.

The above examples mean that (1) a Car class having an Engine object; (2) A University class having a Department object; (3) A House class having a Room object; (4) A Smartphone class having a Camera object; (5) A Library class having a Book object.

Association or Has-a relationship

In OOP, the `association or has-a` relationship refers to a class having an object of another class as one of its attributes.

- A Car `has an` Engine.
- A University `has a` Department.
- A House `has a` Room.
- A Smartphone `has a` Camera.
- A Library `has a` Book.

The above examples mean that (1) a Car class having an Engine object; (2) A University class having a Department object; (3) A House class having a Room object; (4) A Smartphone class having a Camera object; (5) A Library class having a Book object.

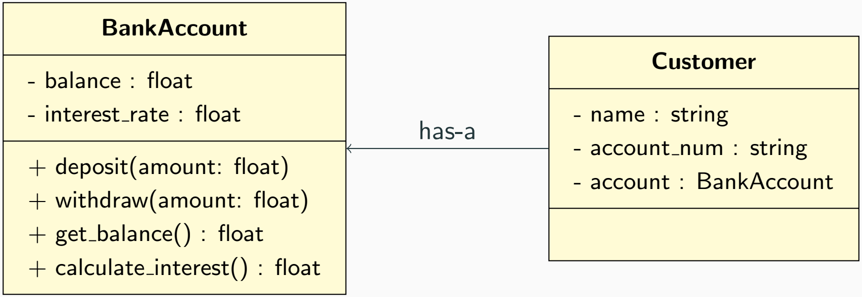
Activity: A bank customer has an account. Modify the class accordingly.

Customer has a BankAccount

```python
from bankaccount import BankAccount

class Customer:

    __id = 0

    def __init__(self, name, amount, int_rate):
        self.name = name
        self.account_num = Customer.__create_account_num()
        self.account = BankAccount(amount, int_rate)

    def __create_account_num():
        Customer.__id += 1
        return f"customer-{Customer.__id}"
```

How will the UML class diagrams look like?

UML Class Diagrams: BankAccount and Customer



Case Study: Blockchain

What is Blockchain Technology?

Blockchain is a decentralized, distributed ledger system that enables secure and transparent recording of transactions across a network of computers. It was initially created as the underlying technology for Bitcoin, but its potential applications extend far beyond cryptocurrencies.

Use Cases: Beyond cryptocurrencies, blockchain technology has applications in various industries, including:

- supply chain management
- healthcare
- finance
- voting systems
- real estate

Blockchain allows for transparent and secure record-keeping, reducing fraud, increasing efficiency, and enabling new business models.

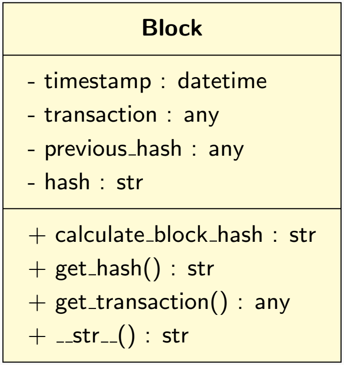
What does the block represent? i

$$\begin{tikzcd}\boxed{\text{Block 1}} \arrow[r] & \boxed{\text{Block 2}} \arrow[r] & \boxed{\text{Block 3}} \arrow[r] & \boxed{\text{Block 4}}\end{tikzcd}$$

- **Block 1, Block 2, Block 3, Block 4:** These blocks represent individual units of data within the blockchain. Each block typically contains transactional data, a timestamp, a reference to the previous block (except for the first block, known as the genesis block), and a cryptographic hash of itself.
- **Arrows:** The arrows between the blocks illustrate the linkage or chaining of these blocks. Each block in the chain contains a reference to the previous block's hash, creating a sequence where each block depends on the data in the preceding block, ensuring the integrity and immutability of the chain.

Class Design Activity: Create a UML class diagram of block.

What does the block represent? ii



Cryptography, hashing and SHA-256

1. Cryptography is the practice of securing communication and data by converting plain text into ciphertext using algorithms and keys.

2. Hashing is a process that converts input data of any size into a fixed-size string of characters, called a digest.

3. SHA-256 (Secure Hash Algorithm 256-bit) is a widely used cryptographic hash function. It generates a 256-bit (32-byte) hash value.

