

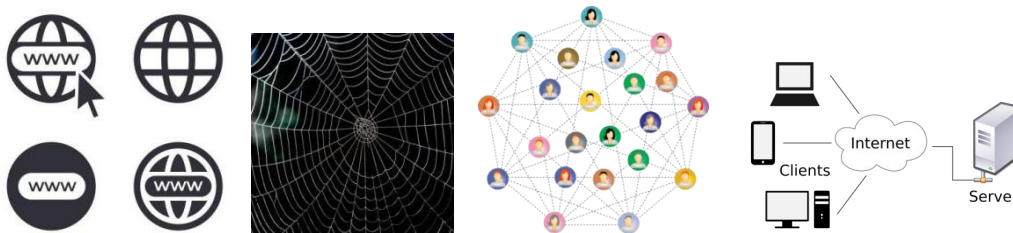
# Chapter 1: Introductions To Web

## World Wide Web (WWW)



The **World Wide Web (WWW)**, commonly known as **the Web**, is an information system where documents and other web resources are identified by **Uniform Resource Locators(URL)** or **Search Engines**, which may be interlinked by **hypertext**, and are accessible over the **internet** through the **Web Browser**.

The **Web**, is a global information infrastructure. A universal library and a collection of **web server**.

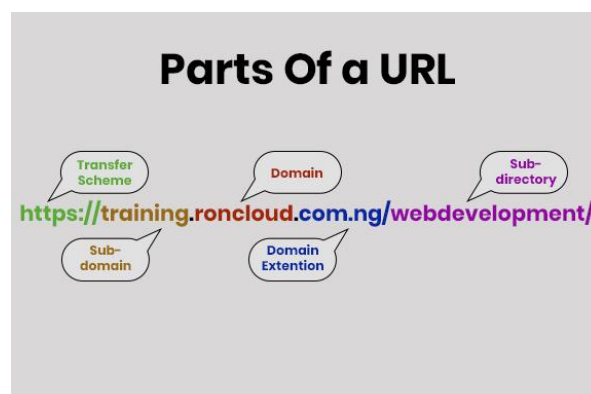


## Uniform Resource Locator(URL)

These are words of patterns, use to refers to website. In other words, they are called domain names.

### Examples of Uniform Resource Locator(URL) are

- `http://www.sitename.com/`
- `http://www.sitename.com/index.html`
- `www.example.com/contactus.php`
- `www.example.com`
- Etc.



## Search Engines

These are tools on the web used to locate searched keywords or phrases.

### Examples of Search Engines are

- Google
- Bing
- Yahoo!
- Baidu
- DuckDuckGo
- Yandex
- Ask.com
- Ecosia
- AOL
- Internet Archive
- Wolfram Alpha
- Lycos
- Etc.



## Web Browser

These are tools or applications, used for accessing web resources. The purpose of a web browser is to read web resources and display them.

### Examples of Web Browser are

- Internet Explorer.
- Google Chrome.
- Mozilla Firefox.
- Safari.
- Opera.
- Konqueror.
- Lynx.
- Tor Browser.
- UC Browser.
- Brave Browser.
- Etc.



## Internet



The **Internet** serves as the means of communications of multiple computers including clients and servers globally.

## Web Server



The Web Server, is a server software or device used for storing web resources which can be accessible by users via the internet using a **web browser**.

The primary function of a web server is to store, process and deliver web resources to clients. The communication between clients and server takes place using the Hypertext Transfer Protocol(HTTP).

## Web Resources

These are files and documents that are accessible on the web.

### Examples of Web Resources are

- Web-pages
- Images
- Documents
- Videos
- Audios
- Etc.

### Tools used in creating web resources are



- HTML
- CSS
- JAVASCRIPT
- PHP
- MySQL
- Etc.

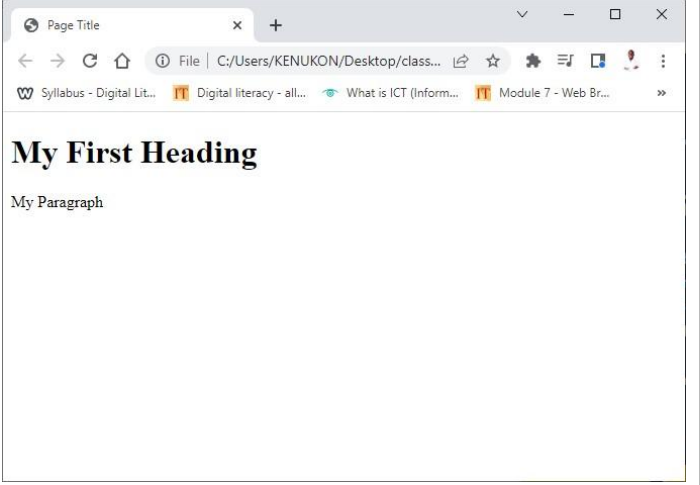
## What is HTML?

