Introduction to OOP

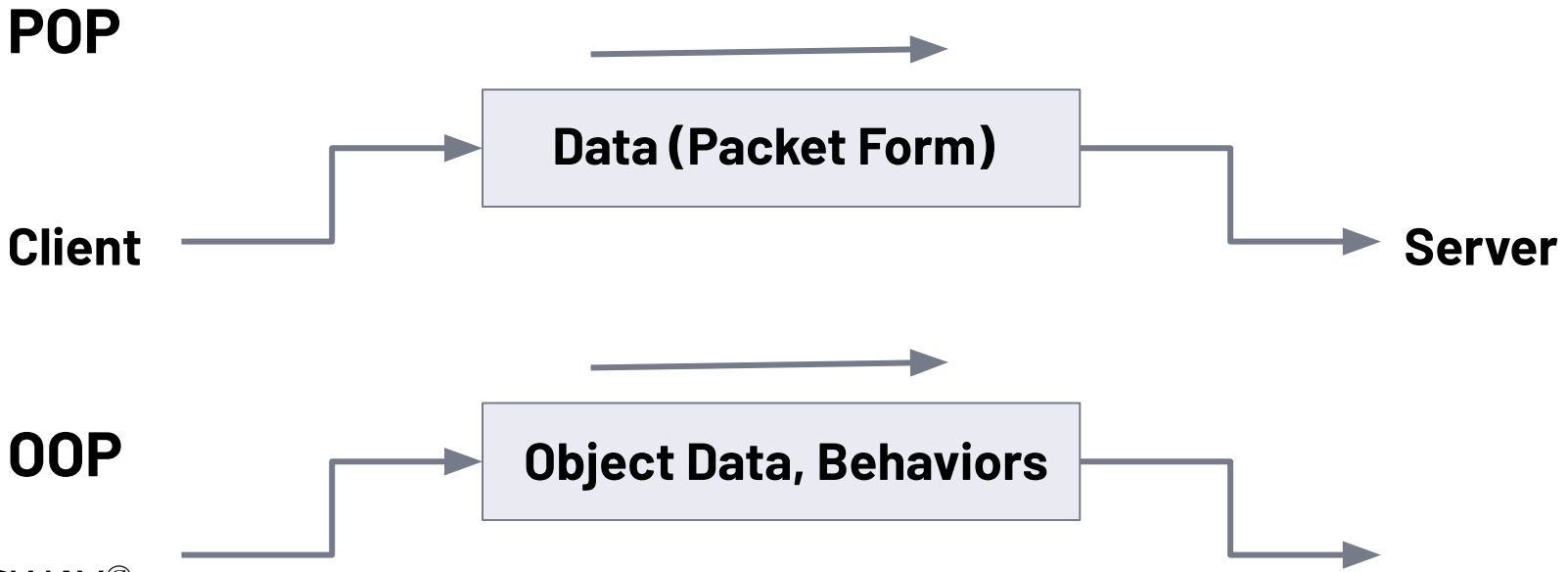
Table of Contents

Did you understand OOP ?

Students, drag the icon!

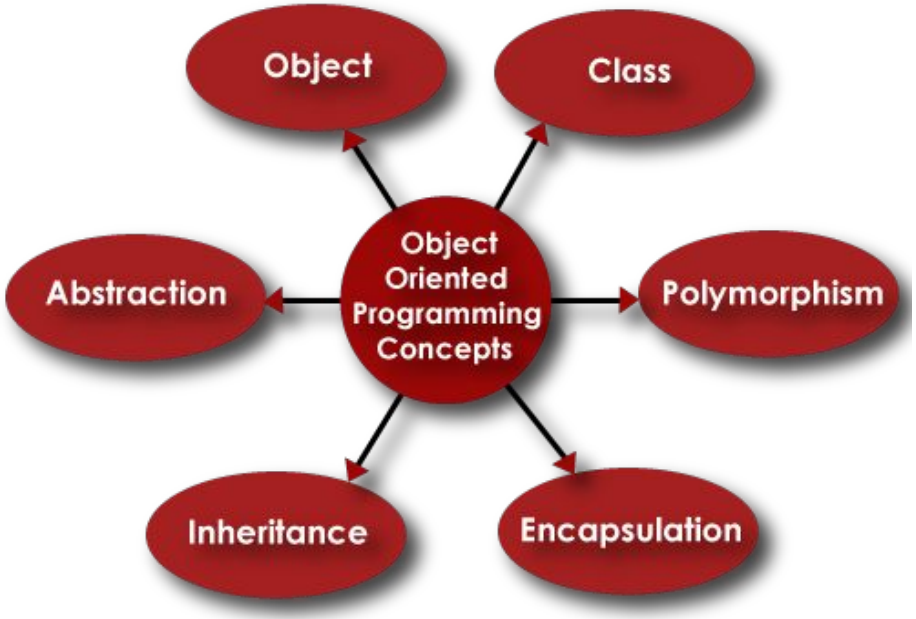
Object Oriented&Procedural Oriented

- Object Oriented Programming (**OOP**) has become popular over recent years and has completely replaced the Procedural Oriented Programming (**POP**).



1 Fundamentals of OOP

Fundamentals



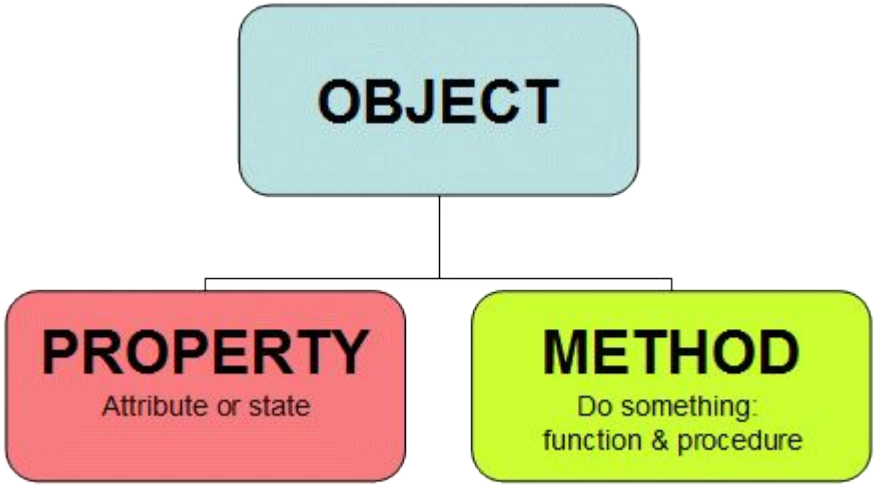
Fundamentals

- **Object-Oriented Programming (OOP)** is a programming paradigm based on the concept of **objects** that interact with each other to perform the program functions.




Fundamentals

- Each object can be characterized by a **state** and **behavior**. An object keeps the current state in the **fields** and the **behavior** in the methods.



