

## INFS3202/INFS7202 Practical 3 – Dynamic Web Design

The goal of this practical is to explore to explore dynamic web site design. This practical counts 5% towards your assessment. You must present this practical to your lab tutor during your scheduled lab sessions in week 6 (week starting 12/04/2010).

This practical builds on the material you have been introduced to in this course: HTML/XHTML, CSS and Javascript. You will use what you have learnt in a more in-depth way than previous practicals.

This practical is divided into 2 Tasks:

- Build web pages that includes dynamic styling of of text (3 marks)
- Building web pages that include dynamic control of images (2 marks)

Unlike practical 2, where you were asked to download and install and use existing third – party code, in this practical you must complete the requirements using your own code. You may look at related example code, and discuss requirements with your friends,. However, **the final solution must essential be your own code**. Submitting work that does not satisfy this will be considered plagiarised work, and subject to disciplinary actions.

### Preparation

Before attempting this practical you should have a good working knowledge of HTML/XHTML, CSS and the basics of Javascript.

Before attempting this practical please ensure:

- You have covered the material in Lectures 1 - 4.
- You have reviewed the work you did in Practical 1 & 2.
- You have a good understanding of Javascript objects, events and how to manipulate the DOM (Document Object Model). Tutorials and examples on using Javascript to manipulate DOM is available from
  - <http://www.w3schools.com/js/default.asp>, in particular [http://www.w3schools.com/js/js\\_ex\\_dom.asp](http://www.w3schools.com/js/js_ex_dom.asp)

### Guidelines

In this practical, you are not required to validate your code against W3C standards. However, you are highly encouraged to not use non-standard, browser-specific features to arrive at your solutions. You can achieve this by developing coding habits and work-flow so that quick and regular validation is part of that work-flow, so that you may learn standard mark-ups and styling. **All requirements in this practical are set so that they can be completed using simple W3C standard compliant strict mark-ups (HTML 4.01 Strict or XHTML 1.0 Strict).**

## Task 1 – Dynamically Styling Text (3 Marks)

### Requirements

- a. Create a web page containing one line of text “Colour Me!”. The colour of the text must be red. See Figure 1 for sample output. Using Javascript, change the colour of the text dynamically when the cursor is placed over the text. The colours should cycle between red, yellow and blue, in that order. Each action of placing the cursor over the text should change the colour once. (1.5 marks)
- b. Add a text box at the bottom of the coloured text to allow a user to add more colours than the three defined in (a). There should be a button call “Add” that when clicked, the value typed into the text box will be added to the list of colours. A dialog box should pop up confirming that the colour has been added. See Figure 1b for sample output. When the user now places their cursor over the text, the colour should cycle between the original three colours plus the new colour added. There should be no limit to how many colours the user can add using this facility. (1.5 marks)

For a video of the requirements in action, look at the following (you will need a Shockwave Flash player installed on your browser to view this):

- [http://www.itee.uq.edu.au/~infs3202/practical/03/prac3\\_1demo.swf](http://www.itee.uq.edu.au/~infs3202/practical/03/prac3_1demo.swf)

### Guidelines:

- a. You do not have to verify that the text entered in the textbox by the user in (b) is correct. Assume that the user will enter correct colour names or codes (eg. “#123” or “#000000”).