**HTML** is a markup language for describing web documents (web pages). it can also be referred to as the building block of a web-pages.

The word **HTML** stands for **H**yper **T**ext **M**arkup **L**anguage.

- A Markup language is a set of markup tags.
- HTML documents are described by HTML tags.
- Each HTML tags describes different document content.

### Example 1

<pre>&lt;!DOCTYPE html&gt; &lt;html&gt;   &lt;head&gt;     &lt;title&gt;Page Title&lt;/title&gt;   &lt;/head&gt;   &lt;body&gt;     &lt;h1&gt;My First Heading&lt;/h1&gt;     &lt;p&gt;My Paragraph&lt;/p&gt;   &lt;/body&gt; &lt;/html&gt;</pre>	
---	---

### Example 1 Explained

- The **DOCTYPE** declaration defines the document type to be **HTML**
- The text between **<html>** and **</html>** describes an **HTML** document
- The text between **<head>** and **</head>** provides information about the document
- The text between **<title>** and **</title>** provides title for the document
- The text between **<body>** and **</body>** describes the visible page content
- The text between **<h1>** and **</h1>** describes a heading
- The text between **<p>** and **</p>** describes a paragraph

## HTML Tags

**HTML tags** are keywords (tag names) surrounded by angle brackets:  
**<tagname>content</tagname>**

- HTML tags normally come in pairs like **<p>** and **</p>**
- The first tag in a pair is the **start tag(<p>)**, the second tag is the **end tag(</p>)**.
- The **end tag** is written like the **start tag**, but with a slash before the tag name.
- The **start tag** is often called the **opening tag**.

- The **end tag** is often called the **closing tag**.

## Tools used in creating and displaying HTML files are

- Code Editors
- Web Browsers

### Code Editors

**Code Editors** are computer programs or applications used in creating and editing programming file and documents.

There are two types of Editors

- Plain Text Editors.
- Integrated Development Environment (IDE).

#### Plain Text Editors

These are editors that edit plain text. Most of these plain text editors are embedded in **operating system(OS)** installed in computers.

##### Examples are

- Notepad
- TextMate
- TextEdit
- Etc.

#### Integrated Development Environment (IDE)

These are editors that has **graphical user interface(GUI)**, which enable users/programmers to reduce coding, has **built-in automation** tools and helps to **debug easily**.

##### Examples are

- Microsoft Visual Studio Code
- Android Studio
- Microsoft Visual Studio
- Adobe Dream-weaver
- Sublime Text
- Eclipse
- IntelliJ
- Komodo
- PyCharm
- Etc.

Stated in the example 1 above, those code can be written using any Editor either **plain text editor** or **IDE**.

To differentiate the type of program your writing, you have to save the file with an extension of **".html"** or **".htm"** for **HTML** files. Otherwise, it will be saved and treated as the default extension of the editor, which might not be visible via the **web browser**.

##### Examples are



- Index.html
- contactus.html
- about\_us.htm
- login.htm
- Etc.

## Web Browser

As said earlier, the purpose of web browser is to read web resources (like HTML) and display them.

The browser does not display the HTML tags, but uses them to determine how to display the document.

## What is CSS?

**CSS** is a style-sheet language that describes the presentation of an **HTML** document. It can also be referred to as the beautifier or styling of a web-page.

The word **CSS** stands for **C**ascading **S**tyle **S**heets

- **CSS** describes how **HTML** elements are to be displayed on screen, paper or in other devices
- **CSS** saves a lot of works, it can control the layout of multiple web pages all at once

## Why use CSS?

CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.

### Example 2

```
h1 {color:blue; align:right; border:1px solid;}  
.nav {  
    font:Bookman Old bold;  
    background-color: yellow;  
    margin: 10px;  
}  
#footer:a {  
    padding-top: 5px;  
    padding-bottom: 15px;  
    border-color: black;  
}
```

### Example 2 Explained

- The **"h1"**, **".nav"**, **"#footer:a"** are known as **selector**. Which refers to the element in an **HTML** file or documents.