Draw lines to match the image to the answer:
How is your pre-class level?

encapsulation

abstraction

containing information in an object

child classes inherit data and behaviors from parent class

only exposing high level public methods

many methods can do the same task

Inheritance

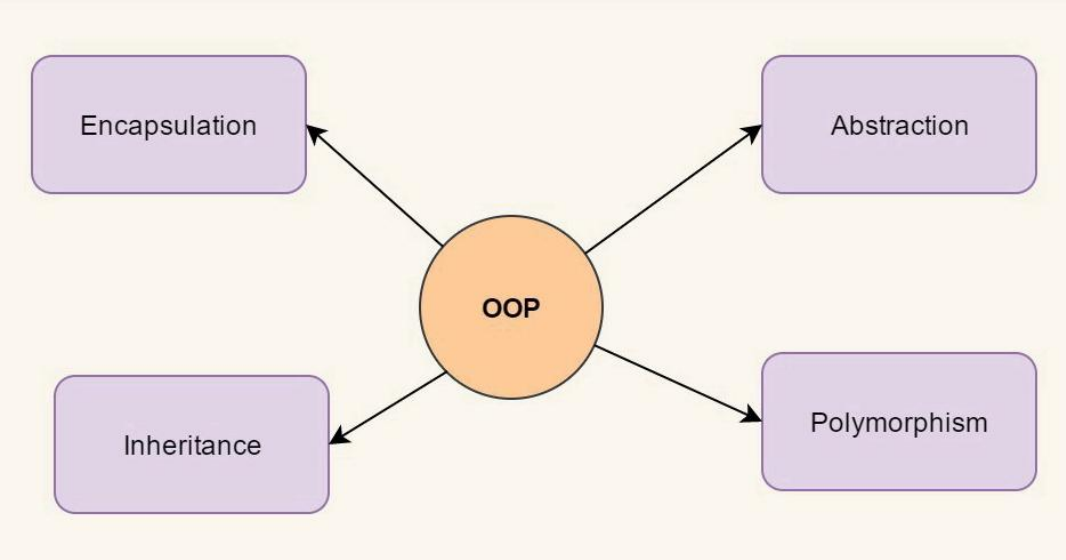
polymorphism

Students, draw anywhere on this slide!

Fundamentals

containing information in an object

child classes inherit data and behaviors from parent class



only exposing high level public methods

many methods can do the same task

2 Objects

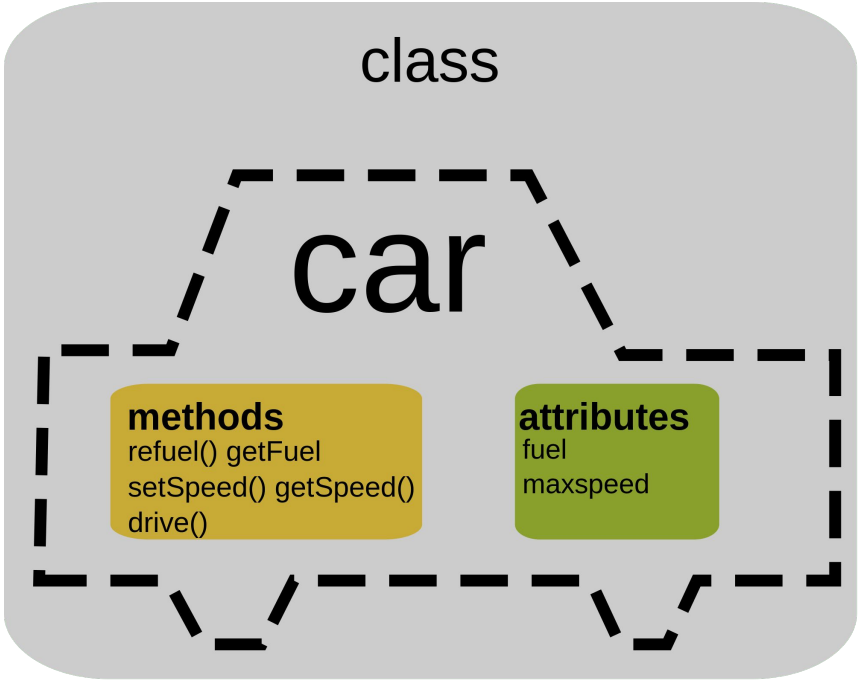
Objects

- The key notion of the OOP is, naturally, an **object**.

3 Classes

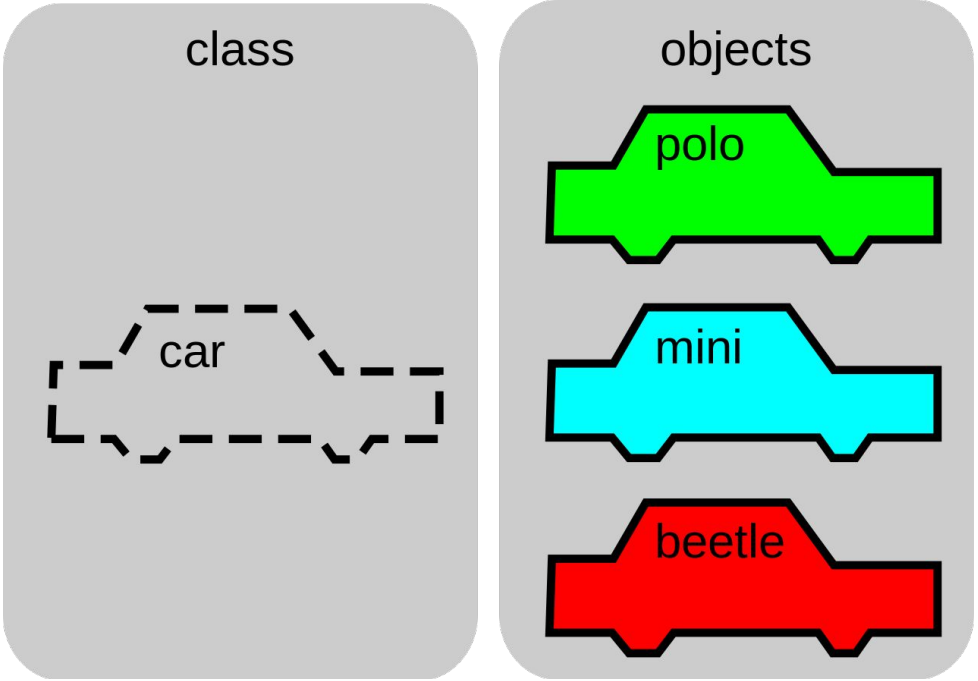
Classes

- A **class** or **type** is a blueprint for the structure of methods and attributes.

