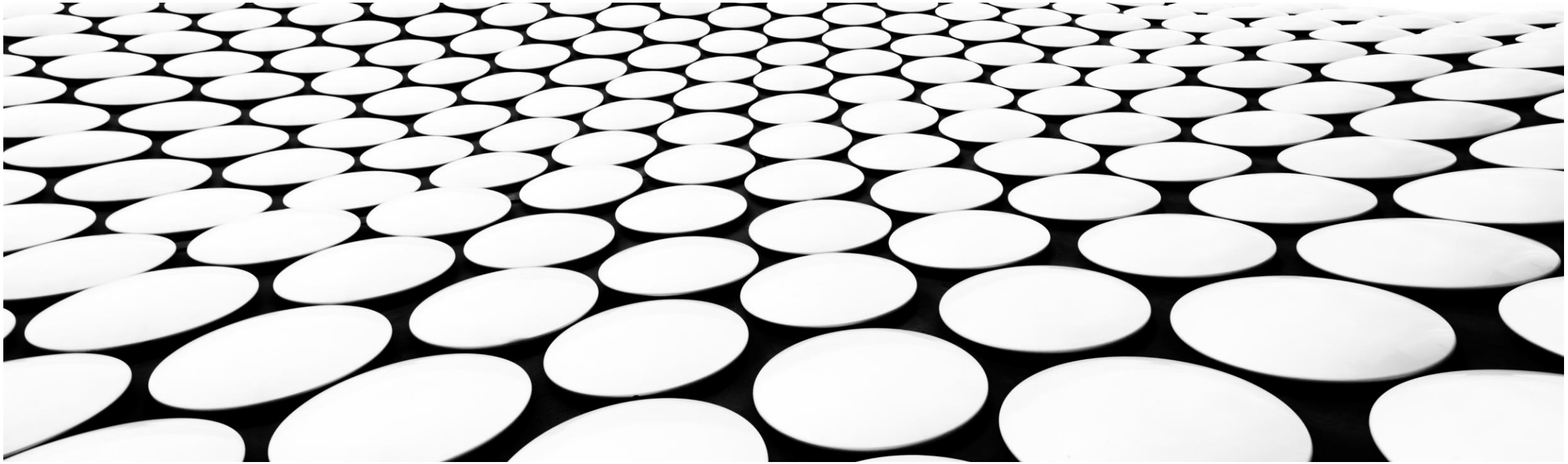
OBJECT ORIENTED PROGRAMMING (OOP)

MAHMOUD HUSSIEN MAHMOUD



DIFFERENCE BET. OOP AND PROCEDURAL PROGRAMMING?

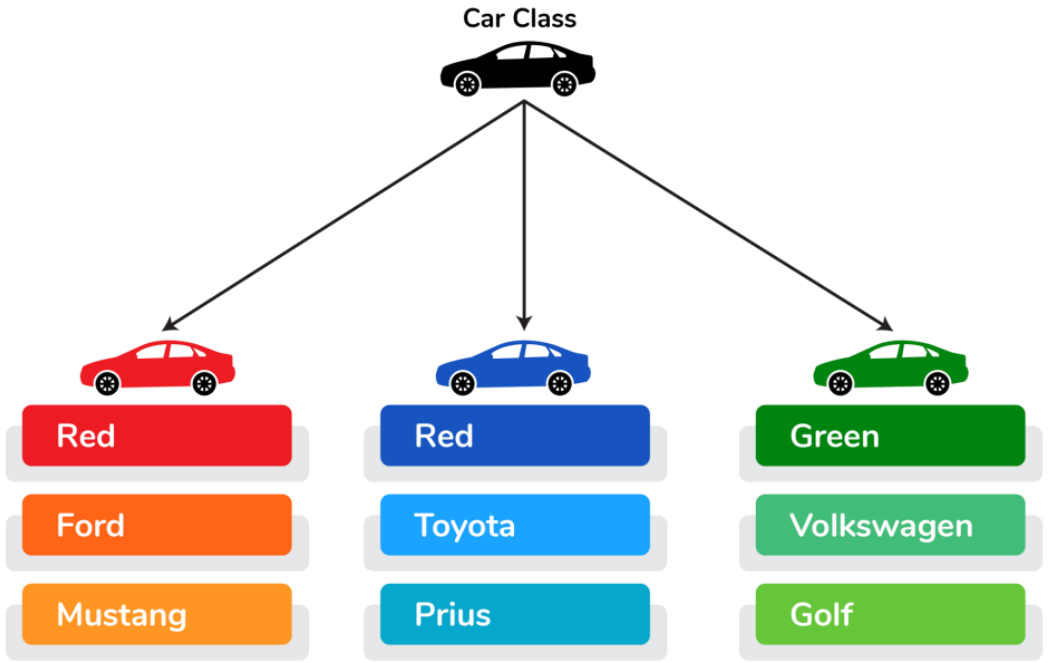
- OOP stands for Object-Oriented Programming.
- **Procedural programming** is about writing procedures or functions that perform operations on the data.
- **object-oriented programming** is about creating objects that contain both data and functions.