- Every other thing between the curly bracket “{” and “}” are known as **declarations**. Which refers to the effect that will take place on the **selectors** been mentioned
- The word(s) before the colon “:” in the declarations are known as **property**, and the one(s) after the colon “:” are known as the **value**.
- The selector(s) points to the **HTML elements** you want to style.
- The **declaration block** contains one or more declaration(s) separated by semicolon(s).
- Each declaration includes a CSS property name and a value, separated by a colon.
- A CSS declaration always ends with a semicolon “;” and declaration blocks are surrounded by curly braces.

## Assessments

1. What is the difference between the web and the internet?
2. What is the difference between uniform resource locator(URL) and search engine?
3. Explain a Mark-up language.
4. Explain a Style-sheet language.

## Quote of the Day.

***“If you can’t fly, run. If you can’t run, walk. If you can’t walk, crawl. Whatsoever you do, don’t stop moving”***

# Chapter 2: HTML

## What is HTML?

As said earlier, **HTML** is a markup language for describing web documents (web pages). it can also be referred to as the building block of a web-pages.

- The word **HTML** stands for **H**yper **T**ext **M**arkup **L**anguage.
- A Markup language is a set of markup tags.
- HTML documents are described by HTML tags.
- Each HTML tags describes different document content.

### Note!

We have two types of web pages.

- Static web-page
- Dynamic web-page

### Static Web-page

A **static web page** contains fixed contents, each is coded in **HTML** and displays the same information to every visitor. **Static pages** are the basic types of web pages and are easiest to create. In other words, they are **web-page** that doesn't requires visitor's data input.

### Dynamic Web-pages

A **dynamic web-page** interact with users and visitors. There are regularly known as the **client** and **server-side web-page**. They are coded with **web programming languages**, but displays with **HTML**. In other words, they are **web-page** that allows visitors to input data.

## HTML Elements

- **HTML documents** are made up by **HTML elements**.
- **HTML elements** are written with **opening tags**, **content** in-between and ends with **closing tags**. **<tagname>content</tagname>**
- The **HTML element** is everything from the **opening tag** to the **closing tag**.

### Note!

Some HTML elements do not have closing tag.



## Empty HTML Elements

- HTML elements with no content are called empty elements.
- **<br>** is an empty element without content or closing tag (the **<br>** tag defines a line break).
- Empty element can be "closed" in the opening tag like this: **<br/>**

### Examples of an HTML Elements

Opening Tag	Element Content	Closing Tag
<code>&lt;h1&gt;</code>	My First Heading	<code>&lt;/h1&gt;</code>
<code>&lt;p&gt;</code>	My First Paragraph	<code>&lt;/p&gt;</code>
<code>&lt;br/&gt;</code>		

## Nested HTML Elements

HTML elements can be nested. That is, placing element(s) inside another element(s).

All HTML documents consist of nested HTML elements.

### Example

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>
  <body>
    <h1>My First Heading</h1>
    <p>My Paragraph</p>
  </body>
</html>
```

### Example Explained

- The **<html>** element defines the **whole document**. It has an opening tag **<html>** and a closing tag **</html>**.
- The element **content** is another HTML element (the **<body>** element and everything after it before the closing tag **</html>**).
- The **<body>** element defines the **document body**. It has an opening tag **<body>** and a closing tag **</body>**. The element **content** has other HTML elements (**<h1>** and **<p>** elements and everything within and after them before the closing tag **</body>**).

- The **<h1>** element defines a **heading**. It has an opening tag **<h1>** and a closing tag **</h1>**. The element **content** is: My first heading.
- The **<p>** element defines a **paragraph**. It has an opening tag **<p>** and a closing tag **</p>**. The element **content** is: My first paragraph.

## Don't Forget the End Tag

Some HTML elements will display correctly, even if you forget the end tag:

### Example

```
<!DOCTYPE html>
<html>
  <body>
    <h1>This is a header
    <p>This is a paragraph
  </body>
</html>
```

### Example Explained

- The example above works in all browsers, because the closing is considered optional.
- Never rely on this. It might produce unexpected results and/or errors if you forget the end tag.

### HTML Tip: Use lowercase Tags

HTML tags are not case sensitive: **<P>** means the same as **<p>**. The HTML5 standard does not require lowercase tags, but W3C recommends lowercase in HTML4 and demands lowercase for strict documents type like XHTML.

## HTML Attributes

Attributes provide additional information about HTML elements

- HTML elements can have **attributes**.
- Attributes provide **additional information** about an element
- Attributes are always specified in the **opening tag**
- Attributes come in names/value pairs like: **name= "value"**

### The Lang Attributes

The document language can be declared in the **<html>** tag. The language is declared in the **lang** attribute.

Declaring a language is important for accessibility applications (screen readers) and search engines:

### Example