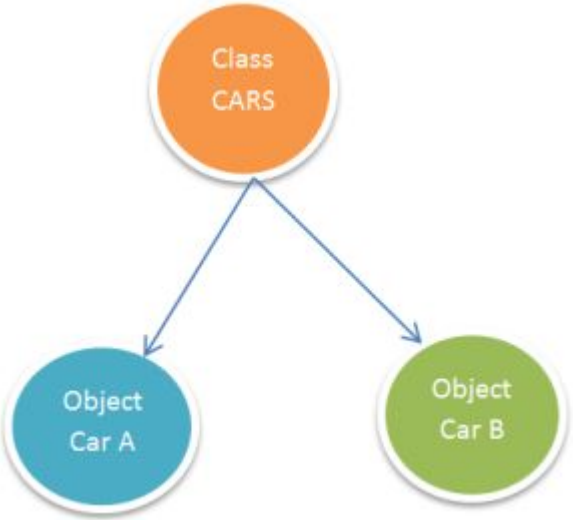


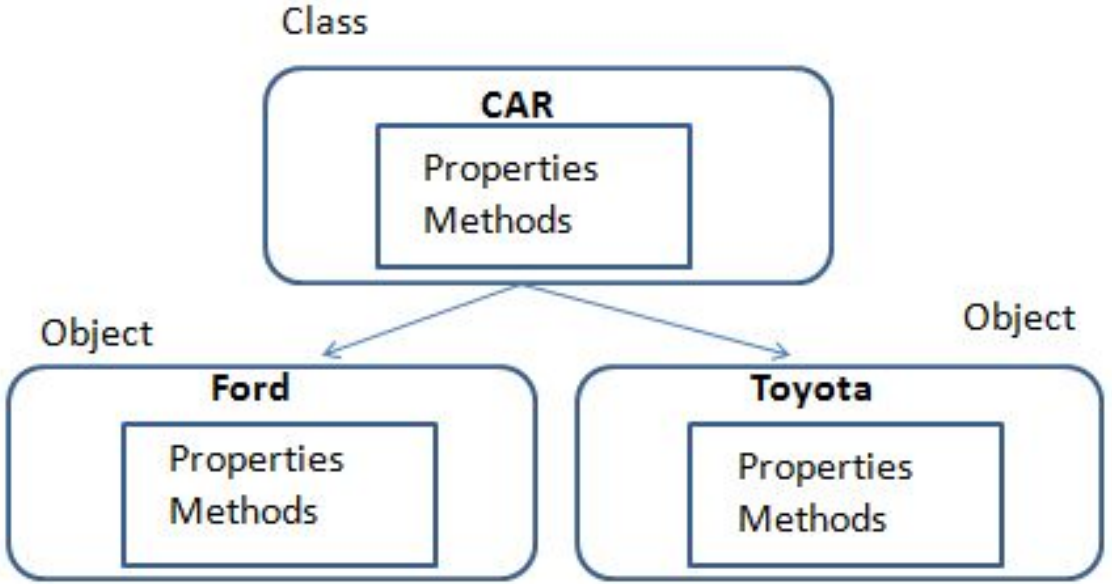
Instances

- An object is an individual **instance** of a class.



Instance Sample





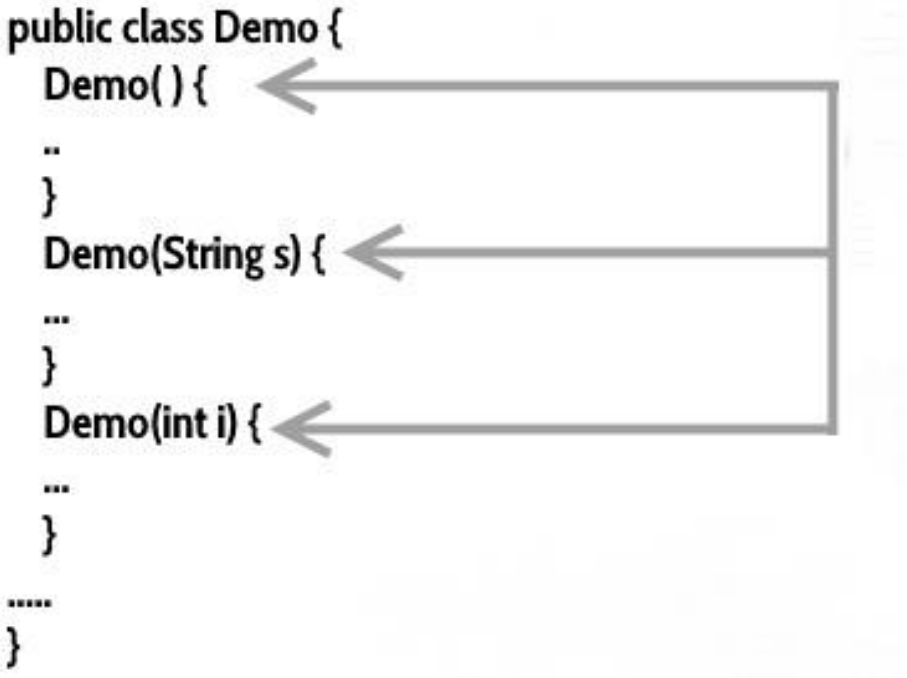
Car Ford = new Car(); Car Toyota = new Car();

Coding Sample

```
public class Car{
    private string _color;
    private string _model;
    private string _makeYear;
    private string _fuelType;

    public void Start(){
        ..
    }

    public void Stop(){
        ..
    }

    public void Accelerate(){
        ..
    }
}
```

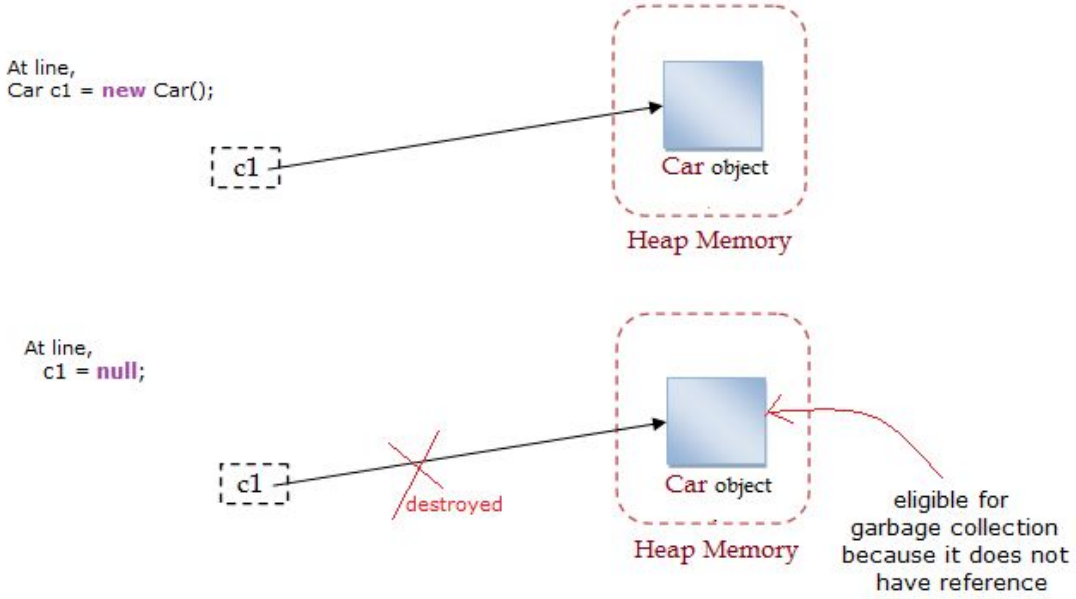
Constructor

- **Constructor** is a special type of method called to create an object.



Destructor

- **Destructor** is a member function which destructs or deletes an object.



