Python Object Oriented Programming

In this presentation, we will explore the principles of Python Object Oriented Programming and its best practices for effective software development.

by Harish Kumar Sharma

Introduction to Python OOP

Understanding the fundamental concepts of OOP and how they are implemented in Python, including encapsulation, abstraction, and inheritance.

Classes and Objects

1

What are Classes?

Learn how to define classes and create instances of objects with attributes and methods.

2

Working with Objects

Explore how to access object attributes and call object methods to perform tasks and manipulate data.

3

Class Relationships

Discover how to establish relationships between classes including composition and aggregation.

Inheritance

Single Inheritance

Learn about single inheritance and how it allows a class to inherit properties and methods from a single parent class.

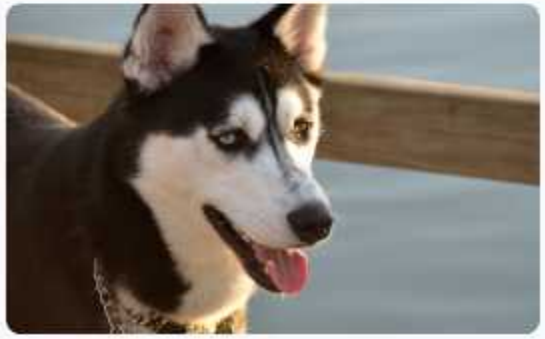
Multiple Inheritance

Explore the concept of multiple inheritance and how it enables a class to inherit from multiple parent classes.

Method Resolution Order

Understand how Python determines the order in which it searches for inherited methods.

Polymorphism



What is Polymorphism?

Discover how polymorphism allows objects of different classes to be treated as objects of a common class.

Method Overriding

Learn how to override methods in child classes to provide specific implementations.

Operator Overloading

Explore how operator overloading enables operators to behave differently based on the types of operands.

Encapsulation

1

What is Encapsulation?

Understand the concept of encapsulation and how it helps achieve data and information hiding.

2

Access Modifiers

Discover different access modifiers in Python and their role in controlling access to class members.

3

Properties and Decorators

Learn about properties and decorators, which allow controlled access to class attributes.

Abstraction

Understanding abstraction and its role in simplifying complex systems by providing a high-level view of functionality.

Best Practices for Python OOP

1 Code Reusability

Explore techniques for writing reusable code using classes and objects.

2 Code Readability

Learn how to write clean and easily understandable code by following coding conventions and naming conventions.

3 Testing and Debugging

Discover strategies for testing and debugging object-oriented Python code for robustness and reliability.