```
<!DOCTYPE html>
<html lang = "en-US">
  <body>
    <h1>My First Heading </h1>
    <p>My First paragraph. </p>
  </body>
</html>
```

### Example Explained

The **lang** is an **attribute name** attached to **<html>** tag, the **en-US** is the **attribute value** which is in double quotation marks.

The first two letters specify the language (en). If there is a dialect, use two more letters like(US).

### The Title Attribute

The title attribute specifies extra information about an element. The information is most often shown as a tool-tip text when the mouse moves over the element.

### Example

```
<p title="About Roncloud"> The Roncloud Technologies, is a software development company and institute that trains people to become software developer and a renowned IT personnel.</p>
```

### Example Explained

The **<p>** element has a **title** attribute. The value of the attribute is **"About Roncloud"** When you move the mouse over the element, the title will be displayed as a tool-tip.

### The href Attribute

HTML links are defined with the **<a>** tag. The link address is specified in the **href** attribute:

### Example

```
<a href="http://www.roncloud.com.ng"> Roncloud Technologies Website </a>
```

### Example Explained

- The **<a>** elements tag serve as links. In other to declare the direction of the link, we use the **href attribute**.

- The **href** is the attribute name, the **"http://www.roncloud.com.ng"** is the value of the attribute.
- When you click on the link, it takes you to the mention page of the value slated in the href attribute.

## Single or Double Quotes?

Double style quotes are the most common in HTML, but single style can also be used. In some situations, when the attribute value itself contains double quotes, it is necessary to use single quotes:

### Example

**<p title='Roncloud "Developers" '>**

Or vice versa:

**<p title="Roncloud 'Developers' ">**

### Example Explanation

The attribute value has quote(s) in-between. It's advisable to use different quote to differentiate so as to avoid errors.

### Attribute Tips: Use lowercase

- W3C recommends lowercase in HTML4, and demands lowercase for strict document type like XHTML.
- The HTML5 standard does not require lowercase attribute name. The **title="About Roncloud"** can also be written as **TITLE="About Roncloud"**
- The HTML5 standard does not require quotes around attribute values.
- The **href="http://www.roncloud.com.ng"** can also be written as **HREF=http://www.roncloud.com.ng**.

### Note!

All web browser displays web pages differently. It's advisable as a web designer or programmer to test run on multiple browser, in other to see how it displays on different browser.

## HTML Comment

HTML comments help in creating documentations for codes which serves as a guide for other developers to understand why certain codes are written.

### Example

**<!-- This is a comment -->**

**<p>This is a paragraph.</p>**

**<!-- Remember to add more information here -->**

Author: Olugbenga Raymond (**www.roncloud.com.ng**)

## NOTE!

Comments are not displayed in the browser, but they can help in documenting your HTML source code.

It can also be used to disallow a certain code(s) from execution.

## Assessment

1. Write a letter to a mentor of you, telling him or her about how you will love to be mentored by him or her using HTML.
2. List and define each Doctype of HTML and their functions.
3. Explain the benefits of attributes on HTML elements.
4. What does W3C stands for, and its function

## Quote of the Day

***It's a foolish man's idea, as a developer, to think he/she can code without having any bugs(errors).***

# Chapter 3: CSS

## What is CSS?

As said earlier, **CSS** is a stylesheet language that describes the presentation of an HTML document.

- **CSS** stands for **C**ascading **S**tyle **S**heets.
- CSS describes how HTML elements are to be displayed on screen, paper or in other devices.
- CSS saves a lot of works, it can control the layout of **multiple web pages** all at once.

## Why Use CSS?

CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.

### CSS Solved a Big Problem

HTML was never intended to contain tags for formatting or styling a web page!

HTML was created to describe the content of a web page like:

<h1> This is a heading </h1>

<p> This is a paragraph. </P>

When tags like **<font>** and color attributes were added to the **HTML3.2 specifications**, it started a nightmare for web developers. Development of large websites, where fonts and color information were added to every single page became a long and expensive process.

To solve this problem, the **World Wide Web Consortium (W3C)** created **CSS**.

### CSS Saves a Lot of Work!

- The style definitions are normally saved in external **.css** files
- With an external stylesheet file, you can change the look of an entire website by changing just one file.

### CSS How To

When a browser reads a style sheet, it will format the HTML document according to the information in the style sheet.



# Ways to insert CSS

There are three ways of inserting a style sheet:

- External style sheet
- Internal Style sheet
- Inline styles

## Inline styles

An inline style may be used to apply a unique style for a single element.

To use inline style, add the style attribute to the relevant element. The style attribute can contain any CSS property.

### Example