4 Recap

Recap


Classes
•Blueprints
Objects
•Instances
Attributes
•Data
Methods
•Behaviors

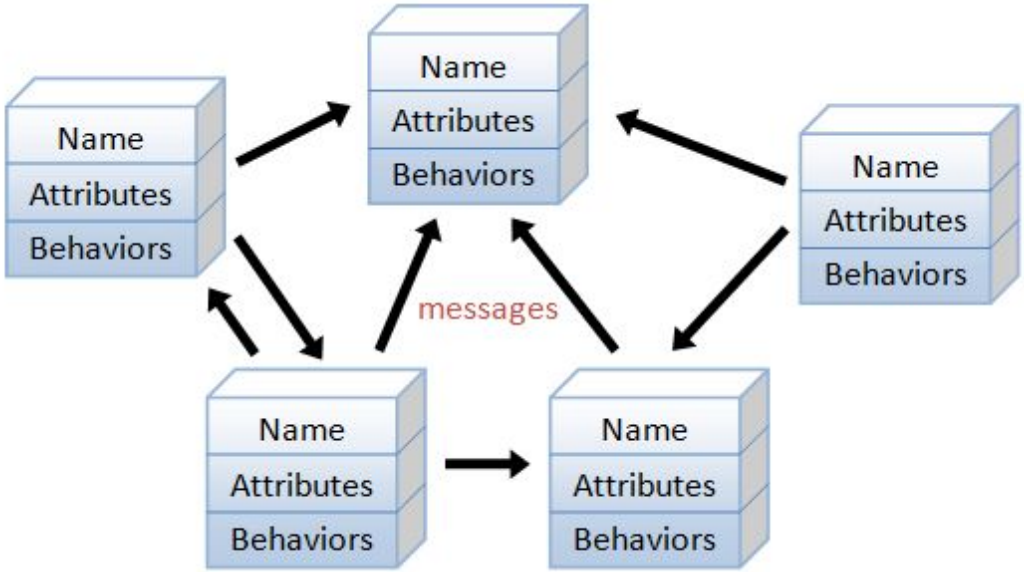
Basic Principles of OOP

Table of Contents

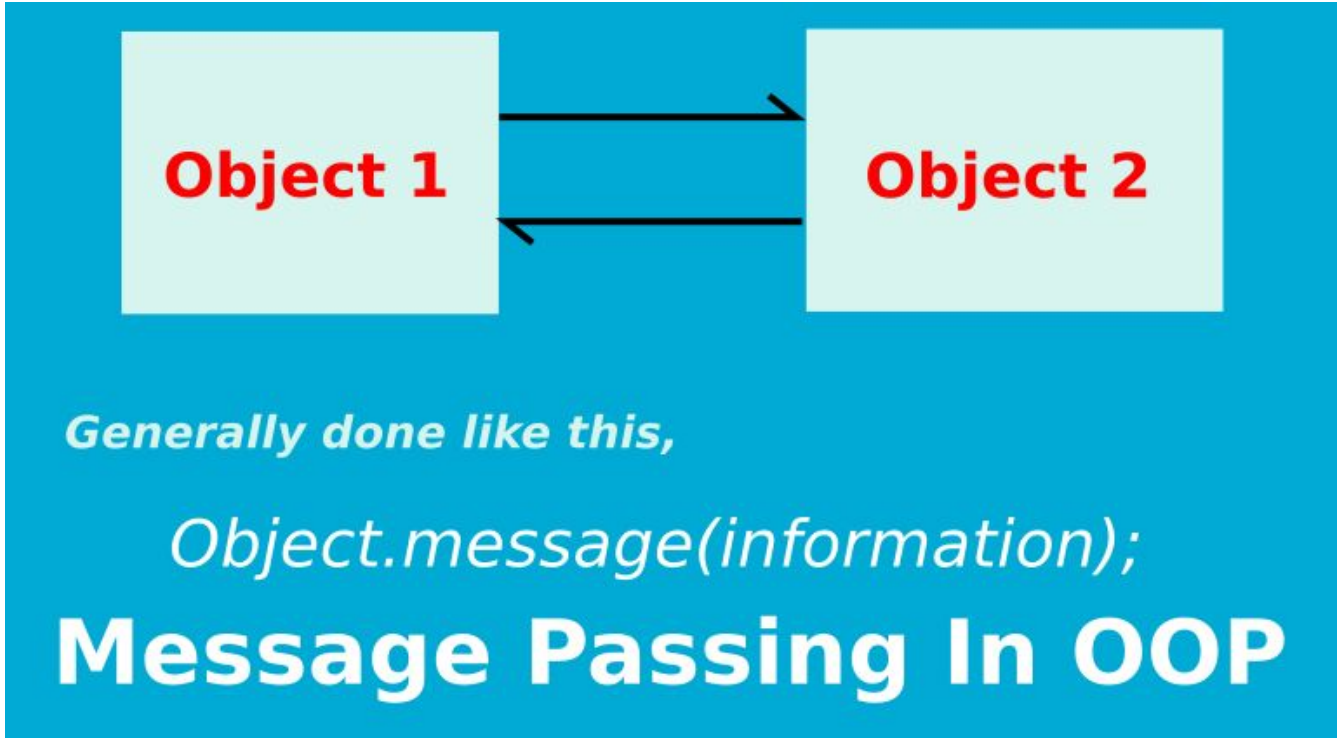
1 Encapsulation

Encapsulation

- An object-oriented program consists of many well-encapsulated objects and interacting with each other by sending **messages**.



Message Sending



Domain-Specific Methods

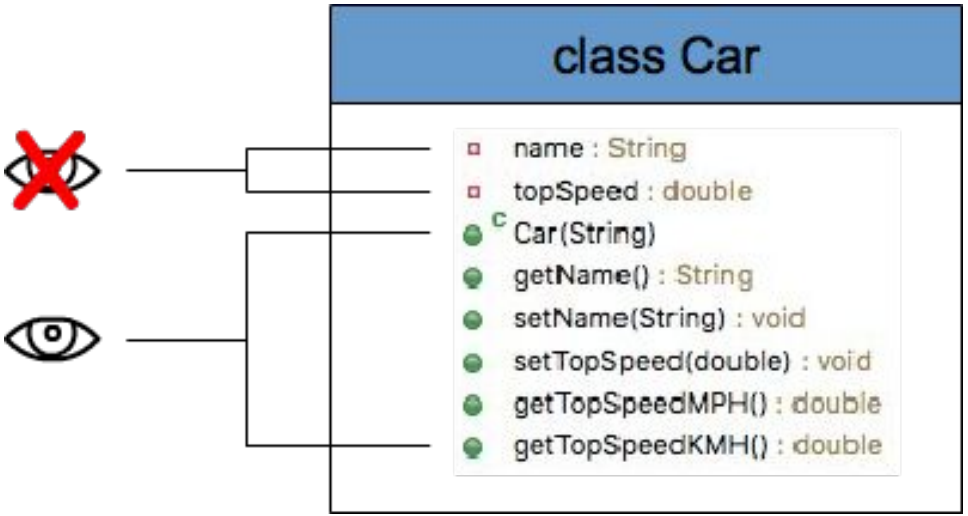
- Operations that how to control the **changes of the data** through methods.