Figure 1

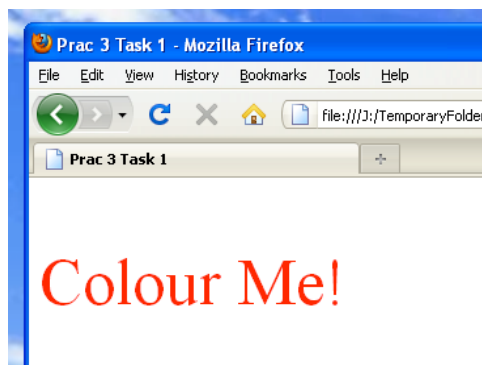
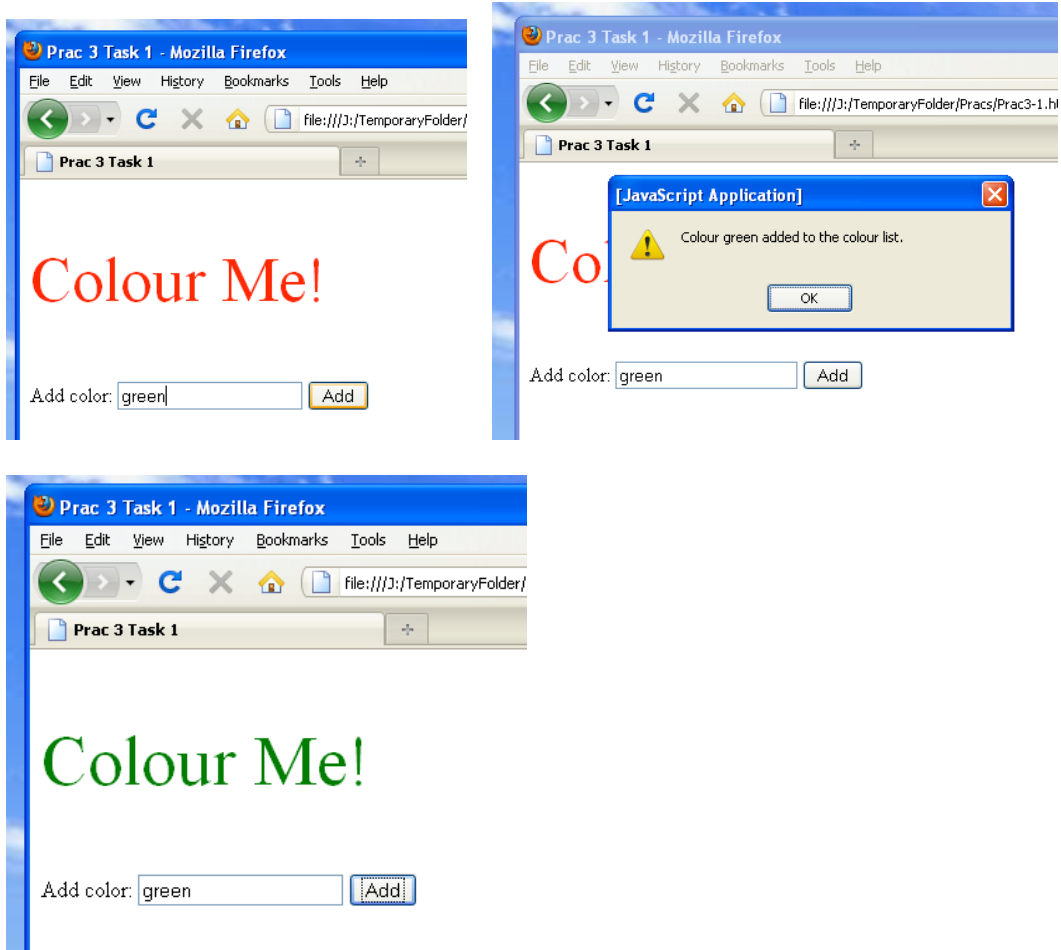