4. Cryptographic hash functions, like SHA-256, are utilized to create unique, fixed-size representations of data. In blockchain, each block contains a cryptographic hash of the previous block, linking them together. Any alteration in data would change the hash, preserving the integrity of the chain.

Cryptography, hashing and SHA-256

1. Cryptography is the practice of securing communication and data by converting plain text into ciphertext using algorithms and keys.

2. Hashing is a process that converts input data of any size into a fixed-size string of characters, called a digest.

3. SHA-256 (Secure Hash Algorithm 256-bit) is a widely used cryptographic hash function. It generates a 256-bit (32-byte) hash value.

4. Cryptographic hash functions, like SHA-256, are utilized to create unique, fixed-size representations of data. In blockchain, each block contains a cryptographic hash of the previous block, linking them together. Any alteration in data would change the hash, preserving the integrity of the chain.

Activity: Cryptography demo with an example and Block creation

Chaining the blocks - Blockchain

Chaining blocks in a blockchain refers to the cryptographic linking of individual blocks in a linear sequence. Each block contains a reference (hash) to the previous block, creating a continuous chain of data.

What when a block (or data) is altered?

This linking mechanism ensures the integrity and immutability of the entire blockchain structure. Altering any block would require recalculating the hash for all subsequent blocks, making tampering computationally infeasible and preserving the integrity of the entire chain.

Chaining the blocks - Blockchain

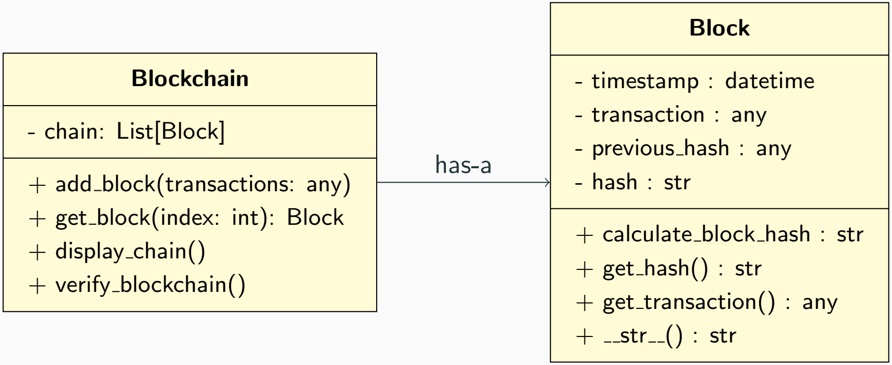
Chaining blocks in a blockchain refers to the cryptographic linking of individual blocks in a linear sequence. Each block contains a reference (hash) to the previous block, creating a continuous chain of data.

What when a block (or data) is altered?

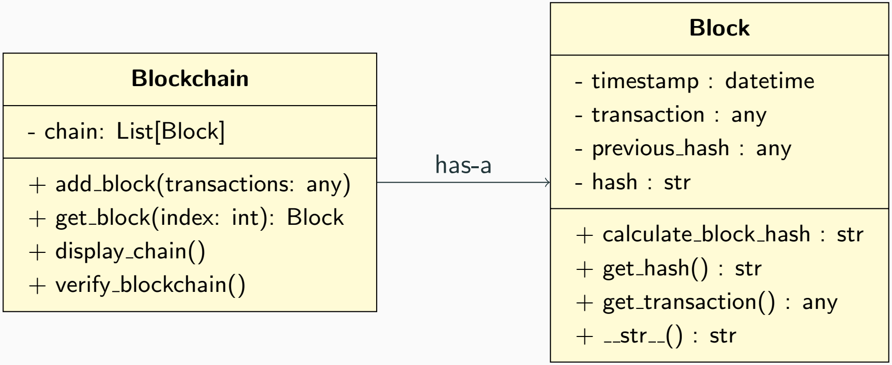
This linking mechanism ensures the integrity and immutability of the entire blockchain structure. Altering any block would require recalculating the hash for all subsequent blocks, making tampering computationally infeasible and preserving the integrity of the entire chain.

Activity: Create a UML class diagram of the blockchain class with any relationship.

Blockchain UML class diagram



Blockchain UML class diagram



Activity: Implement the Blockchain class in Python.