

OOP (Object-Oriented Programming) is a programming approach that revolves around objects. Imagine objects as blueprints for real-world things, like a Dog or a Car.

Benefits of OOP:

- **Code Reusability:** Inheritance allows you to reuse code for similar objects.
- **Modularity:** Classes break down complex programs into manageable units.
- **Data Protection:** Encapsulation safeguards data integrity.

OOP is like building with Legos! You create reusable pieces (classes) and combine them to form more complex objects. This makes coding more efficient and organized.

- ❖ **Objects:** These combine data (attributes, like breed for a Dog) and functionalities (methods, like bark() for a Dog) to represent real-world entities.
- ❖ **Classes:** These are templates that define the structure of objects. They specify what attributes and methods an object of that class will have.
- ❖ **Encapsulation:** This bundles data and methods together within a class, protecting the data from outside access.
- ❖ **Inheritance:** This lets you create new classes (subclasses) based on existing ones (super-classes). Subclasses inherit the attributes and methods of the superclass, promoting code reuse.

Inheritance in Python

Inheritance is a fundamental concept in Object-Oriented Programming (OOP) that allows you to create new classes (subclasses) based on existing classes (super-classes). Subclasses inherit attributes and methods from the superclass, promoting code reuse and creating a hierarchy of related objects.

Types of Inheritance

➤ **Single Inheritance**

A subclass inherits from one parent class. This is the most common and straightforward type of inheritance.

➤ **Multilevel Inheritance**

A subclass inherits from another subclass, which in turn inherits from a parent class. This creates a chain of inheritance.

➤ **Multiple Inheritance**

A subclass inherits from multiple parent classes. This can be trickier to manage as it can lead to diamond problems (where two parent classes have the same method name).

➤ **Hierarchical Inheritance**

Multiple subclasses inherit from a single parent class, creating a hierarchy of specialized classes.

➤ **Hybrid Inheritance**

A combination of multiple inheritance techniques.