```
<h1 style="color: blue; margin-left:30px;">This is a heading.</h1>
```

## Internal style sheet

An internal style sheet may be used, if one single page has a unique style.

Internal styles are within the **<style> element**, nested inside **<head> elements** section of an HTML page:

### Example

```
<style>  
    body { background-color: linen;}  
    h1 { color: maroon; margin-left: 40px; }  
</style>
```

## External style sheet

With an external style you can change the look of an entire website by changing just one file!

Each page must include a reference to the external style sheet file inside the **<link>** element which goes inside the **<head>** section

### Example

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

### Note!

An external style sheet can write on any text editor. The file should not contain any HTML tags. The style sheet file must be saved with a **.css** extension.

## style.css

```
Body { background-color:lightblue;}  
h1 { color: navy; margin-left: 20px;}
```

## Cascading Order

What style will be used when there is a more than one style specified for an HTML element?

Generally speaking, we can say that all the styles will cascade into a new “virtual” style sheet by the following rule, where number one has highest priority:

1. Inline style
2. Internal Stylesheet
3. External Stylesheet
4. Browser Default

So, an inline style has the highest priority, which means that it will override a style defined inside the **<style>** tag or in an external style sheet or a browser default value.

### NOTE!

You can disrupt the cascading order by adding **!important** in front of your preferred line of code to make the certain line of code move to the top of the cascading order

## CSS Syntax and Selectors

- A CSS rule-set consist of a selector and a declaration block:
- The selector points to the HTML element you want to style.
- The declaration includes a CSS property name and a value separated by a colon.
- A CSS declaration block are surrounded by curly braces.

In the following example all **<p>** element will be center-aligned with a red text color.

### Example

```
p { color: red; text-align: center; }
```

### The Id Selector

- The Id selector uses the id attribute of an HTML element to select a specific element
- The Id of an element should be unique within a page, so the id selector is used to select one unique element!
- To select an element with a specific id, write a **hash(#)** character, followed by the id of an element.

- The style rule below will apply to an HTML element with id= "part"

### Example

```
#part {  
    Text-align: center; color:red;  
}
```

## The Class selector

- The class selector selects elements with a specific class attribute.
- To select elements with a specific class, write a period (.) character followed by the name of the class.

In the example below. All HTML elements with class = "center" will be affected

### Example

```
.center { text-align: center; color: red;}
```

You can also specify that only specific HTML elements should be affected by a class. In the example below only **<p>** element with **class="center"** will be affected.

### Example

```
p.center {  
    text-align: center;  
    color: red;  
}
```

HTML elements can also refer to more than one class.

In the example below, the **<p>** element will be styled according to **class="center"** and to **class="large"**:

### Example

```
<p class= "center large">This paragraph refers to two classes. </p>
```

## Grouping Selectors

If you have elements with the same style definitions, like this

```
h1 { text-align: center; color: red; }  
h2 { text-align: center; color: red; }  
h3 { text-align: center; color: red; }  
p { text-align: center; color: red; }
```

you can simply separate the selector with a comma (,) and use the same declaration to represent all. Like this

Author: Olugbenga Raymond ([www.roncloud.com.ng](http://www.roncloud.com.ng))

**h1, h2, h3, p { text-align: center; color: red; }**

## CSS Comments

Comments are used to explain the code and may help when you edit the source code at a later date

Comments are ignored by browsers.

A CSS comment starts with **/\*** and ends with **\*/**. Comments can also span multiple lines:

```
p {  
    Color: red;  
    /* This is a single-line comment */  
    Text-align: center;  
}  
  
