# Name: Ngo Quang Khai StudentID: s3836387

## Activity 1: Styling Text

How do we style the following text based properties in an element with CSS:

* text color  
  Ans: <span style = “color = red”> text</span>
* type of font  
  Ans: font-family=
* size of text  
  Ans: font-size= large
* **boldness** and *slopey-ness*Ans: font-weight and font-style
* underline, ~~strike-through~~ and drop-shadowsAns: font-decoration=
* spacing between letters and lines  
  Ans: word-spacing and line-height
* left, right, centered and justified alignment

Ans: text-align

When using an exotic font (ie one not found on all computers), what options do we have today?

Ans: Using google font library url

## 

## 

## Activity 2: Styling Background and Borders

How do we set the background color?

Ans: background-color

What happens if you specify a background image and a background color together?

Ans: background-img will overwrite background-color

How many different ways can we set a background image in an element?

Ans: 2 ways: by attachment or by image url

What's the difference between a linear gradient and a radial gradient?

Ans: linear gradient: the color will change from top to bottom

Radial gradient: the color will change from inside out (center of the page)

How do we set the border thickness, style and color?

Ans: using border-style for setting style

Border-color for color

Border-width for border thickness

How do we make corners rounded? What about drop-shadows?

Ans: border-radius for corners rounded

Box-shadow for drop shadows

## 

## 

## Activity 3: CSS Box Model

What is the box model? What do the properties width, padding, border and margin refer to?

Ans: We can view each elements of the layout of the page as boxes so we can design easier, this is called box model.

Width refer to the width of the box

Padding refer to the space between the content/value to the border

Border refer to the border of the box

Margin refer to the space between each box

Boxes have 4 sides (going clockwise: top, right, bottom, left), but many dimension related styles have a **compact** version where only 1, 2 or 3 values are specified. Explain how the browser interprets each variant.

Ans: CSS will group those 1-3 values cases into special cases e.g. 2 values will be understand as top/bottom and left/right

How wide will this element's visible box be: <div id='fixed-content'></div> (CSS shown below):

|  |
| --- |
| #fixed-content  {  width: 100px;  border: grey solid 5px;  padding: 20px;  margin: 20px;  }  <div id='fixed-content'></div> |

Ans: 100+5\*2+20\*2+20\*2 = 190 px

What difference does the new CSS3 box-sizing:border-box setting make to the box sizing model?

Ans: box-sizing will change the width set by the user to be the total width of all the padding, border, margin.

How wide will this element's visible box be: <div id='adaptable-content'></div> (CSS shown below), In particular, how large will it be displayed on a large screen (eg on a 27" iMac) or a small screen (eg an older "palette" monitor?):

|  |
| --- |
| #adaptable-content  {  width: 50%;  min-width: 600px;  max-width: 1800px;  margin: 0 auto;  }  <div id='adaptable-content'></div> |

Ans: The wide of the element will be 50% the width of the device the user is using If it’s between 600px and 1800px

How does the CSS3 calc() and box-sizing styles help us fine-tune box dimensions?

|  |
| --- |
| #fine-tuned-content  {  box-sizing: border-box;  width: calc(100% - 60px);  }  <div id='fine-tuned-content'></div> |

Ans: calc() function calculate and resize the width of the box to save space for other elements

Box-sizing made all of the calculation under the hood for user.

Those functions help developer to fine-tune box dimensions to most devices.

## Activity 4: Floating Page Elements, Inline and Flexible Blocks



Block elements usually take up the entire width of the parent element, but there are times we want to float an image to the left or right (eg these mugs) and have text float around it (eg these paragraphs).



When we set an element to float:left or float:right any block elements that come after it are placed in the gaps. This might be what you want, but more often than not it isn't. Paragraphs will break up and page footers will look like inline elements (see below).

|  |
| --- |
| Footer content | T&C | Links | Map | Uh oh, look at this image → |

To "stop the floats", you will need an element that has the style clear:both to restore the normal flow layout.

<http://www.w3schools.com/cssref/pr_class_float.asp>

<http://www.w3schools.com/cssref/pr_class_clear.asp>

CSS3 allows us to have the best of both worlds with floating blocks. Elements with display:inline-block will have inline flow properties but retain a box shape.

|  |
| --- |
| #keep-content-together  {  display:inline-block;  }  <div id='keep-content-together'></div> |

Internally, elements will not be reshaped and don't get squashed like the above paragraphs do.

<http://www.w3schools.com/cssref/pr_class_display.asp>

A new advanced family of styles (supported in modern browsers only) is flex and grid which allows you to create flexible elements that resize according to rules laid down by their parent element and their sibling elements. These styles are likely to dominate page layout design in the near future.

[https://www.w3schools.com/cssref/css3\_pr\_flex.asp](http://www.w3schools.com/cssref/css3_pr_flex.asp)

<https://www.w3schools.com/cssref/pr_grid.asp>

## Activity 5: Positioning

The default style for positioning is static. What does this mean?

Ans: the object is in its original assigned position and does not able to move

What other positioning styles are there? What happens to the original space that the element occupied in each case?

Ans:

* Static
* Relative: the original space will be blank and still be occupied by it
* Absolute: the original space will be completely blank
* Fixed: the original space will be completely blank
* Sticky: the original space will be blank

How do top, bottom, left and right help us to adjust the position of a non-static element?

Ans: It is like a coordinate that help to place the object where we want

## 

## Activity 6: Display and Visibility

To make elements "disappear" we have at least two options:

|  |
| --- |
| #gollum, #bilbo, #frodo, #harry, .friends-of-harry  {  visibility: hidden;  /\* to make visible, use visibility: visible; \*/  }  #tardis  {  display: none;  /\* to re-display, use display: block; OR inline-block; OR inline; etc \*/  } |

What is the difference between the two styles above?

Ans: Visibility: hidden: just hide the object from the user, the original space of the object is still there

Display:none will not display the objectS, there will be no missing spaces like visibility