CSS Basics

Introduction to CSS

- A CSS (cascading style sheet) file allows you to separate your web sites (X)HTML content from it's style.
- you use your (X)HTML file to arrange the content, but all of the presentation (fonts, colors, background, borders, text formatting, link effects & so on...) are accomplished within a CSS.
- you can use the CSS, either internally or externally.

Internal Stylesheet

- you are simply placing the CSS code within the <head></head> tags of each (X)HTML file you want to style with the CSS. The format for this is shown in the example below.
-

```
<head>
<title><title>
<style type="text/css">
CSS Content Goes Here
</style>
</head>
<body>
```

- each (X)HTML file contains the CSS code needed to style the page. Meaning that any changes you want to make to one page, will have to be made to all.
- This method can be good if you need to style only one page, or if you want different pages to have varying styles.

External Stylesheet

- An external CSS file can be created with any text or HTML editor such as "Notepad" or "Dreamweaver".
- A CSS file contains no (X)HTML, only CSS. save it with the .css file extension.
- Link to the file externally by placing one of the following links in the head section of every (X)HTML file you want to style with the CSS file.

```
<link rel="stylesheet" type="text/css" href="Path To stylesheet.css" />
```

- Or you can also use the @import method as shown below

<style type="text/css">@import url(*Path To stylesheet.css*)</style>

example:

<head>
<title><title>

gkagombe@jkuat.ac.ke

<link rel="stylesheet" type="text/css"href="style.css" />
</head>
<body>

Or

<head>
<title><title>
<style type="text/css"> *@import url(Path To stylesheet.css)*
</style>
</head>
<body>

- By using an external style sheet, all of your (X)HTML files link to one CSS file in order to style the pages.
- if you need to alter the design of all your pages, you only need to edit one .css file to make global changes to your entire website.

Here are a few reasons this is better.

- Easier Maintenance
- Reduced File Size
- Reduced Bandwidth
- Improved Flexibility

Cascading Order

You can have both internal, external, and inline styles.

Inline Styles

- in a way they defeat the purpose of using CSS in the first place.
- They are defined right in the (X)HTML file along side the element you want to style. See example below.

  Some red text

- Inline styles will NOT allow the user to change styles of elements or text formatted this way

- All the various methods will cascade into a new "pseudo" stylesheet in the following order:

1. Inline Style (inside (X)HTML element)
2. Internal Style Sheet (inside the <head> tag)
3. External Style Sheet

Users with Disabilities

The use of external style sheets also can benefit users that suffer from disabilities. For instance, a user can turn off your stylesheet or substitute one of their own to increase text size, change colors and so on.

Power Users

Swapping stylesheets is beneficial not only for users with disabilities, but also power users who are particular about how they read Web documents.

Browser Issues

You will discover as you delve farther into the world of CSS that all browsers are not created equally, to say the least. CSS can and will render differently in various browsers causing numerous headaches.

CSS Syntax

The syntax for CSS is different than that of (X)HTML markup.
It consists of only 3 parts.

```
selector { property: value }
```

- The selector is the (X)HTML element that you want to style.
- The property is the actual property title, and
- the value is the style you apply to that property.

Each selector can have multiple properties, and each property within that selector can have independent values.
The property and value are seperated with a colon and contained within curly brackets.
Multiple properties are seperated by a semi colon.
Multiple values within a property are sperated by commas, and if an individual value contains more than one word you surround it with quotation marks. As shown below.

```
body {
        background: #eeeeee;
        font-family: "Trebuchet MS", Verdana, Arial, serif;
}
```

-The final result sets the body color to light grey, and sets the font to ones that most users will have installed on their computer.

