

Object-Oriented Programming

...

Objects, Classes & Messages

Introduction

- **What makes a program useful ?** Abstraction + Computation + Complexity
- **Abstract Data Types (ADTs):** class of objects whose logical behavior is defined by a set of values and a set of operations
- **How is this related to OOP:** OOP provides us with rules and techniques to extend our use of ADTs.

What is OOP ?

- **Mainstream definition (Working OOP):** is a computer programming model (paradigm) that organizes software design around data, or objects, rather than functions and logic.
- **Historically Intended definition (Philosophical OOP):** it is all about **things** sending **messages** back and forth between **black boxes**, treating objects as collaborating, independent entities. => Message driven development.

Comparison Between Procedural vs OOP code

- Car example
- Exercise:
 - Living Being
 - Human
 - Profession: Engineer, Doctor, Accountant, ...
 - Gender: Man, Woman
 - Animal
 - Kind: Carnivore, Vegetarian, Omnivore

A word cloud of Object-Oriented Programming (OOP) concepts. The words are arranged in a circular pattern on a dark blue background. The colors of the words include pink, yellow, and white. The sizes of the words vary, with the largest words being 'inheritance', 'oop', 'encapsulation', 'polymorphism', 'object', 'abstraction', 'methods', 'class', 'constructor', 'interface', 'messaging', 'property', 'parents', 'reuse', 'attribute', 'module', 'delegation', and 'methods'. Other words include 'composition', 'relationship', 'subclass', 'smalltalk', 'instance', 'type', 'prototype', 'mixins', 'dependency (1)', 'receiver', 'dependency', 'nested class', 'field', 'dispatch', 'behavioral delegation', 'overloading', 'pattern', 'accessor methods', 'java', 'object serialization', 'multiple inheritance', 'destructor', 'javascript', 'factory', 'member', 'collaboration', 'simula67', 'open recursion', 'state', 'control', and 'aggregation'.

messaging

composition

relationship

subclass

smalltalk

instance

type

prototype

property

parents

mixins

dependency (1)

class

constructor

interface

receiver

dependency

nested class

reuse

inheritance

field

dispatch

behavioral delegation

attribute

oop

encapsulation

overloading

module

pattern

accessor methods

java

object serialization

multiple inheritance

destructor

javascript

factory

member

collaboration

simula67

open recursion

state

control

aggregation

polymorphism

abstraction

methods

delegation

Key Concepts of OOP ?

- **Class:** Pattern or blueprint for creating an object. A class contains all attributes and behaviors that describe or make up the object.
 - Attributes
 - Methods
 - Instantiation
- **Object:** A thing that can be seen and used, It is, in most cases, an instance of a class. It is a construct that combines a state (data) and behavior (operations). When combined, the state and the behavior represent an abstraction of a "real-world" object.
 - Has state
 - Has behavior
 - Can communicate through messages
- **Instance:** An object created from a class (is an instance of that class)

Key Concepts of OOP

- **Attribute:** Characteristics that describe the object (sometimes referred to as properties).
=> Simply, variables.
- **Methods:** Operations (or actions) that objects perform or operations which are performed to an object. Sometimes referred to as behaviors.
 - It is triggered(called/invoked) when a **message** is received.
 - It is a function or a procedure
- **Signature (Annotation):** defines the inputs and outputs for a function (method in our case) and their types.

Key Concepts of OOP

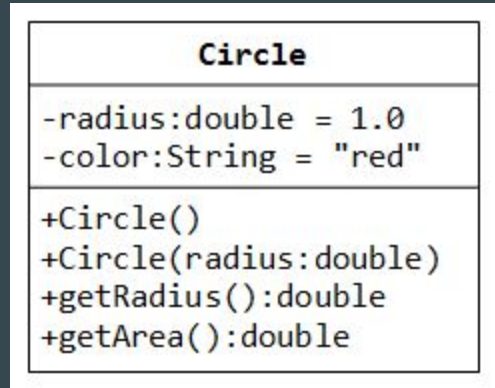
- **Message:** Way to communicate between objects, it has 4 parts:
 - identity of the recipient object
 - code to be executed by the recipient
 - arguments for the code
 - return value

=> It is simple function (method) call

Key Concepts of OOP

- **Abstraction:** Refers to hiding the internal details of an object from the user.
Or simply, eliminate irrelevant details.
Example: Car => (make, year, maxSpeed ...)
- **Encapsulation:** Refers to the combining of an object's attributes and behaviors into a single package, hiding it from external users.
 - Access modifiers/specifiers (public, private, protected, package)
- **Inheritance:** Refers to the capability of creating a new class from an existing class.
 - Base Class (parent)
 - Derived Class (child)
- **Polymorphism:** refers to the ability of a variable, function or object to take on multiple forms.

UML building blocks



Key Concepts of OOP

Primitive data types are passed by value in JS/TS, while complex data types (objects and arrays) are passed by reference.

A static class is a class that cannot be instantiated or subclassed.

A abstract class is a class that cannot be instantiated, but can be subclassed.

Static properties: they are bound to the Class itself, and not to an instance

- Static methods are often utility functions, such as functions to create or clone objects.
- Static properties are useful for caches, fixed-configuration, or any other data you don't need to be replicated across instances.

Instance properties: they are bound to a particular instance of a class (actual objects)

Key Concepts of OOP

Access modifiers (or access specifiers) are keywords in object-oriented languages that set the accessibility of classes, methods, and other members. Access modifiers are a specific part of programming language syntax used to facilitate the encapsulation of components.

- **Public:** it can be accessed everywhere.
- **Protected:** it can be accessed only within the class itself and by inheriting child classes.
- **Private:** it may only be accessed by the class that defines the member.

Key Concepts of OOP

Constructor is a class that cannot be instantiated or subclassed.

Destructor is a class that cannot be instantiated, but can be subclassed.

Property accessors (getters and setters): provide access to an object's properties by using the dot notation or the bracket notation.

Polymorphism mechanisms:

- **Overloading:** is the action of defining multiple methods with the same name, but with different parameters.
- **Overriding:** An override is a type of function which occurs in a class which inherits from another class. An override function "replaces" a function inherited from the base class.