/* This is a  
multi-line  
comment */
```

### Note!

Comments don't display on browsers, and has no special effect on document but to give description of document.

## Assignment

1. Explain why CSS was introduced to web development.
2. Explaining in details the CSS syntax.
3. Style the letter writing in the Chapter 2 Assignment.
4. Explain in details the cascading order and how it can be disrupted

## Quotes of the Day

***"Tomorrow belong to those who can solve the problems of today."***

***By Chief Michael Orobosa***

# Chapter 4: HTML/CSS Links, Images & Source Path

## HTML Links

Links are found in nearly all web pages. Links allow users to click their way from page to page.

### HTML Links – Hyperlinks

A hyperlink is a text or an image you can click on and jump to another document.

### HTML Links – Syntax

In HTML, links are defined with the `<a>` tag:

**`<a href="url">link text</a>`**

#### Example

**`<a href="http://www.roncloud.com.ng/">Visit our Official Website</a>`**

- The **href attribute** specifies the destination address **`http://www.roncloud.com.ng`**
- The link text is the visible part (**`Visit our Official Website`**).
- Clicking on the link text, will send you to the specified address.

#### Note!

- The link text does not have to be text. It can be an HTML image or any other HTML element.
- Without a trailing slash on subfolder addresses, you might generate two requests to the server. Many servers will automatically add a trailing slash to the address, and then create a new request.

## Local Links

The example above used an absolute URL (A full web address).

A local link (link to the same web site) is specified with a relative URL (without `http://www....`). more on this in this chapter

#### Example

**`<a href="contactus.php">Contact Us</a>`**

## HTML Links – Colors

When you move the mouse over a link, two things will normally happen:

- The mouse arrow will turn into a little hand
- The color of the link element will change

Author: Olugbenga Raymond (**`www.roncloud.com.ng`**)

By default, a link will appear like this (in all browsers):

- An unvisited link is underlined and blue
- A visited link is underlined and purple
- An active link is underlined and red
- You can change the default colors, by using styles:

### Example

```
<style>
a:link {color:green; background-color:transparent; text-decoration:none}
a:visited {color:pink; background-color:transparent; text-decoration:none}
a:hover {color:red; background-color:black; text-decoration:underline}
a:active {color:yellow; background-color:transparent; text-decoration:underline}
</style>
```

## HTML Links – The Target Attribute

The target attribute specifies where to open the linked document.

This example will open the linked document in a new browser window or in a new tab:

### Example

```
<a href="http://www.roncloud.com.ng/" target="_blank">Visit RonCloud!</a>
```

### Target Value Description

#### **\_blank**

Opens the linked document in a new window or tab

#### **\_self**

Opens the linked document in the same frame as it was clicked (this is default)

#### **\_parent**

Opens the linked document in the parent frame

#### **\_top**

Opens the linked document in the full body of the window frame name Opens the linked document in a named frame

If your webpage is locked in a frame, you can use **target="\_top"** to break out of the frame:



## HTML Links – Create a Bookmark

- HTML bookmarks are used to allow readers to jump to specific parts of a Web page.
- Bookmarks are practical if your website has long pages.
- To make a bookmark, you must first create the bookmark, and then add a link to it.
- When the link is clicked, the page will scroll to the location with the bookmark.

### Example

- First, create a bookmark with the id attribute:

**<h2 id="tips">Useful Tips Section</h2>**

- Then, add a link to the bookmark ("Useful Tips Section"), from within the same page:

**<a href="#tips">Visit the Useful Tips Section</a>**

- Or, add a link to the bookmark ("Useful Tips Section"), from another page:

**<a href="html\_tips.html#tips">Visit the Useful Tips Section</a>**

## HTML Images Syntax

In HTML, images are defined with the **<img>** tag.

The **<img>** tag is empty, it contains attributes only, and does not have a closing tag.

The **src** attribute specifies the URL (web address) of the image:

****

### The alt Attribute

- The **alt attribute** specifies an **alternate text** for **an image**, if the image cannot be displayed.
- The **alt attribute** provides **alternative information** for an image if a user for some reason cannot view it (because of slow connection, an error in the **src attribute**, or if the user uses a screen reader).
- If a browser cannot find an image, it will display the alt text:

### Example

****

The alt attribute is required. A web page will not validate correctly without it.

## HTML Screen Readers

A screen reader is a software program that can read what is displayed on a screen. Screen readers are useful to people who are blind, visually impaired, or learning disabled.

## Note!

Screen readers can read the alt attribute.

## Image Size – Width and Height

You can use the style attribute to specify the width and height of an image. The values are specified in pixels (use **px** after the value):

### Example

```

```

Alternatively, you can use width and height attributes. Here, the values are specified in pixels by default:

### Example

```

```

## Width and Height or Style?

Both the width, height, and style attributes are valid in the latest HTML5 standard. We suggest you use the style attribute. It prevents styles sheets from changing the original size of images:

### Example

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      img {
        width:100%;
      }
    </style>
  </head>
  <body>
    
    
  </body>
</html>
```

## Images in Another Folder

- If not specified, the browser expects to find the image in the same folder as the web page.
- However, it is common to store images in a sub-folder. You must then include the folder name in the **src attribute**: you will about this in the Source Path section below

## Example

```

```

## Images on Another Server

Some web sites store their images on image servers.

Actually, you can access images from any web address in the world:

## Example

```

```

## Animated Images

The GIF standard allows animated images:

## Example

```

```

Note that the syntax of inserting animated images is no different from non animated images.