Inheritance

When you nest one element inside another, the nested element will inherit the properties assigned to the containing element.
Unless you modify the inner elements values independently.
For example, a font declared in the body will be inherited by all text in the file no matter the containing element, unless you declare another font for a specific nested element.

```
body {font-family: Verdana, serif;}
```

Now all text within the (X)HTML file will be set to Verdana.
If you wanted to style certain text with another font, like an h1 or a paragraph then you could do the following.

```
h1 {font-family: Georgia, sans-serif;}
p {font-family: Tahoma, serif;}
```

Now all <h1> tags within the file will be set to Georgia and all <p> tags are set to Tahoma, leaving text within other elements unchanged from the body declaration of Verdana.
NOTE: There are instances where nested elements do not inherit the containing elements properties. For example, if the body margin is set to 20 pixels, the other elements within the file will not inherit the body margin by default.

```
body {margin: 20px;}
```

Combining Selectors
You can combine elements within one selector in the following fashion.

```
h1, h2, h3, h4, h5, h6 {
        color: #009900;
        font-family: Georgia, sans-serif;
}
```

- in the above code, I have grouped all the header elements into one selector. Each one is separated by a comma.
- The final result of the above code sets all headers to green and to the specified font. If the user does not have the first font I declared it will go to another sans-serif font the user has installed on their computer.

Comment tags
Comments can be used to explain why you added certain selectors within your css file. So as to help others who may see your file, or to help you remember what you we're thinking at a later date. You can add comments that will be ignored by browsers in the following manner.

```
/* This is a comment */
```

You will note that it begins with a / (forward slash) and then an * (asterisks) then the comment, then the closing tag which is just backward from the opening tag * (asterisks) then the / (forward slash).

CSS Classes

- The class selector allows you to style items within the same (X)HTML element differently.
- the style can be overwritten by changing out stylesheets. You can use the same class selector again and again within an (X)HTML file.

To put it more simply, this sentence you are reading is defined in my CSS file with the following.

```
p {
font-size: small;
color: #333333
}
```

say that I wanted to change the word "sentence" to green bold text, while leaving the rest of the sentence untouched. I would do the following to my (X)HTML file.

```
<p>
To put it more simply, this <span
class="greenboldtext">sentence</span> you are reading is styled in my CSS file by the
following.
</p>
```

Then in my CSS file I would add this style selector:

```
.greenboldtext{
font-size: small;
color: #008080;
font-weight: bold;
}
```

The final result would look like the following:

To put it more simply, this **sentence** you are reading is styled in my CSS file by the following.

Note

- a class selector begins with a (.) period.
- A selector can be named whatever name you want. Though it is good practice to use selector names that are descriptive.
- You can reuse the "greenboldtext" class as many times as you want.

CSS IDs

- IDs are similar to classes, except once a specific id has been declared it cannot be used again within the same (X)HTML file.

gkagombe@jkuat.ac.ke

The main container for this page is defined by the following.

```
<div id="container">
Everything within my document is inside this division.
</div>
```

I have chosen the id selector for the "container" division over a class, because I only need to use it one time within this file.
Then in my CSS file I have the following:

```
#container{
width: 80%;
margin: auto;
padding: 20px;
border: 1px solid #666;
background: #ffffff;
}
```

Note:

- the id selector begins with a (#) number sign instead of a (.) period, as the class selector does.

CSS Divisions

- the (X)HTML side of using CSS.

Divisions

- Divisions are a block level (X)HTML element used to define sections of an (X)HTML file.
- It can contain all the parts that make up your website. Including additional divisions, spans, images, text and so on.

You define a division within an (X)HTML file by placing the following between the <body></body> tags:

```
<div>
Site contents go here
</div>
```

- To add some style to it. You can do that in the following fashion:

```
<div id="container">
Site contents go here
</div>
```

The CSS file contains this:

```
#container{
width: 70%;
```

```
margin: auto;
padding: 20px;
border: 1px solid #666;
background: #ffffff;
}
```

Now everything within that division will be styled by the "container" style rule, I defined within my CSS file.

- A division creates a linebreak by default.
- You can use both classes and IDs with a division tag to style sections of your website.

