Practical Exercise 9 – Classes and objects

The purpose of this practical is to demonstrate how to use classes to create and use your own objects.

## Class

* A ***class*** is a template for creating an object, including the initial values of any **variables** as well as **method** definitions.
* Any number of objects can be created (i.e. instantiated) from a class.
* Classes can *inherit* variables and methods from a parent class.

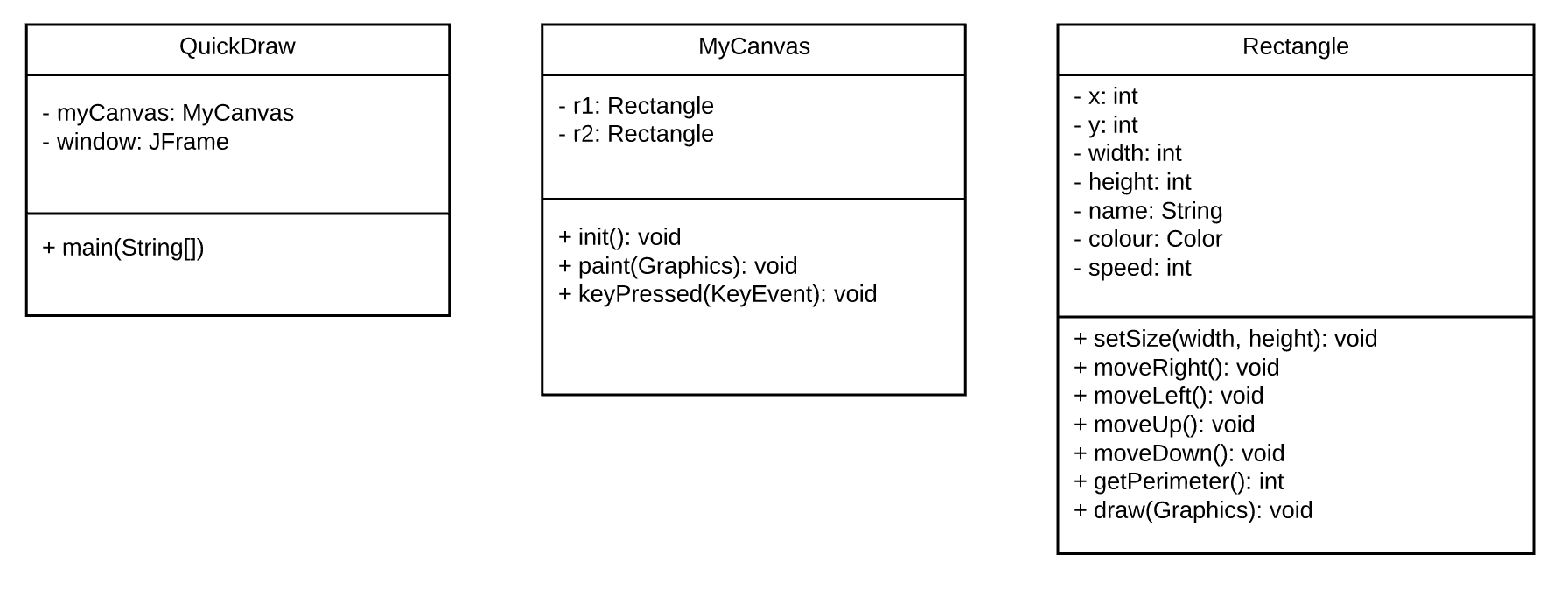
## Object

* An ***object*** is a data structure that includes values (i.e. *variables*) and ways to manipulate data (i.e. *methods*).
* An object is an *instance* of a class.

## UML class diagram

* A ***class diagram*** in the *Unified Modelling Language (UML)* describes the structure of a particular system by showing the system’s classes, their variables and methods, and the relationships between objects.

Here is a UML class diagram for the sample code in this prac…



# Task 1 – Rectangle stats

The following code defines a class used to create a program called QuickDraw that draws a rectangle that can be moved around the screen using the arrows keys. The rectangle is defined in a separate class.

