Object Oriented Programming

- **What is object-oriented programming?**
  - A way of thinking and structuring a program
  - Instead of just writing line-by-line, make objects and then use them

- **Why do we care?**
  - NLTK (Natural Language Toolkit), a very important library for linguists
  - We've been working with objects all along! (Strings and lists)
  - It can make programming simpler
What Is An Object?

- An object is hard to define
  - Think of it as a container for data
  - Two important things: fields and methods
  - Fields: data stored in the object
  - Example: A string is a type of object
  - A data field in a string would be its length
  - A method for a string would be split
How Do You Use An Object?

- Let's say you have an object called a Person.
- Person has the data fields name and age, and method greet.

```python
doug = Person()
doug.name = "Doug Jones"
doug.age = 22
doug.greet() ← assume we have this defined
→ "Doug Jones says hello!"
```
- Does this syntax look familiar?
Why Use Objects?

- Imagine trying to compare two bottles
- These two bottles have a name, indicating what liquid is in them
- They also have a number indicating how much liquid is in them
Imagine having to change the level in one of these bottles, repeatedly, because someone is drinking from it.

With an object, you can have a method that represents this.

Instead of:
```
bottle1.level = bottle1.level - 0.25
```
```
bottle.drink(0.25)
```
How To Make Objects

class Bottle:
    def __init__(self, name, level):
        self.name = name
        self.level = level

    def displayName(self):
        print "This is", self.name

    def displayLevel(self):
        print "There are", self.level, "liters left."

    def drink(self, amount):
        self.level = self.level - amount
Some notes on that

- What is this “self” thing?
  - Objects are built from the inside, so an object needs a way to refer to itself
  - `__init__` is how you build an object – every object has one
  - The method you build an object with is called a constructor
  - It carves out a space in memory and inserts everything into the correct place
How To Use Objects

```python
>>> import Bottle
>>> coke = Bottle("Coke", 2.0)
>>> coke.name
'Coke'
>>> coke.level
2.0
>>> coke.displayName()
This is Coke
>>> coke.displayLevel()
There are 2.0 liters left.
>>> coke.drink(0.5)
>>> coke.level
1.5
>>> coke.displayLevel()
There are 1.5 liters left.
```
How To Use Objects, Part 2

```python
>>> coke = Bottle("Coke", 2.0)
>>> coke.name
'Coke'
>>> coke.level
2.0
>>> coke.displayName()
This is Coke
>>> coke.displayLevel()
There are 2.0 liters left.
>>> coke.drink(0.5)
>>> coke.level
1.5
>>> coke.displayLevel()
There are 1.5 liters left.
>>> water = Bottle("water", 2.0)
>>> water.level < coke.level
False
>>> water == coke
False
>>> water.name
'water'
>>> water.displayName()
This is water
>>> water.displayLevel()
There are 2.0 liters left.
```
Again, Why Do We Care?

- You have been working with objects all along (Strings and lists)
- Knowing how objects work helps you understand the way people program
- Many libraries which you might use if you get deeper into programming use objects (NLTK, BeautifulSoup, minidom, etc.)