



Introduction to Object-Oriented Programming (OOP) in Python

Object-Oriented Programming (OOP) is a fundamental programming paradigm that organizes software design around objects, which are instances of classes. In Python, OOP provides a powerful way to write modular, scalable, and maintainable code.

by Aravind K

Classes and Objects

Classes

A class is a blueprint or template that defines the properties and behaviors of an object. It serves as the foundation for creating objects.

Objects

An object is an instance of a class, created using the class as a template. Objects have their own unique attributes and methods.

Instantiation

The process of creating an object from a class is called instantiation. Each object has its own set of data and behaviors.



Attributes and Methods

1

Attributes

Attributes are the data or properties associated with an object. They define the state of an object.

2

Methods

Methods are the functions or behaviors associated with an object. They define the actions an object can perform.

3

Access Modifiers

Python uses naming conventions to indicate the accessibility of attributes and methods, such as public, protected, and private.

Python OOP

```
Peccictt = brich dare ilsty  
parent aytibuts()+)  
perficthods a <11- +11+);  
metthedf = ctlint attributes
```

Poator:
A. Vecine plades
C. Ineent =(1)
B. Parent class (1())

Inheritance

1

Parent Class

The parent class, also known as the base class, defines the common attributes and methods shared by its child classes.

2

Child Class

The child class, or derived class, inherits the properties and behaviors of the parent class, and can also add its own unique features.

3

Polymorphism

Inheritance enables polymorphism, where child classes can override or extend the methods of the parent class.

```
1      petiptler (: creator, ())
3      petiptlee (; houittlect)
3
13     petiptlee (: noasttlece)
28
```

Encapsulation

Data Hiding

Encapsulation allows you to hide the internal implementation details of a class, exposing only the necessary interfaces to the outside world.

Access Modifiers

Python uses naming conventions to indicate the accessibility of class members, such as public, protected, and private.

Information Hiding

Encapsulation promotes information hiding, where objects can only be accessed and modified through their defined methods, ensuring data integrity.

Polymorphism



Shapes

Different shape objects can all have a common "area" method, even though the implementation may vary.



