Learn Python: Objected Oriented Programming

Abrar Hussain

July 15 2015

Overview of OOP Terminology

- **Class:** A prototype for an object that defines a set of attributes and methods that characterize any object that's a an instance of the class.
- **Object:** A unique instance of a class. An object has its own data members and methods (existing as an extension of the class that it's a member of).
- **Instance:** An individual object of a class. An object rect that belongs to a class Rect, for example, is an instance of the class Rect.
- **Constructor:** A special method which initializes an instance of a class.
- **Instantiation:** The creation of an instance of a class.
- **Method:** A kind of function that is defined inside of a class definition.
- **Data member:** A class variable or instance variable that holds data associated with a class and its objects.
- **Class variable:** A variable that is shared among all instances of a class. Class variables are defined inside of the class, but outside of any of the class's methods.
- **Instance variable:** A variable that is associated with only a specific instance of a class.
- **Inheritance:** Enables new objects to take on the properties of existing objects. This allows the transfer of characteristics of a class to other classes that are derived from it.

Class and Object Story

OK. So now to talk about object-oriented programming. This is a programming paradigm that views concepts as objects with data fields and related procedures called methods.

An overall concept is generalized into a class, and then specific objects of that class are called "instances of the class." The way that object-oriented programming is viewed is that the relationships among objects and classes are important for their functionality.

Classes can inherit properties and functions of other classes and can then go on to provide additional functionality to classes which extend it.

I always like to think of dogs when talking about object-oriented programming. There is a Dog, Sam, who is a Rottweiler. A Rottweiler is a Dog, so it inherits all properties of Dogs (four legs, two eyes, etc.). Sam is also a Rottweiler. So, it inherits all of the properties and functions of Rottweilers.

But, Sam is a specific Rottweiler that has a set mass, height, and scent. Tom, another Rottweiler, has a different mass, height, and scent. Not only that, but Tom can play dead!

When thinking about object-oriented programming, the benefit that comes to mind is that it allows for modular design and abstraction. Maintenance and modification of code becomes a simpler task (or so they say).