1. Create two classes, *QuickDraw.java* and *Rectangle.java* in the same Java Project, and check to see that QuickDraw runs. It should display a single rectangle that can be moved left and right with th e arrow keys.
2. Fix the program so that the arrow keys allow you to move the first rectangle around the window in four directions.
3. Currently only one rectangle is displayed. Fix the program to display both rectangles, r1 and r2.
4. Fix the getPerimeter() method in the Rectangle class and use it to display the perimeters of each rectangle.
5. Add a getName() method to the Rectangle class and use it to display the names of each rectangle.
6. Add a getArea() method to the Rectangle class and use it to display the areas of each rectangle.

**import** java.awt.\*;

**import** javax.swing.\*;

**import** java.awt.event.KeyEvent;

**import** java.awt.event.KeyListener;

**class** MyCanvas **extends** JPanel **implements** KeyListener

{

Rectangle r1, r2;

**public** **void** init() {

setFocusable(**true**);

addKeyListener(**this**);

r1 = **new** Rectangle("Sarah", 50,100,20,40);

r2 = **new** Rectangle("Harry", 100,50,10,100);

setSize(300,200);

}

**public** **void** paint(Graphics g)

{

paintComponent(g);

g.drawString("Perimeter of r1: "+r1.getPerimeter(), 20, 20);

r1.draw(g);

}

**public** **void** keyPressed(KeyEvent e)

{

**switch** (e.getKeyCode())

{

**case** KeyEvent.***VK\_LEFT***:

r1.moveLeft();

**break**;

**case** KeyEvent.***VK\_RIGHT***:

r1.moveRight();

**break**;

**case** KeyEvent.***VK\_UP***:

r1.moveUp();

**break**;

**case** KeyEvent.***VK\_DOWN***:

r1.moveDown();

**break**;

}

repaint();

}

**public** **void** keyReleased(KeyEvent e)

{

}

**public** **void** keyTyped(KeyEvent e)

{

}

}

**public** **class** QuickDraw

{

**public** **static** **void** main(String[] a)

{

MyCanvas myCanvas = **new** MyCanvas();

myCanvas.init();

JFrame window = **new** JFrame();

window.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

window.setBounds(30, 30, 400, 300);

window.getContentPane().add(myCanvas);

window.setVisible(**true**);

}

}

**import** java.awt.Color;

**import** java.awt.Graphics;

**public** **class** Rectangle

{

String name;

**int** x, y, width, height;

Color colour;

**int** speed;

**public** Rectangle(String name, **int** x, **int** y, **int** width, **int** height)

{

**this**.name = name;

**this**.x = x;

**this**.y = y;

**this**.width = width;

**this**.height = height;

colour = Color.***BLUE***;

speed = 10;

}

**public** **void** setSize(**int** width, **int** height)

{

**this**.width = width;

**this**.height = height;

}

**public** **void** moveRight()

{

x = x + speed;

}

**public** **void** moveLeft()

{

x = x - speed;

}

**public** **void** moveUp()

{

// to be implemented

}

**public** **void** moveDown()

{

// to be implemented

}

**public** **int** getPerimeter()

{

**return** width + height;

}

**public** **void** draw(Graphics g)

{

g.setColor(colour);

g.drawRect(x,y,width,height);

}

}

# Task 2 – A new shape

1. Create a class for another shape (either a simple shape such as a circle, or a more complex or compound shape, like a polygon or a house!). Your class should include the attribute(s) to set the size (e.g. radius for a circle, width and height for a triangle) as well as methods for getting the size, perimeter, and area.
2. Use your class to display a shape that the user can move around the window.

# Task 3 – Variable shape

1. Add at least one method that enables a property of your shape to be changed. E.g.
   * setColour(Color) – change the shape to a specific colour.
   * setRandomColour() – change the shape to a random colour.
   * decreaseSize() – decrease the size of the shape by some amount.
   * increaseSize() – increase the size of the shape by some amount.
   * setSpeed(int speed) – change the speed of the shape.
2. Modify the keyPressed() method so that the user can change a property of a shape by pressing a particular key (or keys). E.g. when the space bar is pressed, the shape’s colour is set randomly.

# Task 4 – Constrained shape (stretch goal - bonus task)

1. Modify your code so that the user is unable to move the shape outside of the window.