CSS Spans

- very similar to divisions except they are an inline element versus a block level element. No linebreak is created when a span is declared.
- You can use the span tag to style certain areas of text, as shown in the following:

```
<span class="italic">This text is italic</span>
```

Then in my CSS file:

```
.italic{
font-style: italic;
}
```

The final result is: *This text is italic.*

CSS Margins
Inherited: No

-it is a property that declares the margin between an (X)HTML element and the elements around it.

-The margin property can be set for the top, left, right and bottom of an element. For example;

```
margin-top: length percentage or auto;
margin-left: length percentage or auto;
margin-right: length percentage or auto;
margin-bottom: length percentage or auto;
```

As you can also see in the above example you have 3 choices of values for the margin property

- length
- percentage
- auto

You can also declare all the margins of an element in a single property as follows:

```
margin: 10px 10px 10px 10px;
```

If you declare all 4 values as I have above, the order is as follows:

1. top
2. right
3. bottom
4. left

If only one value is declared, it sets the margin on all sides. (see below)

```
margin: 10px;
```

If you only declare two or three values, the undeclared values are taken from the opposing side. (see below)

```
margin: 10px 10px; /* 2 values */
margin: 10px 10px 10px; /* 3 values */
```

You can set the margin property to negative values.

```
margin: -10px;
```

If you do not declare the margin value of an element, the margin is 0 (zero).

- Elements like paragraphs have default margins in some browsers, to combat this set the margin to 0 (zero).

```
p {margin: 0;}
```

Note: You do not have to add px (pixels) or whatever units you use, if the value is 0 (zero).
You can see in the example below, the elements for this site are set to be 20px (pixels) from the body

```
body{
margin: 20px;
background: #eeeeee;
font-size: small;
font-family: Tahoma, Arial, "Trebuchet MS", Helvetica, sansserif;
text-align: left;
}
```

CSS Padding
Inherited: No

- Padding is the distance between the border of an (X)HTML element and the content within it.
- Most of the rules for margins also apply to padding, except there is no "auto" value, and negative values cannot be declared for padding.

```
padding-top: length percentage;
padding-left: length percentage;
padding-right: length percentage;
padding-bottom: length percentage;
```

- Note from the example, you have 2 choices of values for the padding property
  - Length
  - percentage

You can also declare all the padding of an element in a single property as follows:

```
padding: 10px 10px 10px 10px;
```

If you declare all 4 values as I have above, the order is as follows:

1. top
2. right
3. bottom
4. left

If only one value is declared, it sets the padding on all sides.

```
#container{
width: 70%;
margin: auto;
padding: 30px;
border: 1px solid #666;
background: #ffffff;
}
```

CSS Text Properties
Inherited: Yes
Color

You can set the color of text with the following:

```
Color:value
```

Possible values are

color name - example:(red, black...)
hexadecimal number - example:(#ff0000, #000000)
RGB color code - example:(rgb(255, 0, 0), rgb(0, 0, 0))

Letter Spacing

- adjust the space between letters in the following manner.
- Setting the value to 0, prevents the text from justifying. You can use negative values.

```
letter-spacing: value;
```

- Possible values are
    - normal
    - length

Example:
T h e s e l e t t e r s a r e s p a c e d a t 5 p x .

Text Align

You can align text with the following:
text-align: value;

Possible values are
- left
- right
- center
- justify

Text Decoration

- decorate text with the following: text-decoration: value;

Possible values are
- none
- underline
- overline
- line through
- blink

Text Indent

indent the first line of text : text-indent: value;

Possible values are
- length
- percentage

Text Transform

- control the size of letters: text-transform: value;

- Possible values are
    - none
    - capitalize
    - lowercase
    - uppercase

Examples:
This First Letter In Each Word Is Capitalized, Though It Is Not In My File.
THIS TEXT IS ALL UPPERCASE, THOUGH IT IS ALL LOWERCASE IN MY FILE.
this text is all lowercase. though it is all uppercase in my file.

