

CSC150: Object Oriented Programming in Python

Kante Easley

October 5, 2010

- In many applications it makes sense to organize the application data into abstract data types (ADTs)
- Abstract data types represent a logical partitioning of the data, and a description of the operations that act on each partition.
 - The concept of a list is an ADT and we've seen many operations that are logically coupled with list data (`max`, `sort`, `pop`, ...)
 - What are some other ADT's you've seen implemented in Python?



- Class definitions are the way we implement our own ADTs in Python.
- The basic syntax for defining a class is:

```
class ClassName:  
    <function definitions>
```

- It is possible to put statements other than function definitions in a class definition.
 - We will discuss this when we talk about scope and namespaces.



- Here is a very simple class definition

```
class HelloWorldClass:  
    def say_hello(self, target="World"):  
        print "Hello %s\n" % (target)
```

- The class definition is like an ADT. It simply defines what the data is, and what operations are defined on that data.
- You need to create an *instance* in order to actually create some data or use the operations.
- To create a new instance of a class (an *object*) you simply use the class name followed by some round parentheses

```
my_instance = HelloWorldClass()
```



- When you create a new object, the `__init__()` method is implicitly called.
 - You can see this function in all the objects we've seen previously if you use the `dir()` builtin.
- You can also define your own `__init__()` method

```
class HelloWorldClass:
    def __init__(self, target="World"):
        self.target = target

    def say_hello():
        print "Hello %s\n" % (self.target)
```

- The parameter `self` should always appear first in your method definitions, and contains a copy of the instantiated object.

- In the previous example, the `target` is an *attribute* of the object.
- Each object instance has its own copy of the `target` attribute, and can be operated on independently of any other instances of the `HelloWorldClass`.

- Let's take a look at the Stack ADT, and a Python class implementing it.
- We will see how this simple ADT can be very useful in solving a problem that would be a bit tricky without organizing the data in a Stack.