Figure 1b



## Task 2 – Dynamically Changing Image Properties (2 Marks)

### Requirements

- a. Create a web page containing one image of size 110x90px. Use any image of your choice. Using Javascript, code for the behaviour that when the user clicks on the image, the image will change its size, cycling between limits of 70px wide to 150px wide. Specifically the behaviour must be as follows: at the beginning, each user click will increase the image width by 10px, while maintaining the image's original width-height ratio. For every user click, the image's width must keep to the direction of increasing by 10px, until the width reaches a maximum of 150px. At that stage, the image must change direction and start **decreasing** by 10px for every subsequent user click. The decrease must again continue for every user click until it reaches a minimum of 70px. At this stage, the image must change direction start increasing again. This cycle between increasing and decreasing must continue as long as the user keeps clicking. There must be no limit to how many times the user can click. The image must maintain its original width-height ratio at all times. See Figure 2a for example screen outputs as the image changes sizes. (1 mark)
- b. (Challenge Task) Modify the requirements in (a) so that instead of just having one image on the page, there are at least 3 images on the page. Each image must behave as in (a), but do so **independently**. This means that the user should be able to click on any image and have it change without affecting the others. Add a "Resize all images" that resizes all images at once, but each image still changing independently according to their own current direction. This means that some images on the page could be increasing in size, and some decreasing. If some of the images have been previously clicked on and changing in one direction, clicking on the "Resize all images" button should not result in all images increasing or decreasing the same way. (1 mark)

For videos of the requirements in action, look at the following (you will need Shockwave Flash players installed on your browser):

- [http://www.itee.uq.edu.au/~infs3202/practical/03/prac3\\_2ademo.swf](http://www.itee.uq.edu.au/~infs3202/practical/03/prac3_2ademo.swf)
- [http://www.itee.uq.edu.au/~infs3202/practical/03/prac3\\_2bdemo.swf](http://www.itee.uq.edu.au/~infs3202/practical/03/prac3_2bdemo.swf)

In this task, you are required to keep proper separation of content, presentation and behaviour in your documents. This means that all style elements should be in an external style-sheet and linked from instead of embedded in your HTML/XHTML documents. Your Javascript code should be in an external JS file, and linked from instead of embedded in your HTML/XHTML document. Additionally, you may use **only one CSS style-sheet and one JS file** to complete all requirements. Specifically, you may not create multiple style-sheets and dynamically change the style-sheet link in your HTML/XHTML to complete the requirements.

The above means that your solution for (b) should not involve different CSS and JS for different number of images. Changing the number of images should only involve changing HTML/XHTML content. For assessment, create multiple HTML/XHTML documents, each containing different number of images, but linked to the same one CSS file and one JS file.

Figure 2a

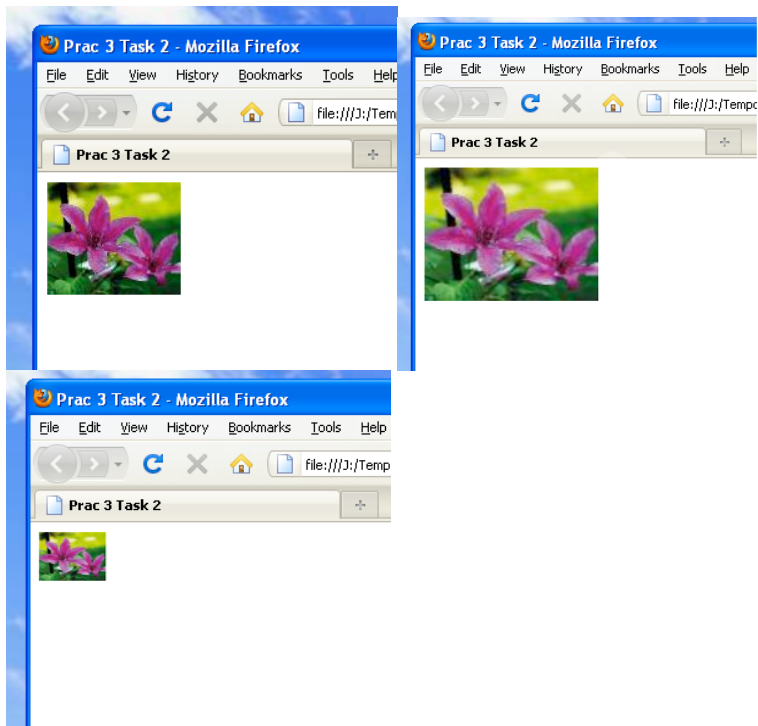


Figure 2b

