ENSF 592: Programming Fundamentals for Data Engineers

Yves Pauchard

Lecture 9: More Classes

October 4, 2019

Midterm

Oct 16, 2pm - 3:15pm. Location ENG 224

It will be a paper-based exam, 70min, 20pts. You will need a pen, and you are allowed a summary on a single page (letter, front and back). No calculators, phones, other devices.

Agenda

1. Key ideas in Ch 17 and 18

2. More on Classes

4. Preparation for next lecture

Notes: Key ideas Classes and Methods

- Python is object oriented
- methods are similar to functions but:
  - included in class definition
  - calling methods is different than functions
- methods by convention has first argument called self
- `__init__()` method is the constructor
- `__str__()` method returns a string representation of the object
- Operator overloading: `__add__()` will override `+`
- For others see http://docs.python.org/3/reference/datamodel.html#specialnames
- Type based dispatch: check type and do different things.
- There is `radd()` if object appears on the right side of the `+`
- Polymorphism
- Initialize all attributes in `__init()__`
- built-in function `vars()` to access attributes
- Interface and implementation -> more work to change the interface
- Checkout http://thinkpython2.com/code/BadKangaroo.py

Notes: Key ideas Inheritance

- Class attributes vs instance attributes
- `__lt()__` overrides the less than
- tuple comparison
- `str()` invokes `__str()__`
- vaneer method: a shiny sirface
- Inheritance: Define a new class based on an existing class
- Relationship similar but different lends itself to inheritance
- self as the first argument per convention
- Class diagrams: IS-A (hollow triangle) and HAS-A (arrow) relationships
- Method resolution order

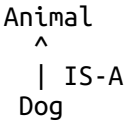
More on classes

```
class Animal:
    """ Represents an animal
        Attributes:
            kind (Class): string
            name: string
    """
    # Class attribute
    kind = 'Animal'

    def __init__(self, name):
        """ Constructor """
        self.name =  name

    def make_noise(self, noise):
        """A method: Makes animal noise"""
        print(noise)
```

More on classes

```
class Dog(Animal):
    """ Represents a Dog extends Animal
        Attributes:
            kind (Class): string
            name: string
            age: int
    """
    kind = 'Dog'
    def __init__(self, name, age):
        super().__init__(name) # calling parent class constructor
        self.age = age
    # Overriding method
    def make_noise(self, noise):
        """Makes animal noise"""
        print(noise.upper())
    # Extending class with new method
    def eat(self):
        """Not implemented yet"""
        pass
```

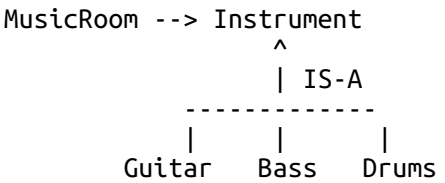
Class diagram



Assignment04: Music

See `Assignment04.pdf` and `Music.py`

Class diagram



Preparation For Next Lab/Lecture

Read/follow Ch 19 Goodies in Think Python 2e

Regular expressions

Read *Flexible Pattern Matching with Regular Expressions* (pp. 76 -- 83) in *A Whirlwind Tour of Python* by Jake VanderPlas

Lambda functions

Read *Anonymous (lambda) Functions* (pp. 44 -- 45) in *A Whirlwind Tour of Python* by Jake VanderPlas

A Whirlwind Tour of Python by Jake VanderPlas available online: https://jakevdp.github.io/WhirlwindTourOfPython/