White Space

- control the whitespace: white-space: value;

Possible values are

- normal
- pre
- nowrap

Word Spacing

- adjust the space between words. You can use negative values.

  text-transform: value;

Possible values are

- normal
- length

CSS Text Properties

- Inherited: Yes
  - Color
  - Letter Spacing
  - Text Align
  - Text Decoration
  - Text Indent
  - Text Transform
  - White Space
  - Word Spacing

CSS Font Properties

Inherited: Yes

Font

- This property can set the style, weight, variant, size, line height and font: Example:

```
font: italic bold normal small/1.4em Verdana, sans-serif;
```

Font-Family

- set what font will be displayed in an element :
- 2 choices for values:
  - family-name
  - generic family
- example:

```
font-family: Verdana, sans-serif;
```

NOTE: If you set a family name, best to add the generic family at the end. This is a priortized list, if the user does not have the specified font name it will use the same generic family.

Font Size

- set the size of the text used in an element by using the fontsize property.
- choices for values:
  - xx-large
  - x-large
  - larger
  - large
  - medium
  - small
  - smaller
  - x-small
  - xx-small
  - length
  - % (percent)

Further read on these:
- What size text should I use in my css by Paul O'B
- Dive into accessibility - Font Sizes

```
font-size: value;
```

Font Style

- set the style of text
- Possible values are
  - normal
  - itailc
  - oblique

```
font-style: value;
```

Font Variant

- the font-variant property
- Possible values are
  - normal
  - small-caps

```
font-variant: value;
```

Font Weight

- control the weight of text in an element with the font-weight property

```
font-weight: value;
```

- Possible values are
  - lighter
  - normal
  - 100
  - 200
  - 300
  - 400
  - 500
  - 600
  - 700
  - 800

- 900
- Bold
- bolder

CSS Anchors, Links and Pseudo Classes

Below are the various ways you can use CSS to style links.

```
a:link {color: #009900;}
a:visited {color: #999999;}
a:hover {color: #333333;}
a:focus {color: #333333;}
a:active {color: #009900;}
```

- The 1st sets the color of a link when no event is occurring
- The 2nd - the color a link changes to, after the user has visited that url
- 3rd -color of link as the user places their mouse pointer over the link
- 4th - primarilly for the same purpose as the last one, -as the user tabs through the links, user not using mouse.
- 5th - sets the color a link changes to as it is pressed.

Pseudo Classes

- say you want your links in the content area to have a different color to the links in the left or right column of your webpage.

You can do this in the following fashion:

```
#content a:link {color: #009900;}
#content a:visited {color: #999999;}
#content a:hover {color: #333333;}
#content a:focus {color: #333333;}
#content a:active {color: #009900;}
```

- assuming that you have your main content in a division named"content"

Then for the links in a column you could use the following:

```
#column a:link {color: #009900;}
#column a:visited {color: #999999;}
#column a:hover {color: #333333;}
#column a:focus {color: #333333;}
#column a:active {color: #009900;}
```

- assumes the name of the column division.

can also be accomplished by declaring a class instead of an id.
in this case you will need to add a class to each link

```
a.column:link {color: #009900;}
a.column:visited {color: #999999;}
```

```
a.column:hover {color: #333333;}
a.column:focus {color: #333333;}
a.column:active {color: #009900;}
```

Then in the (X)HTML file for each link

```
<a class="column" href="" title="">some link text</a>
```

But, there is still yet an easier way

```
.column a:link {color: #009900;}
.column a:visited {color: #999999;}
.column a:hover {color: #333333;}
.column a:focus {color: #333333;}
.column a:active {color: #009900;}
```

ON THE HTML file

```
<div class="column"><a href="" title="">some link text</a></div>
```

NOTE: Almost any property that can be used to style text and fonts can be used to style links also.

