Application Programming

Week 2

Lecture 3

Deliverables

- Class Relationships Java
- Polymorphism
- Inheritance
- Encapsulation

Discussion

- Vehicle
- Has-a
- Is-a

Class Relationships in Java

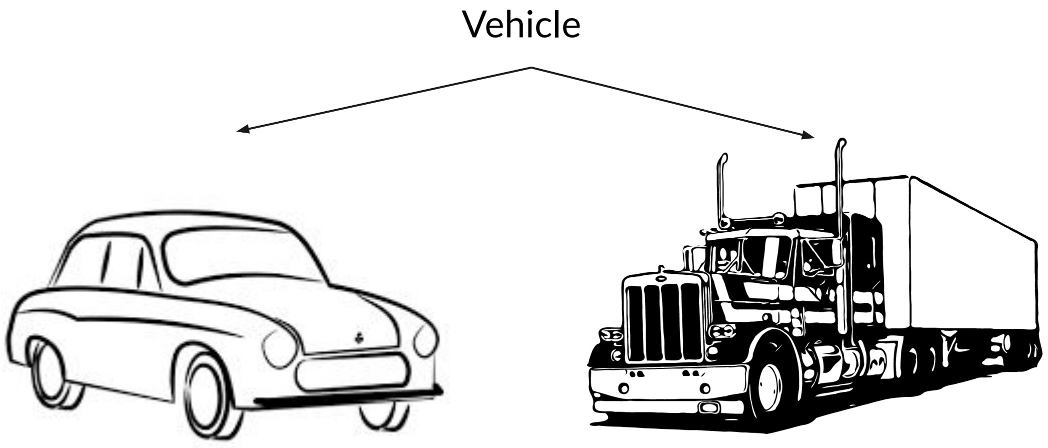
- "Has-a" relationship
    - Used to connect two classes, where one contains a reference/variable for another.
    - Example: Create a class for `Bank.java`, where all banks have one or more Account objects.
- "Is-a" relationship
    - Polymorphism - one class is related to the other.
    - Example: Create a class for `SavingsAccount.java`, where SavingsAccount is a type of Account.

Polymorphism

- Polymorphism
  - The provision of a single interface to entities of different types.
- Inheritance
  - Enables new objects to take on the properties of existing objects
  - *Superclass* → a class that is used as the basis for inheritance.
  - *Subclass* → a class which inherits from a superclass
- Encapsulation
  - Binds together the data (& functions that manipulate the data).
  - Keeps both safe from outside interference & misuse.
  - Lead to the important OOP concept of data hiding.

Example

- Cars are a type of vehicle.
- Cars have tires, a steering wheel, pedals.
- Cars can drive forward, turn, and drive backward.



Example

	Car.java
● Cars are a type of vehicle.	Subclass of Vehicle.java
● Cars have tires, a steering wheel, pedals.	Variables
● Cars can drive forward, turn, and drive backward.	Methods

Example

- *Car* inherits variables and methods from *Vehicle*

Vehicle.java

Variables:

- `steeringWheel`
- `gasPedal`
- `brake`
- `tires`

Methods:

- `drive(String dir)`
- `trun(String dir)`

Car.java

Variables:

- `heatedSeat`

Methods:

-