course: CSC 135-01 - Computing Theory and Programming Languages

instructor: Ted Krovetz

related_notes: 2022-02-03 Python - Classes

Python Classes

W05.4 | Thursday, February 3, 2022 | 09:02 AM

Notes

1. Our course will be mostly data structures

Point Class in Java

```
// Point.java
public class Point{
        // Start with fields and they tend to be private
        private double x;
        private double y;

        // then have a constructor and one would have additional
        // ones with more specificity
        public Point(double x, double y){
                // This reference the x in the object the class variable
                // not the argument variable
                this.x = x;
                this.y = y;
        }

        // Second constructor where no (x, y) were provided
        // making the point at the origin
        public Point(){
                this.x = 0;
                this.y = 0;
        }

        // Good idea to write a toString
        public String toString(){
                // the x below refers to the one in most closest in scope,
                // so it's the one in the object (the class varible)
                return "(x=" + x + ", y=" + y + ")";
```

```
        }

        public double distance(Point other){
                // Can use the pothag theorem
                double dx = this.x - other.x;
                double dy = this.y - other.y;
                return Math.sqrt(dx*dx + dy*dy);
        }
}
```

Point Class in Python

You'll notice that you don't have to provide a type for a variable; for, Python is a dynamically typed language.

In Python there isn't any "access control" for it was a system program; where as, Java is built with security in mind.

Python uses an explicate "this" in contrast to Java's implicate implementation

Python doesn't allow a second constructor nor does it have function/method overloading, so one may utilize default values.

In Java the variables will be automatically promoted

```
"(x=" + self.x + ", y=" + self.y + ")";
```

> Can only concatenate str (not "float") to str

So you have to do

```
"(x=" + str(self.x) + ", y=" + str(self.y) + ")";
```

```
# point.py
import math

# can import particular functions from a lib/class
# from math import sqrt

class Point:
        # all constructors are called __init__
        # Python has an explicite "this" parameter
```

```
        # Python doesn't allow a second constructor nor does it have
        # function/method overloading, so one may utilize default values.
        def __init__(self, x=0.0, y=0.0):
                self.x = x
                self.y = y

        # Java toString equv.
        def __str__(self):
                return "(x=" + str(self.x) + ", y=" + str(self.y) + ")";

        def distance(self, other):
                dx = self.x - other.x
                dy = self.y - other.y
                return math.sqrt(dx*dx + dy*dy)

point_01 = Point(3.0,0.0)
point_02 = Point()

point_03 = Point(0.0,3.0)
point_04 = Point(4.0,0.0)
print(p1.distance(p2))
```

CodeStepByStep Python Class For Rectangle

```
class Rectangle:
        def __init__(self, x, y, w, h):
                self._x = x
                self._y = y
                self._width = w
                self._height = h
                if w < 0 or h < 0:
                        raise ValueError()

        def __str__(self):
                return "Rectangle [x=" + str(self._x) + \
                        ", y=" + str(self._y) + ", width=" + \
                        str(self._width) + ", height=" + \
                        str(self._height) + "]"

        @property
        def x(self):
                return self._x

        @property
        def y(self):
                return self._y
```

```
        @property
        def width(self):
                return self._width

        @property
        def height(self):
                return self._height
```