CSS Backgrounds
Inherited: No
Background

You can style the background of an element in one declaration with the background property.

```
background: #ffffff url(path_to_image) top left no-repeat fixed;
```

Values:

- attachment
- color
- image
- position
- repeat

Or you can set each property individually

Background Attachment

If you are using an image as a background. You can set whether the background scrolls with the page or is fixed when the user scrolls down the page with the background-attachment property

```
background-attachment: value;
```

Values:

- fixed
- scroll

Background Color

You can specifically declare a color for the background of an element using the background-color property.

```
background-color: value;
```

Values:

- color name
- hexadecimal number
- RGB color code
- transparent

Background Image

the background-image property. background-image: url(path_to_image);

Values:

- url
- none

Background Position

- position an image used for the background .

background-position: value;

Values:

- top left
- top center
- top right
- center left
- center center
- center right
- bottom left
- bottom center
- bottom right
- x-% y-%
- x-pos y-pos

Background Repeat

- if an image set as a background of an element is to repeat (across=x and/or down=y) the screen using the background-repeat property.

background-repeat: value;

Values:

- no-repeat
- repeat
- repeat-x
- repeat-y

CSS Borders

Inherited: No

Border

You can set the color, style and width of the borders around an element in one declaration by using the border property. Example;

border: 1px solid #333333;

Values:

- color
- style
- width

Or you can set each property individually

Border Color

- color of a border - border-color property.

```
border-color: value;
```

Values:

- color name
- hexadecimal number
- RGB color code
- transparent

Border Style

style of a border independently with the border-style property-

```
border-style: value;
```

Values:

- dashed
- dotted
- double
- groove
- hidden
- inset
- none
- outset
- ridge
- solid

border-width: value;- width of a border Values:Length,Thin,Medium,Thick

or individually,

border-bottom: 1px solid #333333;

border-bottom-color: value;

List Style Type

- the type of bullet ordered and unordered lists use-the list-style-type property

Values

- disc
- circle
- square
- decimal
- lower-roman
- upper-roman
- lower-alpha
- upper-alpha
- none

```
list-style-type: value;
```

CSS Classification

Inherited: No

Clear

- control if an element allows floated elements to its sides with the clear property

gkagombe@jkuat.ac.ke

Values:

- none
- both
- left
- right

Now, what does all that mean?

None- the default setting, floated elements can appear on either side of the element set to clear: none;
Both- no floated elements to appear on either side of the element set to clear: both;
Left- no floated elements to appear to the left side of the element set to clear: left;
Right - floated elements to appear to the right side of the element set to clear: right;

Clip

- control how much of an element is visible with the clip property

Values:

- auto
- shape

Currently the only shape recognized by the clip property is rect (rectangle)

Display

- control how an element is displayed with the display property

Values:

- block
- inline
- list-item
- none

Now, what does all that mean?

Block-Creates a line break before and after the element
Inline- No line break is created
List Item -Creates a line break before and after the element and adds a list item marker
None - Makes an element not display on the page

Float

- changes how text and or images within an element are displayed

```
float: value;
```

Values:

- left
- right
- none

Now, what does all that mean?

Left- The image/text is displayed to the left of the parent element
Right - The image/text is displayed to the right of the parent element

None - There is no change in the way the image/text is displayed

CSS Positioning
Inherited: No
Position
The position property changes how elements are positioned on your webpage.
Values:

- static
- relative
- absolute
- fixed

Now, what does all that mean?
Static -is by default the way an element will appear in the normal flow of your (X)HTML file. to declare a position of static is no different than not declaring itl.
Relative - places the element in the normal flow of your (X)HTML file and then offsets it by some amount using the properties left, right, top and bottom. This may cause the element to overlap other elements that are on the page, which of course may be the effect that is required.
Absolute - removes the element from the normal flow of your (X)HTML file, and positions it to the top left of it's nearest parent element that has a position declared other than static. If no parent element with a position other than static exists then it will be positioned from the top left of the browser window.
Fixed- is the same as absolute except the parent element is always the browser window. It makes no difference if the fixed element is nested inside other positioned elements. an element that is positioned with a fixed value, will not scroll with the document. It will remain in it's position regardless of the scroll position of the page.

- When positioning elements with relative, absolute or fixed values the following properties are used to offset the element:
- top
- left
- right
- bottom