ADVANTAGE OF OOP OVER PROCEDURAL PROGRAMMING

- OOP is faster and easier to execute
- OOP provides a clear structure for the programs
- OOP helps to keep the C++ code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug
- OOP makes it possible to create full reusable applications with less code and shorter development time

WHAT ARE CLASSES AND OBJECT?

- Classes and objects are the two main aspects of object-oriented programming.



CLASSES AND OBJECTS “INSTANCE”

- Everything in Java is associated with classes and objects, along with its attributes and methods. For example: in real life, a car is an object. The car has **attributes**, such as weight and color, and **methods**, such as drive and brake.
- A Class is like an object constructor, or a "blueprint" for creating objects.
- Creating Object : ClassName ObjName = new ClassName();

CLASS ATTRIBUTES

- Declaring variables in class is actually the real definition of class attributes
- Accessing attributes by : ObjName.”Variable”
- Modifying attributes by : ObjName.”Variable” = newValue

WHAT IS CLASSES CONSIST OF????

- Variable to define data field
- Methods to define Actions
- Constructor which invoked to create a new object

WHAT IS THE DIFFERENCE BETWEEN MAIN CLASS AND ANY CLASS

- Main Class : Class which contain main method
- Any class : Class which not contain main method and it can’t be run so, we have to declare it in the main class
- #Any Program only contains One Main Class and any other classes which not a main one.

STATIC METHOD VS. PUBLIC METHOD

- Static methods which can be accessed without the need to create an object of it’s class
- Public methods needs to be accessed by creating an object of it’s class

CONSTRUCTORS

- constructor in Java is a **special method** that is used to initialize objects. The constructor is called when an object of a class is created. It can be used to set initial values for object attributes.
- There’s no need to implement class constructor as java itself create a default constructor
- Constructors can also take parameters,which is used to initialize attributes

ACCESS MODIFIERS

- Public : The code is accessible for all classes
- Private : The code is only accessible within the declared class
- Default : The code is only accessible in the same package. This is used when you don't specify a modifier.
- Protected : The code is accessible in the same package and **subclasses**.

NON-ACCESS MODIFIERS

- Final : The class cannot be inherited by other classes
- Abstract : the class cannot be used to create objects

ENCAPSULATION

- The meaning of **Encapsulation**, is to make sure that "sensitive" data is hidden from users. To achieve this, you must:
- - Declare Variable as private
- - provide Get and Set methods to access and update the values of the private variable.

GETTER AND SETTER??

- You learned from the previous chapter that private variables can only be accessed within the same class
- Get method : return variable value
- Set method : set the variable value

- Is there any need to create getters/setters? No, NetBeans has a built in getters/setters to implement

WHY ENCAPSULATION??

- Better control of class attributes and methods
- Class attributes can be made **read-only** (if you only use the get method), or **write-only** (if you only use the set method)
- Flexible: the programmer can change one part of the code without affecting other parts
- Increased security of data

INHERITANCE

- In Java, it is possible to inherit attributes and methods from one class to another. We group the "inheritance concept" into two categories:
- **subclass** (child) - the class that inherits from another class
- **superclass** (parent) - the class being inherited from

POLYMORPHISM

- Polymorphism means "many forms", and it occurs when we have many classes that are related to each other by inheritance.
- Like we specified in the previous chapter; **Inheritance** lets us inherit attributes and methods from another class. **Polymorphism** uses those methods to perform different tasks. This allows us to perform a single action in different ways.

SESSION CODE ON GITHUB

- Session Code