Object 1	
Model	Volkswagen Polo
Fuel	Petrol
Make	2017
Start() Break() Accelerate()	

Object 2	
Model	Volkswagen Vento
Fuel	Diesel
Make	2017
Start() Break() Accelerate()	

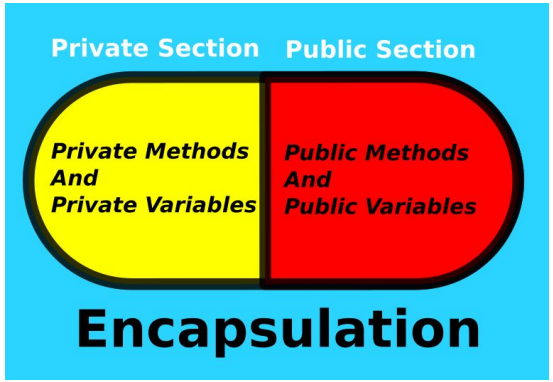
Getters and Setters

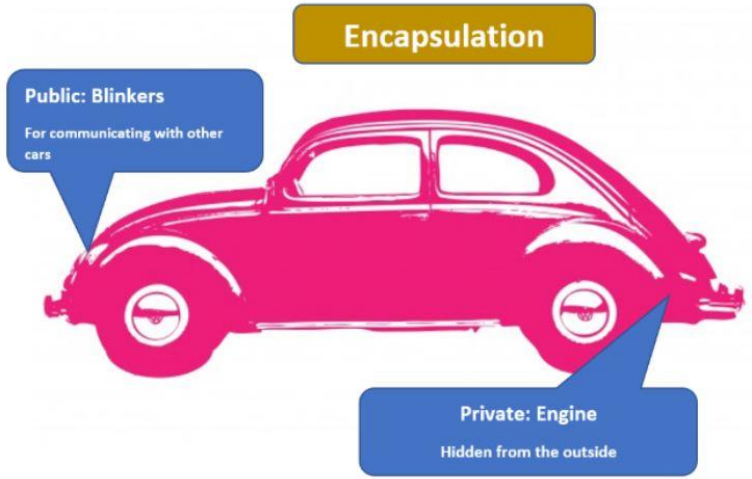
- **Setter** and **Getter** methods to modify and view the variables values.



Protecting Data

- **Encapsulation** adds **security** to code and makes it easier to **collaborate** with external developers. Within classes, most programming languages have **public**, **protected** and **private** sections.



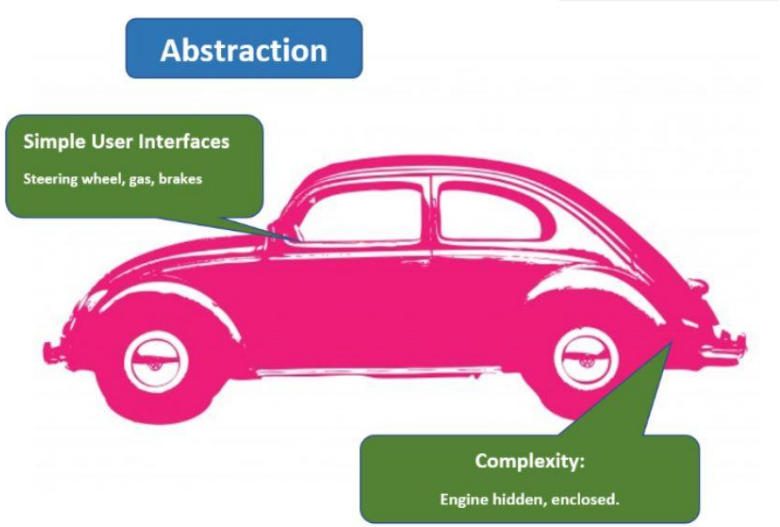


2

Abstraction

Abstraction

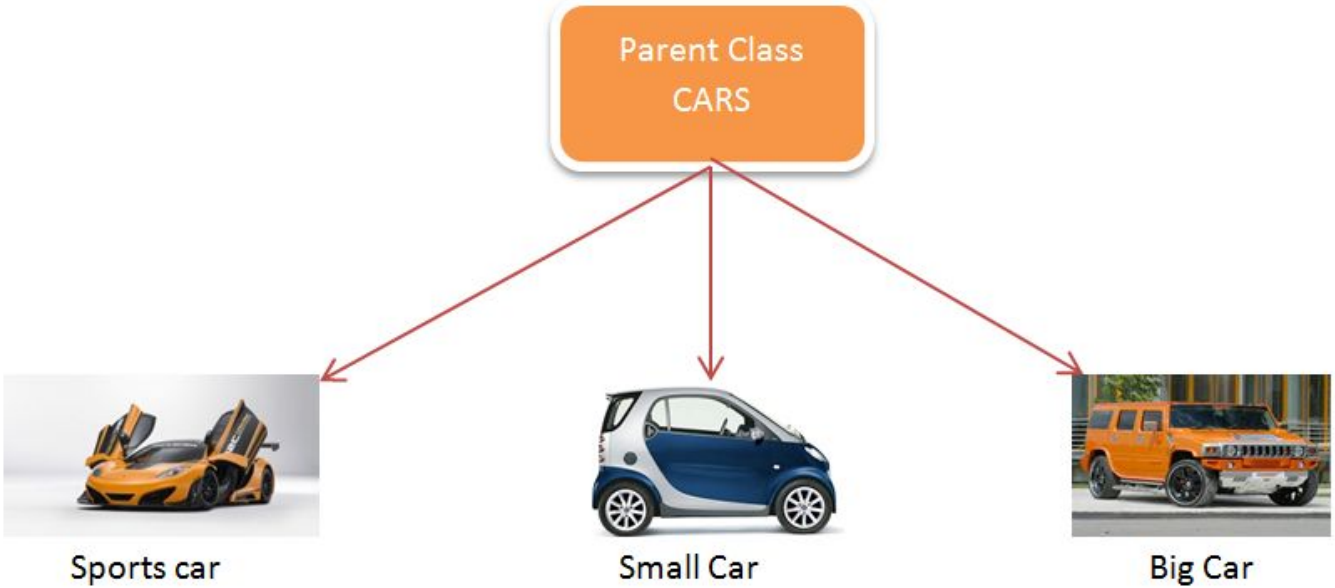
- A process of **hiding** the implementation details from the user.



3 Inheritance

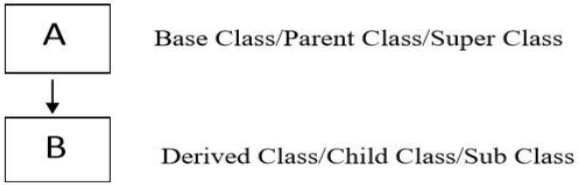
Inheritance

- Inheritance allows **child classes** to inherit features of **parent classes**.

