Object Oriented Programming (OOP) Concepts

Spyros Alertas

Created On: July 22, 2023
Last Edit: July 28, 2023

Contents

1 Introduction

This document is a brief introduction to the basic concepts/principles of Object Oriented Programming (OOP).

The most important OOP concepts are

- Class
- Object
- Inheritance
- Encapsulation
- Abstraction
- Polymorphism

2 Procedural vs Object Oriented Programming

In **Procedural Programming** the program is divided into procedures. Each procedure contains all the logical steps required to solve a specific problem/task.

In **Object Oriented Programming** the program is designed around Objects. Objects contain data and methods (procedures) that act on these data.

3 Why Object Oriented Programming?

Object Oriented Programming (OOP) is very useful to create large and complex applications.

Advantages of OOP:

- Modularity
- Re-usability
- Readability
- Security

Disadvantages of OOP:

- Performance Cost
- Memory Cost
- Steeper Learning Curve
- Design Complexity

4 Class & Object

A **Class** is a user defined data type that acts as blueprint and is used to create **Objects**. A class defines the data and the operations that are allowed on these data.

Objects are instances of classes. Each time an instance of a class is created, all class fields are initialized and separate memory space is allocated for each instance.

5 Inheritance

Inheritance is a mechanism in which one class (called **Child Class** or **Subclass**) acquires all the properties and behaviors of another class (called **Parent**, **Super** or **Base Class**).

6 Encapsulation

Encapsulation is the process of integrating the data and the code (methods) into a single unit (class). The data are hidden from other classes and can be directly accessed only by methods from the same class they are found.

Encapsulation can be achieved by using access modifiers (e.g.: private, public, etc).

7 Abstraction

Abstraction is the process of hiding implementation details in order to reduce complexity and improve efficiency.

Abstraction can be achieved through interfaces and abstract classes.

8 Polymorphism (Compile-Time & Runtime)

Polymorphism is the provision of a single interface to entities of different types. The program has the ability to identify the correct type at each moment.

Compile-Time or **Static** or **Early Binding Polymorphism** can be achieved through method overloading.

Runtime or **Dynamic** or **Late Binding Polymorphism** can be achieved through method overriding.

9 Abstract Class & Abstract Method

An **Abstract Class** is a type of class that has at least one abstract method.

An **Abstract Class**:

- cannot be instantiated (cannot be used to create objects)
- can have variables
- can have non-abstract (implemented) methods
- must have at least one abstract method

An **Abstract Method** can exist only in an abstract class and contains only the definition/signature (return type, parameters and exception) of the method without the body/implementation.

10 Interface

An **Interface** is used as a common protocol for all the classes that implement it, as these classes will have to implement all the methods that are defined in the interface.

An **Interface**:

- cannot be instantiated (cannot be used to create objects)
- can have only static final variables
- cannot have implemented methods

11 Method Overloading & Method Overriding

Method Overloading is when two or more methods share the same name but have different signature (number and/or type of parameters).

Method Overriding happens between child and parent class or classes that child class implicitly inherits from when methods have the same signature (name and parameters)

12 Instance & Class Variables

Instance Variables are unique for each instance of the class. Instance variables can be accessed using the object reference, not the class name.

Class Variables are common to all instances of the class. Class variables have the keyword static and can be accessed using the class name.

13 External Resources

- Procedural Programming - Wikipedia
- Object Oriented Programming - Wikipedia
- Programming Paradigms - Wikipedia

14 Acronyms

OOP Object Oriented Programming

e.g. latin: exempli gratia - english: for example