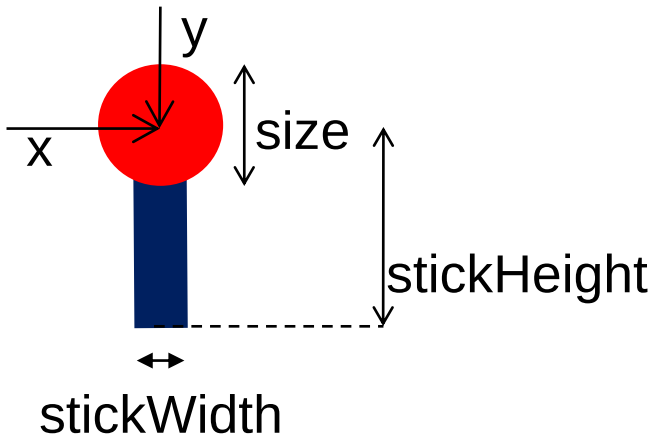
Several methods

COMP 102

Victoria University of Wellington

Improving the program: more methods

- What if we want to be able to change the width of the stick, but keep the current width as the standard width?
- We can create several methods with the same name using different parameters



Improving the program: using parameters

```
public void doDrawLollipop() {
    double size = UI.askDouble("Diameter:");
    double stickHeight =  UI.askDouble("Stick height");
    int width = UI.askDouble ("Stick width");
    this.drawLollipop(300, 180, size, stickHeight, width);
}

public void drawLollipop(double x, double y, double size, double stick, double stickWidth) {
    double left = x – size/2.0;          // left of lollipop
    double top = y – size/2.0;           // top of lollipop
    double bot =  y + stick;             // bottom of stick
    UI.setColor(Color.black);
    UI.setLineWidth(stickWidth);
    UI.drawLine(x, y, x, bot);
    UI.setLineWidth(1);
    UI.setColor(Color.red);
    UI.fillOval(left, top, size, size);
}
```

Parameters

Special variables which are given values each time the method is called.

Body of method can use the values in the parameters

Improving the program: using parameters

- What about the old method? Don't copy it! We can do better

```
public void drawLollipop(double x, double y, double size, double stick, double stickWidth) {
    double left = x – size/2.0;          // left of lollipop
    double top = y – size/2.0;           // top of lollipop
    double bot =  y + stick;             // bottom of stick
    UI.setColor(Color.black);
    UI.setLineWidth(stickWidth);
    UI.drawLine(x, y, x, bot);
    UI.setLineWidth(1);
    UI.setColor(Color.red);
    UI.fillOval(left, top, size, size);
}
public void drawLollipop(double x, double y, double size, double stick) {
    this.drawLollipop(x,y,size, stick, size/8);
}
```

Principle of good design

- Parameterising a method makes it more flexible and general
  - Allows us to call the same method with different arguments to do the same thing in different ways
  - Allows us to reuse the same bit of code
- Note:
  - Argument names do not change the semantics for the computer, therefore
  - You cannot have two methods, with the same name *and* the same types and number of arguments

```
public void drawLollipop(double x, double y, double size, double stick)

public void drawLollipop(double x, double y, double size, double total)
```

Principle of good design

- Parameterising a method makes it more flexible and general
  - Allows us to call the same method with different arguments to do the same thing in different ways
  - Allows us to reuse the same bit of code
- Note:
  - Argument names do not change the semantics for the computer, therefore
  - You cannot have two methods, with the same name *and* the same types and number of arguments. The types is what makes the methods different, not the parameter names

```
public void drawLollipop(double x, double y, double size, double stick)

public void drawLollipop(double x, double y, double size, int stick)
```