## Using an Image as a Link

To use an image as a link, simply nest the **<img>** tag inside the **<a>** tag:

## Example

```
<a href="default.asp">
  
</a>
```

### Note!

Add "border:0;" to prevent IE9 (and earlier) from displaying a border around the image.

## Image Floating

Use the CSS float property to let the image float.

The image can float to the right or to the left of a text:

## Example

```
<p>
  
  The image will float to the right of the text.
</p>
<p>
  
  The image will float to the left of the text.
</p>
```

## Image Maps

- Use the **<map>** tag to define an image-map.
- An image-map is an image with clickable areas.
- The name attribute of the **<map>** tag is associated with the **<img>**'s **usemap attribute** and creates a relationship between the image and the map.
- The **<map>** tag contains a number of **<area>** tags, that defines the clickable areas in the image-map:

### Example

```

<map name="planetmap">
    <area shape="rect" coords="0,0,82,126" alt="Sun" href="sun.htm">
    <area shape="circle" coords="90,58,3" alt="Mercury" href="mercur.htm">
    <area shape="circle" coords="124,58,8" alt="Venus" href="venus.htm">
</map>
```

## Source Path

Source path are location of resource in a giving directory either locally or globally. They are two type of paths

- Relative Path
- Absolute Path

### Relative Path

As the name implies, relative paths are paths that can be active depends on how a certain file is related to the other files it referred or referring to in the same project achieve.

## Assessment

- Explain, and give practical examples of pseudo-class and elements.
- List and explain the different types of image.
- Create an image map of Africa, specifying clickable link of various countries.

## Quote of the Day

If you can think it, you can do it.

# Chapter 5: HTML/ CSS Table, List and Display

## HTML Tables

Number	First Name	Last Name	Points
1	Melvin	Daniel	96
2	Daniel	Yusuf	88
3	John	Doe	97
4	Jane	Doe	85

### Defining HTML Tables Example Example explained:

```
<table>
  <tr>
    <th>Number</th>
    <th>First Name</th>
    <th>Last Name</th>
    <th>Points</th>
  </tr>
  <tr>
    <td>1</td>
    <td>Melvin</td>
    <td>Daniel</td>
    <td>96</td>
  </tr>
  <tr>
    <td>2</td>
    <td>Daniel</td>
    <td>Yusuf</td>
    <td>88</td>
  </tr>
  <tr>
    <td>3</td>
    <td>John</td>
    <td>Doe</td>
    <td>97</td>
  </tr>
  <tr>
    <td>4</td>
    <td>Jane</td>
    <td>Doe</td>
    <td>85</td>
  </tr>
</table>
```

- Tables are defined with the **<table>** tag.
- Tables are divided into table rows with the **<tr>** tag.
- Table rows are divided into table data with the **<td>** tag.<sup>1</sup>
- A table row can also be divided into table headings with the **<th>** tag.

#### Note!

- Table data **<td>** are the data containers of the table.
- Tables can also be divided into table columns with the **<tr>** Tag
- They can contain all sorts of HTML elements like text, images, lists, other tables, etc.
- By default, all major browsers display table headings as bold and centered:

## An HTML Table with a Border Attribute

If you do not specify a border for the table, it will be displayed without borders. A border can be added using the border attribute:

### Example

```
<table border: 1px solid black;>
  .
  .
  .
</table>
```

### Note!

- The border attribute is on its way out of the HTML standard! It is better to use CSS.
- To add borders, use the CSS border property:

### Example

```
table, th, td {
  border: 1px solid black;
}
```

- Remember to define borders for both the table and the table cells.
- An HTML Table with Collapsed Borders .
- If you want the borders to collapse into one border, add CSS border-collapse:

### Example

```
table, th, td {
  border: 1px solid black;
  border-collapse: collapse;
}
```

## An HTML Table with Cell Padding

Cell padding specifies the space between the cell content and its borders. If you do not specify a padding, the table cells will be displayed without padding. To set the padding, use the CSS padding property:

### Example

```
table, th, td {
  border: 1px solid black;
  border-collapse: collapse;
}
th, td {
  padding: 15px;
}
```



## An HTML Table with Border Spacing

Border spacing specifies the space between the cells.

To set the border spacing for a table, use the CSS border-spacing property:

### Example

```
table {  
    border-spacing: 5px;  
}
```

### Note!

If the table has collapsed borders, border-spacing has no effect.

