Object Oriented Programming

Mradul Singhal

April 11, 2025

Contents

Introduction

Object-Oriented Programming (OOP) organizes software design around data, or objects, rather than functions and logic. Here are the fundamental concepts of OOP:

- **Classes:** Templates for creating objects, defining their structure and behavior.
- **Objects:** Instances of classes that represent specific elements with attributes and behaviors.
- **Encapsulation:** Hiding the internal state of an object and requiring all interaction to be performed through an object's methods.
- **Abstraction:** Exposing only the necessary and relevant parts of an object to the outside world.
- **Inheritance:** A mechanism for one class to inherit the properties and behavior of another class.
- **Polymorphism:** The ability to present the same interface for differing underlying data types.

Association, Aggregation & Composition

Association

- When an object have a relationship with another object
- **Example:**

```
class Foo {
    private Bar bar;
};
```

  `Foo` uses `Bar`

- It may be *one-to-one, one-to-many, many-to-one* or *many-to-many* relationship

Aggregation

- A relationship where the child can exist independently of the parent.
- **Example:**

```
class Foo {
    private Bar bar;
    Foo(Bar bar) {
       this.bar = bar;
    }
}
```

  When `Foo` dies, `Bar` may live on

- It may be *one-to-one, one-to-many, many-to-one* or *many-to-many* relationship

Composition

- An object owns another object and is responsible for that object's lifetime.
- **Example:**

```java
class Foo {
    private Bar bar = new Bar();
}
```

When `Foo` dies, so does `Bar`

- It may be *one-to-one* or *one-to-many* relationship