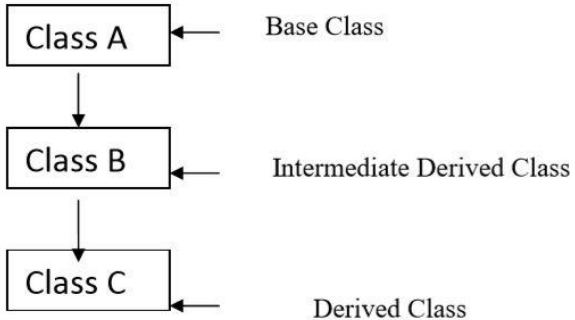


Types of Inheritance

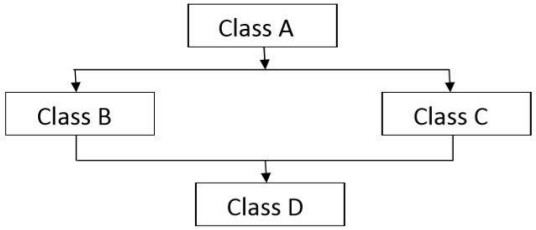
Single Inheritance



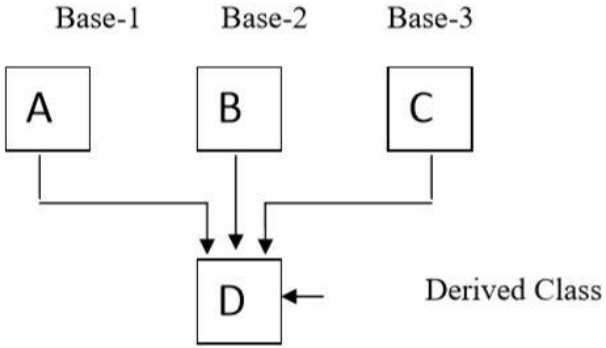
Multilevel Inheritance



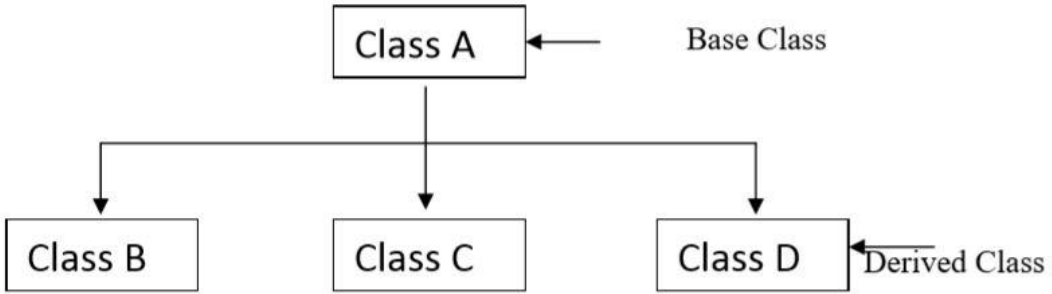
Hybrid Inheritance



Multiple Inheritances

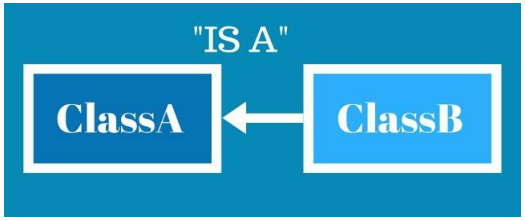


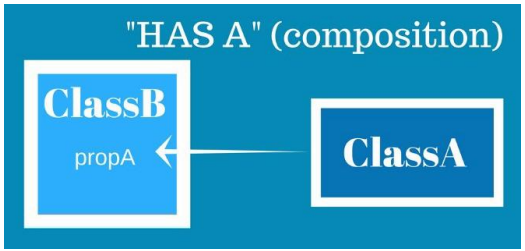
Hierarchical Inheritance

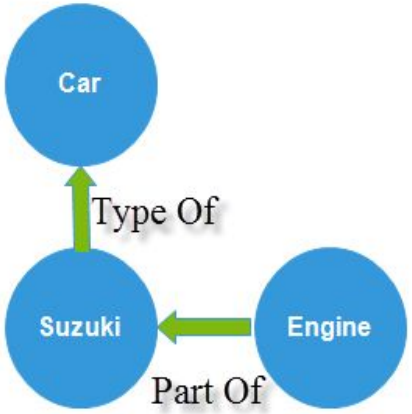


Types Of Relationship

- One of the advantages of Object-Oriented programming language is code **reuse**. This reusability is possible due to the **relationship** b/w the classes.

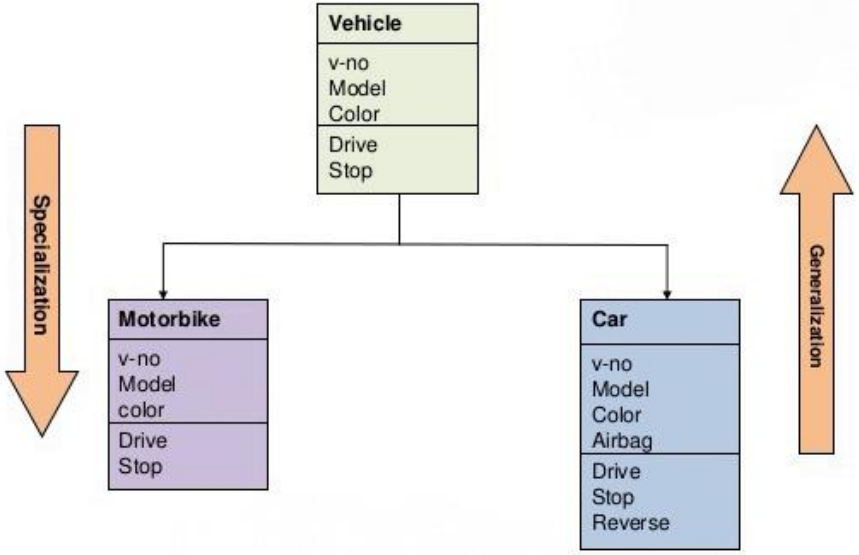






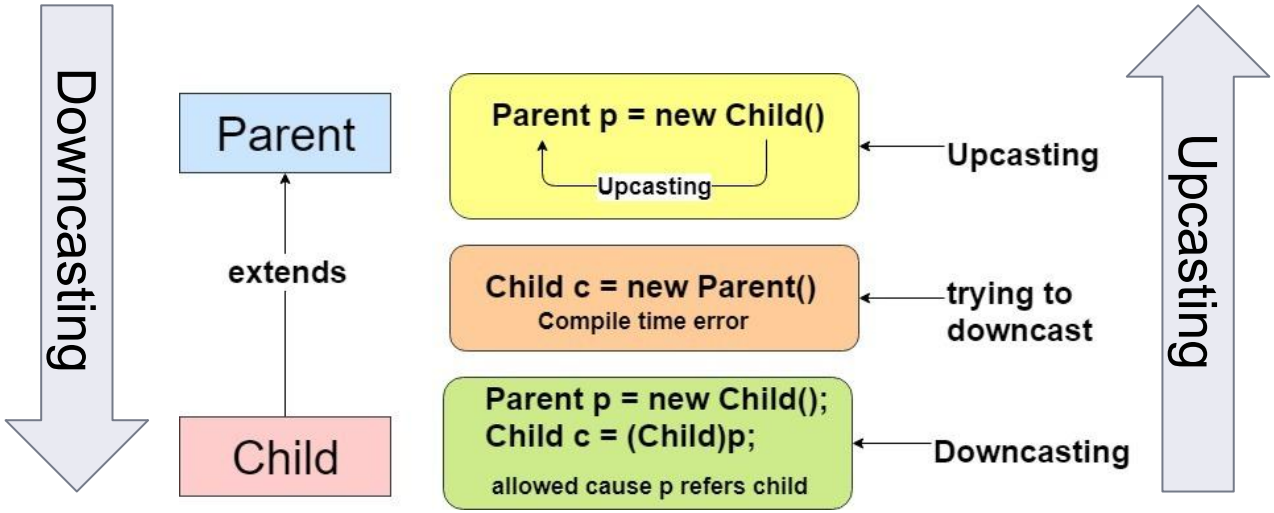
Generalization / Specialization

- Vehicle generalize what is common between Car and Motorbike.
- Car and Motorbike specialize Vehicle to their own sub-type.



