Aggregation in Java

Aggregation is a relationship between two classes where one class contains a reference to another class (has-a relationship). It represents a **weak association** where the child object can exist independently of the parent. If the parent class is destroyed, the child class can still exist.

Key Points:

- Represents a **"has-a"** relationship.
- The contained object (child) **can exist independently** of the container (parent).
- There is **no ownership** between the parent and child.

Example of Aggregation:

Consider a `Library` class that has a collection of `Book` objects. Even if the `Library` object is destroyed, the `Book` objects can exist independently.

```java
class Book {
    String title;
    String author;

    public Book(String title, String author) {
        this.title = title;
        this.author = author;
    }

    public void display() {
        System.out.println("Title: " + title + ", Author: " + author);
    }
}

class Library {
    private List<Book> books;

    public Library(List<Book> books) {
        this.books = books;
    }

    public void displayBooks() {
        for (Book book : books) {
            book.display();
        }
    }
}

public class Main {
    public static void main(String[] args) {
        Book book1 = new Book("Java Basics", "John Doe");
        Book book2 = new Book("Advanced Java", "Jane Smith");
        // Aggregating books into the library
        List<Book> bookList = new ArrayList<>();
        bookList.add(book1);
        bookList.add(book2);

        Library library = new Library(bookList);
        library.displayBooks();
    }
}
```

Inheritance in Java Inheritance is one of the core concepts of object-oriented programming. It is a mechanism where one class acquires the properties and behaviors (fields and methods) of another class. Inheritance allows for code reusability and polymorphism.

Key Points: Represents an "is-a" relationship. Inheritance promotes code reuse and allows the creation of hierarchical relationships between classes. The class that inherits is called the subclass or child class. The class that is inherited from is called the superclass or parent class. In Java, inheritance is achieved using the extends keyword. Types of Inheritance in Java: Single Inheritance: A class inherits from one superclass. Multilevel Inheritance: A class inherits from a subclass, forming a chain. Hierarchical Inheritance: Multiple classes inherit from the same superclass. (Java does not support Multiple Inheritance directly): A class cannot inherit from more than one class directly, but interfaces can be used to achieve multiple inheritance behavior.

```
// Parent class (Superclass)
class Animal {
    String name;

    public void eat() {
        System.out.println(name + " is eating.");
    }
}

// Child class (Subclass) inherits from Animal
class Dog extends Animal {
    public void bark() {
        System.out.println(name + " is barking.");
    }
}

public class Main {
    public static void main(String[] args) {
        Dog dog = new Dog();
        dog.name = "Rex";

        // Accessing parent class method
        dog.eat();

        // Accessing child class method
        dog.bark();
    }
}
```

Multi-level inheritance

```java
// Grandparent class
class Animal {
    public void eat() {
        System.out.println("Animal is eating.");
    }
}

// Parent class
class Dog extends Animal {
    public void bark() {
        System.out.println("Dog is barking.");
    }
}

// Child class
class Bulldog extends Dog {
    public void showStrength() {
        System.out.println("Bulldog is strong.");
    }
}

public class Main {
    public static void main(String[] args) {
        Bulldog bulldog = new Bulldog();

        // Access methods from all levels of inheritance
        bulldog.eat();       // From Animal
        bulldog.bark();      // From Dog
        bulldog.showStrength();  // From Bulldog
    }
}
```