COMP1551 Core Programming

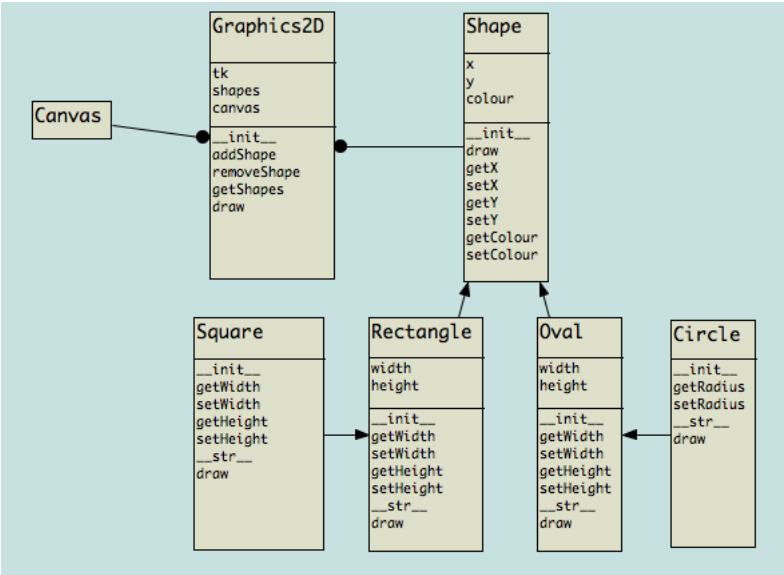
Objects Worksheet (Week 9 - w/c 24th November 2014)

November 24, 2014

Aims:

- → to understand inheritance
- → to practise implementing an object oriented design
- → to practise notating inheritance in UML

Implement a simple 2-dimensional graphics system using object oriented concepts. The graphics system consists of a base class called `Shape` and a class called `Graphics2D`. The `Shape` class is the base class for all objects that will be drawn by the graphics system. The `Graphics2D` class will manage drawing of shape in the graphics system, the system will draw using the tkinter library. Each class that extends the `Shape` class will implement a method called `draw`, the `draw` method will accept an instance of the `tkinter.Canvas` class and will draw itself on the canvas. The `Graphics2D` class will have an instance of the `tkinter.Canvas` class and a list of shapes to draw on the canvas, the `draw` method of the `Graphics2D` class will iterate through the list of shapes calling the `draw` method on each shape passing its canvas object to it. A UML diagram is provided to help in the implementation of the solution.



1. Implement the class `Shape`, the class should have;

   - an instance variables for the x and y coordinate called `x` and `y` respectively of type int. These variables will the be center point of the shape.
   - an instance variable for the colour called `colour`.
   - a constructor that accepts three parameters, the x, y and colour values.
   - getter and setter methods for all instance variables.

- a method called `draw`, the method will accept one parameter called `canvas`. The implementation of this method will be delegated to the subclasses. The method should raise a `NotImplementedError`.

2. Implement the class `Rectangle` which extends the `Shape` class, the class should have;
   - an instance variables for the width and height of the rectangle of type int, the default values should be 1 and 2 respectively.
   - a constructor that accepts 5 parameters, the parameters should be x,y, width, height and colour, the default value for colour should be blue.
   - getter and setter methods for all an instance variables.
   - an implementation of the `draw` method. The method should draw a rectangle with the parameters of the object.
   - an implementation of the `__str__` method. The method should return a string which will allow the unique identification of the rectangle.
3. Implement the class `Graphics2D`, the class should have;
   - an instance variable called `shapes` to store a list of `Shape` objects.
   - an instance variable called `canvas` which is an instance of the `tkinter.Canvas` class.
   - a constructor that accepts 3 parameters, a background colour of the `tkinter.Canvas`, the width of the Canvas and the height of the Canvas. The default parameters should be "white", 300, 300 (respectively).
   - a method called `addShape` which accepts a parameter called shape and adds it to the list `shapes`.
   - a method called `removeShape` which accepts a parameter called shape and removes the shape from the list `shapes`.
   - a method called `draw` which accepts no parameters. The method should iterate over the shapes in `shapes` calling the `draw` method on each shape, the draw method will accept a Canvas object. Don't forget to call the `pack` method on the canvas object to ensure that your shapes are displayed on screen.
4. Using a similar pattern to the question 2, implement a class that represents a;
   (a) square, the class should be called `Square` (you may wish to have `Square` extend `Rectangle` as a square is a special find of rectangle).
   (b) oval, the class should be called `Oval`.
   (c) circle, the class should be called `Circle` (you may wish to have `Circle` extend `Oval` as a circle is a special find of oval).
   (d) regular polygon, the class should be called `RegularPolygon`.
5. Write a program that allows a user to create `Shape` objects using a terminal application, the program should display the set of shapes using tkinter. The application should;
   - display a main menu prompting the user to select a shape to draw.
   - for each shape the user should be prompted to enter relavent details.
   - allow the user to delete a shape.
6. Draw a UML diagram for your system.