Type Casting

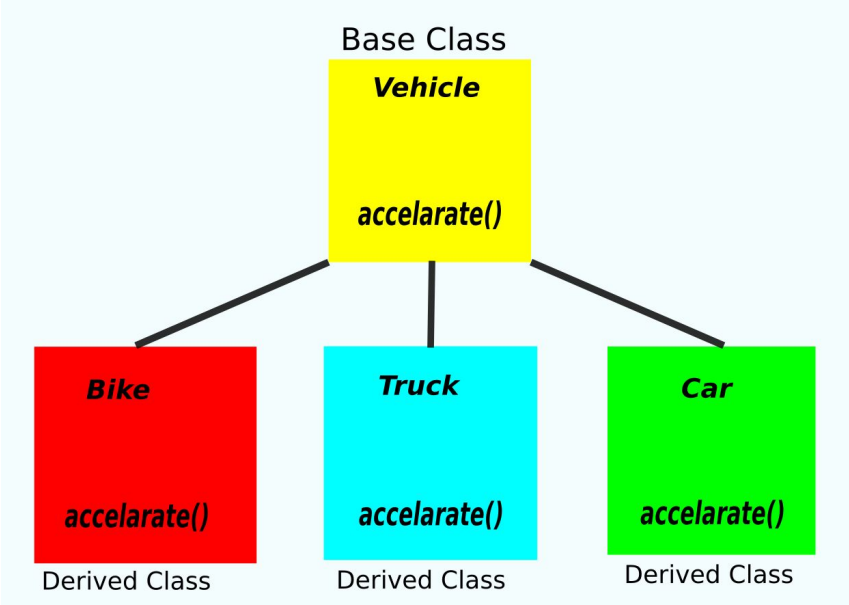
- Upcasting is casting a subtype to a supertype.
- When Subclass type refers to the object of Parent class, it is known as downcasting.



4 Polymorphism

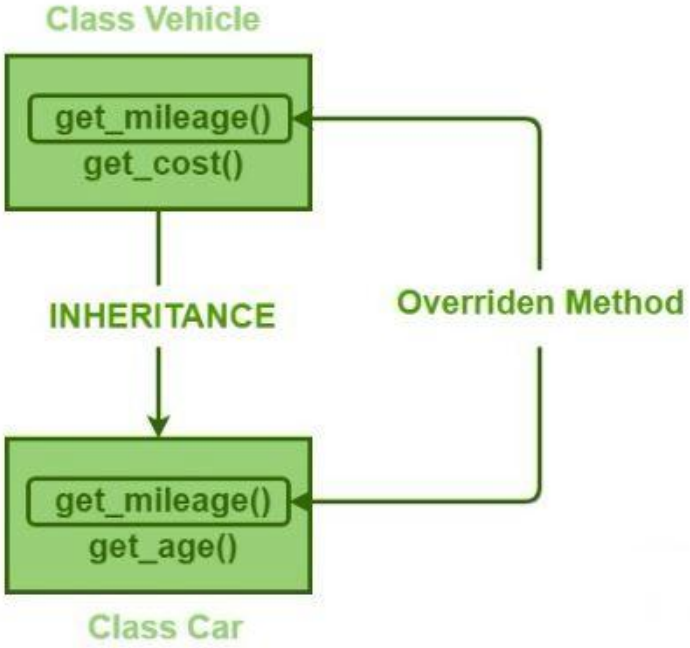
Polymorphism

- Poly means "many" and morphism means "form". Polymorphism means objects can take **different forms** under different conditions.



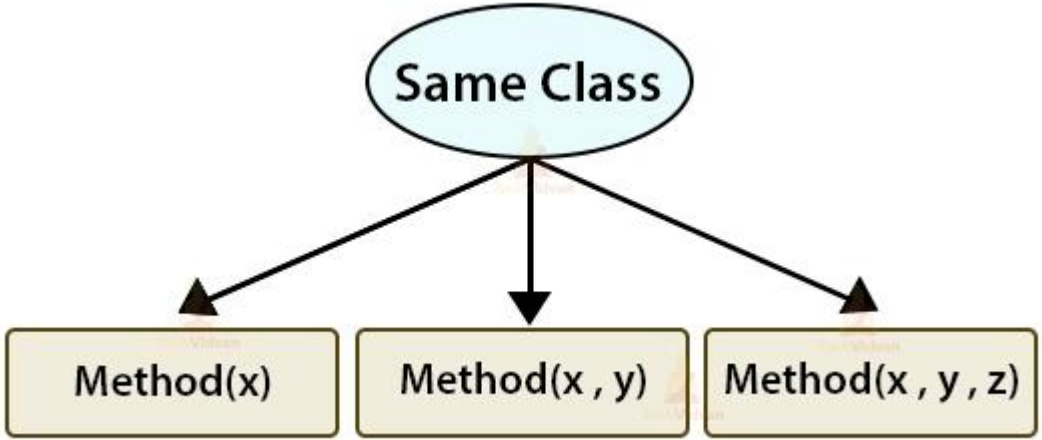
Overriding

- A child class can provide a **different implementation** than its parent class.



Overloading

- Methods or functions may have the same name, but a **different number of parameters** passed into the method call.



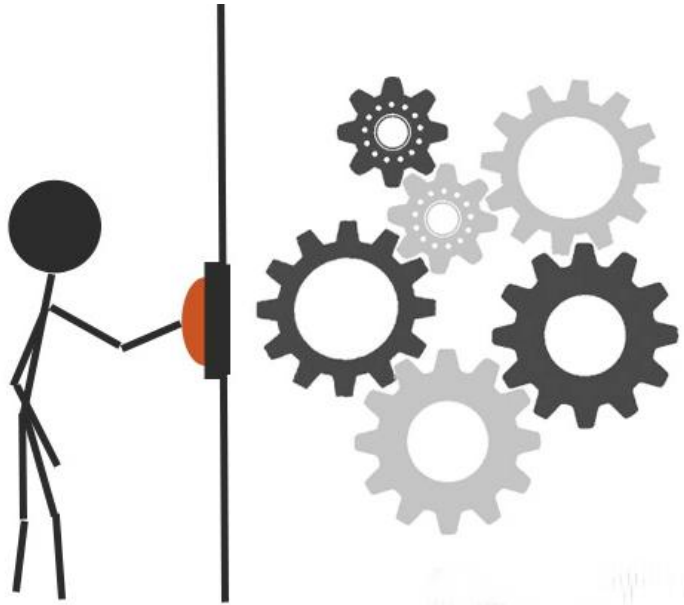
Introduction to Interfaces

Table of Contents

1 Basic Definitions

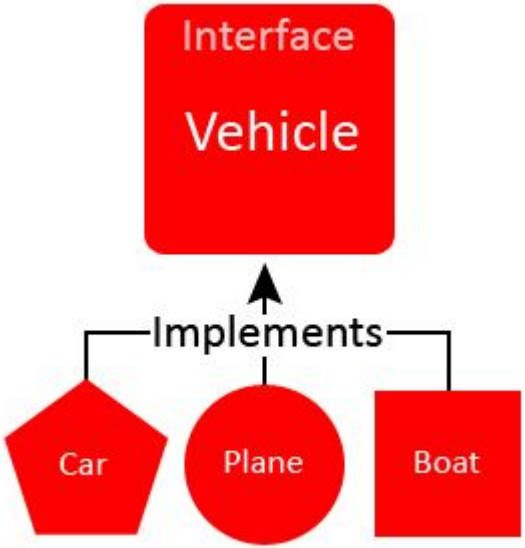
Basic Definitions

- An **interface** is a collection of methods that describes the behavior of an object.



Basic Definitions

- A class that **implements** an interface must implement all the methods declared in the interface. The methods must have the exact same **signature** (name + parameters) as declared in the interface.

