# 03 Composition Lab

## Part 1

For this part, you will use the NumberPanelExample program from the 03 Examples.zip file on moodle. The program is made up of three tabs: (i) the main setup()/draw() tab; (ii) the Button class and (iii) the NumberPanel class.

1. Study the code for the Button class. The strokeColor, fillColor and textColor properties all have default values. Change these to some alternative colours that you prefer.
2. The text property of the Button class has a default value of “OK”. However, at some point in the program, this default is changed because none of the displayed buttons show the text “OK”. Verify this by investigating the clickStart() method of the Button class. Modify clickStart() so that whenever the button is clicked, the button that was clicked on prints a message to the console stating that the button was clicked and printing out the value of text. For example:  
   A button with text property ‘5’ was just clicked!
3. Study the NumberPanel class now. This class contains a private arraylist of Button objects, and this arraylist is populated inside the constructor. The constructor contains a loop inside which the buttons are created, and adds the buttons in rows of three to produce a 3x3 button panel (with one extra row for the ‘9’ button!). Modify the program so that the buttons are arranged into rows of four instead of rows of three.
4. Add two further buttons to the panel below the existing buttons: one button with a text property of ‘+’ and one button with a text property of ‘=’. You can do this inside the constructor but after the loop.
5. Switch back to the Button class and examine the clickEnd() method. There is an important piece of code here:  
   if (state==STATE\_CLICKED)

if (panel!=null)  
 panel.buttonClicked(this);  
Comment this code out and see what happens when you run the program. Make sure you uncomment it again before proceeding!

1. Examine the buttonClicked() method of the NumberPanel class. This is the method that is called by the buttons whenever one of them is clicked. Verify this by printing out the text property of the method’s button parameter (which you can access inside the method using whichOne.text). Check to make sure that the new buttons you added in step 4 above (i.e. a ‘+’ or ‘=’), also cause an appropriate message to be printed out to the console.
2. Modify the NumberPanel example so that it becomes a calculator. The NumberPanel class should have a private sum property, and then whenever the user types in a number and clicks ‘+’, the sum should be updated and the current input cleared. Whenever the user clicks ‘=’, the current sum should be displayed.

## Part 2

For this part, you will need at least four different images, the more the better. They should be relatively small (e.g. 300x200) so you can fit several on the screen at once. You should also review the Processing functions related to text, especially the text() function: <http://www.processing.org/reference/text_.html>.

1. Consider the way that images are usually displayed in books and magazines. They are positioned somewhere on the page and they often have a numbered captioned attached to them, e.g.:

|  |
| --- |
|  |
| **Fig. 4. A silver fern frond unfurling.** |

1. Design a class for a captioned figure called Figure. The properties of a captioned figure should include (i) the image itself, (ii) the figure number, (iii) the caption text, and (iv) the position of the figure on the screen. The captioned figure should have a draw() method so it can draw itself. Ensure that the caption wraps around underneath the figure and does not go beyond the left/right bounds of the image. This means that you will need to test your class with wordy captions.
2. Design another class called Page. A page has a title, some main page text and several figures stored in an arraylist. It should also be able to draw itself.
3. Demonstrate your Page class by creating, inside setup(), two page objects about your favourite topic, each page having some text with at least two captioned images inside them.
4. Modify both the Page and Figure classes so that they now can detect whenever the user clicks inside the image. Make your Figure class print out a message whenever a figure is clicked, stating the number of the figure that was clicked.