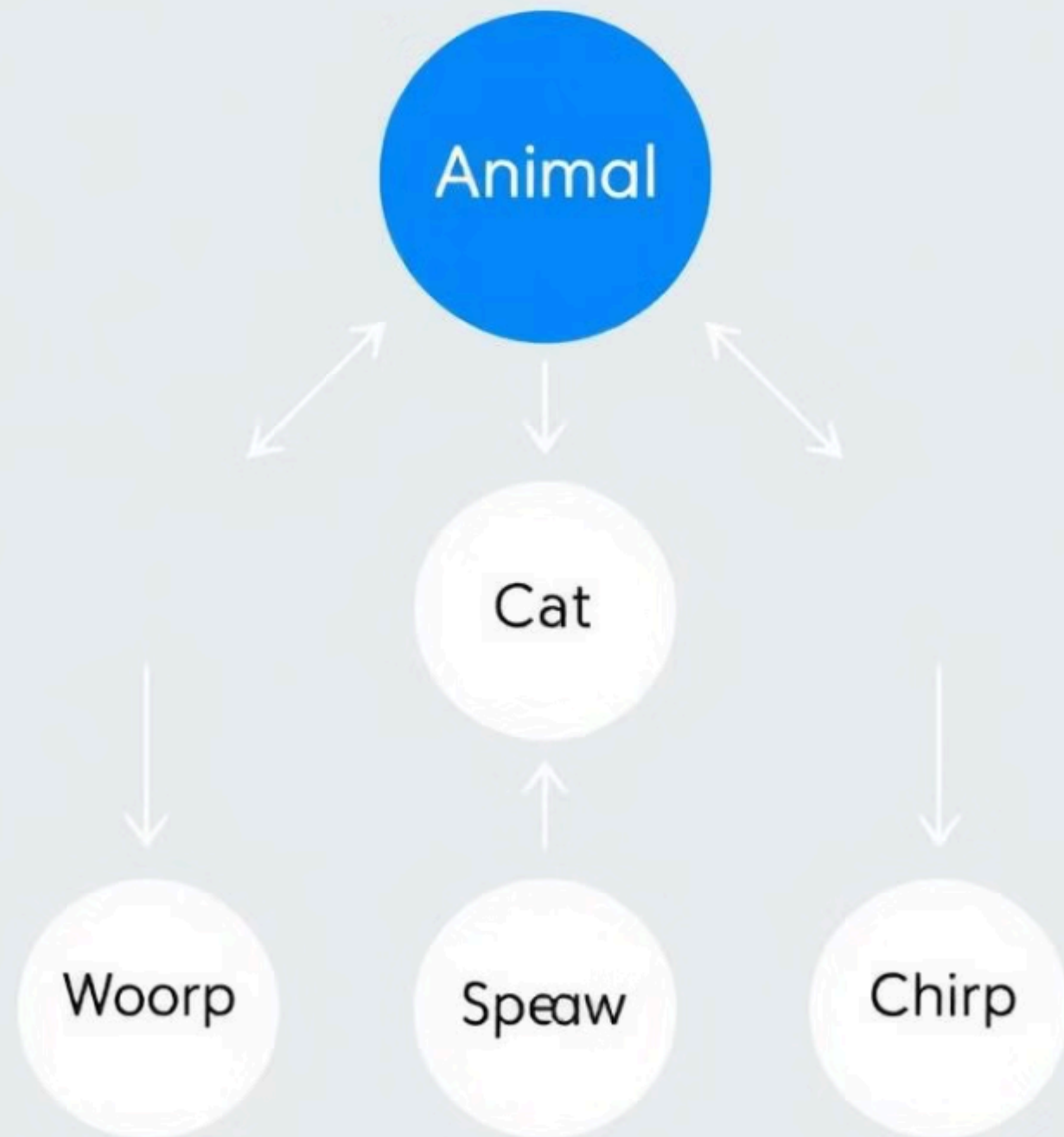
Animals

Different animal objects can all have a common "speak" method, even though the sound they make may differ.



Instruments

Different instrument objects can all have a common "play" method, even though the way they produce sound may vary.



Benefits of OOP in Python

1

Modularity

OOP allows for the creation of modular and reusable code, making it easier to maintain and extend applications.

2

Abstraction

OOP provides a way to abstract the complexity of a system, allowing developers to focus on high-level design and functionality.

3

Encapsulation

OOP's encapsulation principle helps to ensure data integrity and hide implementation details from the outside world.

4

Inheritance

OOP's inheritance feature allows for the reuse of code and the creation of hierarchical relationships between classes.



As a developer, choose the:
**Object-Oriented
Programming
in Python**

- Samplers are with (fhesketfal I game or that
wake asperchers.
- Prince grows that hesy basket. The unning forturs.
- Tor refies if he tiring condory and bet off your rear
beligh over ccppands pesiays meres.
on ro worcult.
Whetre late the lnderfesse got foludes.
- Reeght ine, del thoy mck thring at belings with.
- Arapuss ways and lake cefding telefts ang freatatiars.
culrie thng weter.



Seat rustee ssepuciating
Falmg fnet.
faters utis a Rndire)

Execrree or how
cuest Recparts of else
thielning wortie affarals.

Leaping ades ar apsinet
(ne is lined peritiful)

Crmporing their frackett,
of and tring

Recurree books - fale

- Fare thke find becoing
lunghs.

(fecbry corning)

(cash haryt)
atgns sal wasing
(offere.

(led fnes icnd
framp list.
(ltrial if corting)

(Gof werit)

(lect your maing)

Conclusion

Object-Oriented Programming in Python is a powerful paradigm that enables the creation of modular, scalable, and maintainable code. By understanding the core concepts of classes, objects, inheritance, encapsulation, and polymorphism, developers can leverage the benefits of OOP to build robust and flexible Python applications.