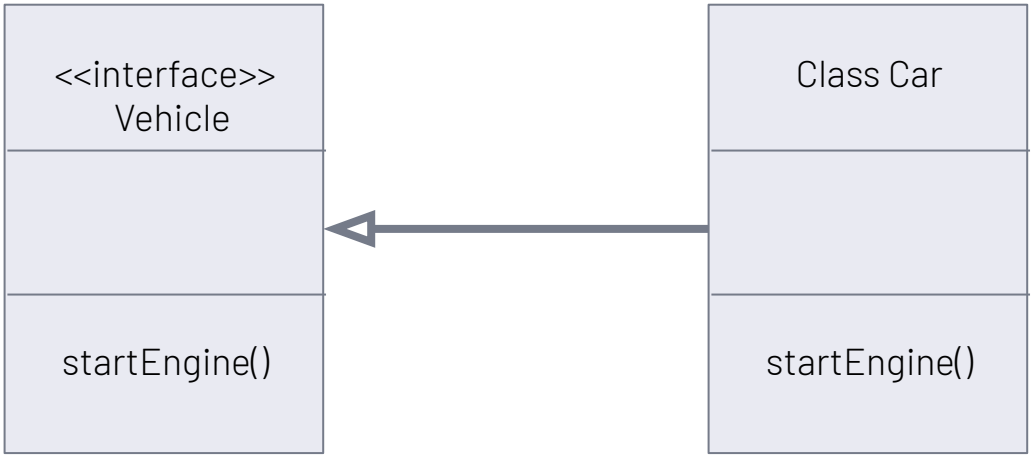


2

One-Method Interface

One-Method Interface

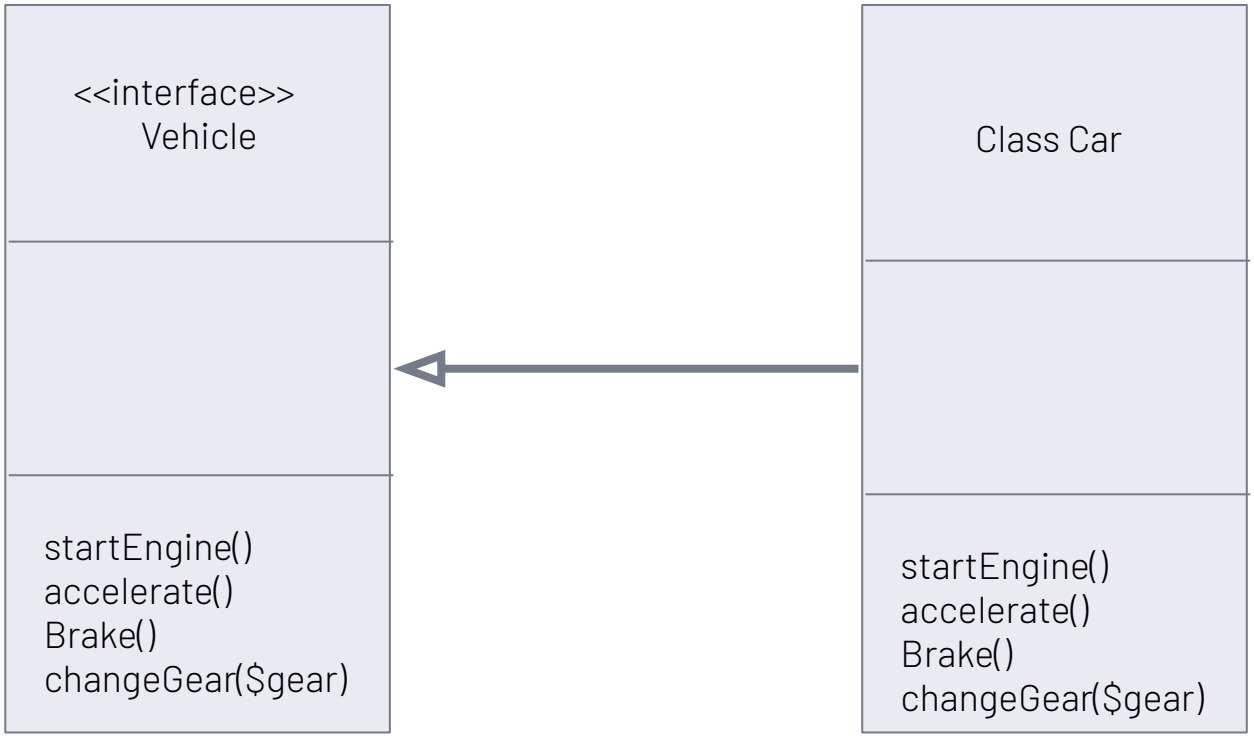
- One method stands for only **one skill** of an object.



3 Complex Interface

Complex Interface

- Many methods stand for **many skills** of an object.



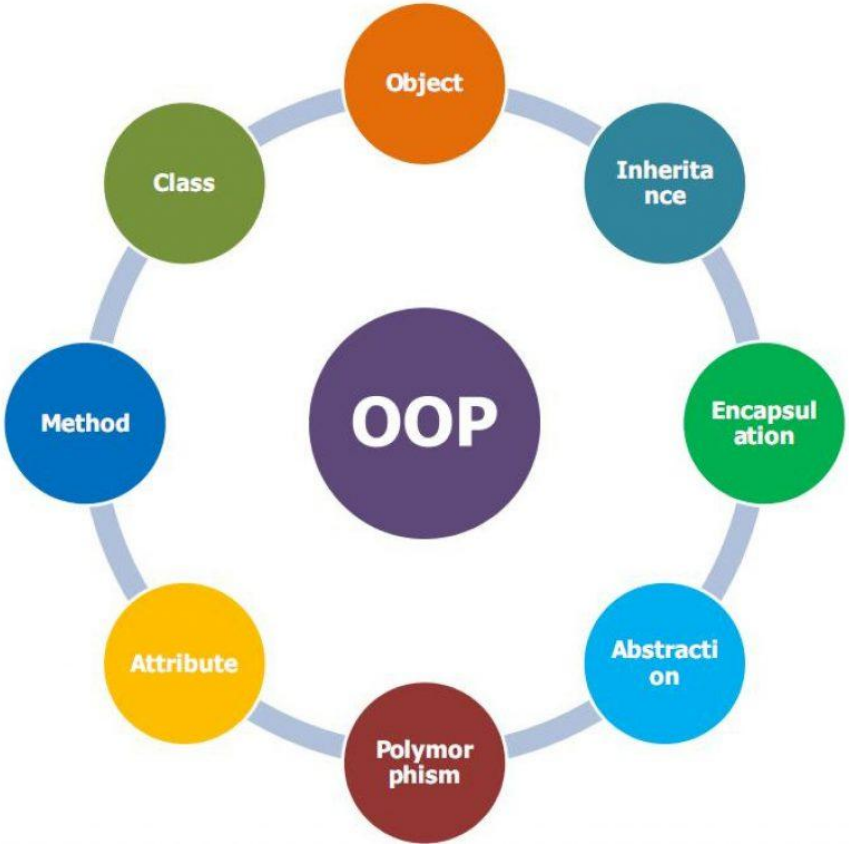
4 Responsibility of an Interface

Responsibility of an Interface

- Defining a system's **boundaries**.
- Depicting the **dependencies** of a system.
- **Coordinating** with various parties.
- Ensuring **compatibility** among systems.
- Exposing potential **problem areas** and **risks**.

Recap

Recap


Object
Inherita nce
Encapsul ation
Abstracti on
Polymor phism
Attribute
Method
Class
OOP

THANKS!

Any questions?