THE LEARNING COMMUNITY FOR TECHNOLOGY

Object Oriented Programming

Ryan Kasichainula

Lecture Goals

- Understand what is meant by “Object Oriented Programming”
- Be able to explain when should you use Object Oriented Programming
- Learn the difference between a Class and Instance
- Understand the design principles of OOP

Functional vs Object Oriented Programming

- In functional programming, you create functions and then apply them.
- In object oriented programming, you create objects which have properties and methods to alter and access those properties.
- Easier to write unit tests

Benefits of Functional Programming

- Often easier to debug
- Often easier to understand program flow

Benefits of Object Oriented Programming

- We interact with objects in the real world, so easier to understand concepts (in theory).

- Encapsulation
- Inheritance
- Polymorphism
- Abstraction

OOP Principle: Encapsulation

Encapsulation means that data is bundled inside a class with the methods that will operate on that data.

It protects the data from unintentional manipulation by outside methods.

OOP Principle: Inheritance

Inheritance means you can create a ‘subclass’ that is a more specific version of a class that inherits properties from the more general version.

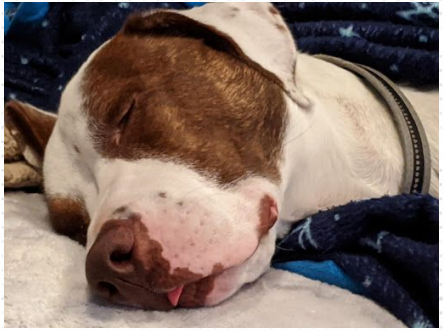
Think “animal” is a base class. “Dog” inherits from “animal”.

Inheritance Lasagna

In python, a subclass can inherit from multiple classes. It's easy to create messy hierarchies with conflicting methods.

Animal -> Predator -> Wolf -> Dog

Animal -> Cute -> Pet -> Dog



```
def blep(self):
  self.tongue_state = 'out'
```

OOP Principle: Polymorphism

Polymorphism means that a subclass can define new behavior for methods in the base class.

If animal has a base class with a method “make_sound”. Dog could return “bark”, and cat could return “meow”.

Related Concept: Duck Typing

“If it walks like a duck and quacks like a duck, it’s a duck.”

Python uses ‘duck typing’, meaning that if a class has the same attributes as a data type, it can be used as that data type.

OOP Principle: Abstraction

With abstraction, you can create an “abstract” version of a class that does not have implementations. It just has a framework which subclasses will build on.

Class and Instance

A Class definition is like a blueprint.

An Instance is an object created from that blueprint.

Many instances can be ‘instantiated’ from one Class, and each contains data that is private to that instance.

Class

Instance

Additional resources

- Embracing the Four Python Programming Styles
- Functional Vs. Object-Oriented Programming in Python
- Composition over Inheritance
- Is Object-Oriented Programming an Overrated Garbage?