## Table Cells that Span Many Columns

To make a cell span more than one column, use the **colspan attribute**:

### Example

```
<table style="width:100%">  
  <tr>  
    <th>Name</th>  
    <th colspan="2">Telephone</th>  
  </tr>  
  <tr>  
    <td>Bill Gates</td>  
    <td>555 77 854</td>  
    <td>555 77 855</td>  
  </tr>  
</table>
```

## Table Cells that Span Many Rows

To make a cell span more than one row, use the **rowspan attribute**:

### Example

```
<table style="width:100%">  
  <tr>  
    <th>Name:</th>  
    <td>Bill Gates</td>  
  </tr>  
  <tr>  
    <th rowspan="2">Telephone:</th>  
    <td>555 77 854</td>  
  </tr>  
  <tr>  
    <td>555 77 855</td>  
  </tr>  
</table>
```

## An HTML Table with a Caption

To add a caption to a table, use the `<caption>` tag:

### Example

```
<table style="width:100%">
  <caption>Monthly savings</caption>
  <tr>
    <th>Month</th>
    <th>Savings</th>
  </tr>
  <tr>
    <td>January</td>
    <td>$100</td>
  </tr>
  <tr>
    <td>February</td>
    <td>$50</td>
  </tr>
</table>
```

### Note!

- The **<caption>** tag must be inserted immediately after the **<table>** tag.
- A Special Style for One Table
- To define a special style for a special table, add an id attribute to the table:

### Example

```
<table id="t01">
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
    <th>Points</th>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
</table>
```

Now you can define a special style for this table:

```
table#t01 {
  width: 100%;
  background-color: #f1f1f1;
}
```

And add more styles:

```
table#t01 tr:nth-child(even) {
  background-color: #eee;
}
```

```
table#t01 tr:nth-child(odd) {
    background-color: #fff;
}
table#t01 th {
    color: white; background-color: black;
}
```

## HTML Lists

Example of an unordered list and an ordered list in HTML:

### Unordered List:

- Item
- Item
- Item
- Item

### Ordered List:

1. First item
2. Second item
3. Third item
4. Fourth item

## Unordered HTML Lists

An unordered list starts with the **<ul>** tag. Each list item starts with the **<li>** tag. The list items will be marked with bullets (small black circles):

### Example

```
<ul>
    <li>Coffee</li>
    <li>Tea</li>
    <li>Milk</li>
</ul>
```

### Unordered HTML Lists – The Style Attribute

A style attribute can be added to an unordered list, to define the style of the marker:

Style	Description
list-style-type: disc;	The list items will be marked with bullets
list-style-type: circle;	The list items be marked with circles
list-style-type: square;	The list items will be marked with squares
list-style-type: none;	The list items will not be marked

### Disc:

```
<ul style="list-style-type: disc;">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

### Square:

```
<ul style="list-style-type: square;">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

### Circle:

```
<ul style="list-style-type: circle;">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

### None:

```
<ul style="list-style-type: none;">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

## Ordered HTML Lists

An ordered list starts with the **<ol>** tag. Each list item starts with the **<li>** tag. The list items will be marked with numbers:

### Example

```
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

## Ordered HTML Lists – The Type Attribute

A type attribute can be added to an ordered list, to define the type of the marker:

Type	Description
type="1"	This list items will be numbered with numbers(default).
type="A"	This list items will be numbered with uppercase letters
type="a"	This list items will be numbered with lowercase letters
type="I"	This list items will be numbered with uppercase roman numbers
type="i"	This list items will be numbered with lowercase roman numbers

### Numbers:

```
<ol type="1">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

### Uppercase Letters:

```
<ol type="A">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

### Lowercase Letters:

```
<ol type="a">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

## Uppercase Roman Numbers:

```
<ol type="I">
    <li>Coffee</li>
    <li>Tea</li>
    <li>Milk</li>
</ol>
```

## Lowercase Roman Numbers:

```
<ol type="i">
    <li>Coffee</li>
    <li>Tea</li>
    <li>Milk</li>
</ol>
```

## HTML Description Lists

- HTML also supports description lists.
- A description list is a list of terms, with a description of each term.
- The **<dl>** tag defines the description list, the **<dt>** tag defines the term (name), and the **<dd>** tag describes each term:

### Example

```
<dl>
    <dt>Coffee</dt>
    <dd>- black hot drink</dd>
    <dt>Milk</dt>
    <dd>- white cold drink</dd>
</dl>
```

## Nested HTML Lists

List can be nested (lists inside lists):

### Example

```
<ul>
    <li>Coffee</li>
    <li>Tea
        <ul>
            <li>Black tea</li>
            <li>Green tea</li>
        </ul>
    </li>
    <li>Milk</li>
</ul>
```

### Note!

List items can contain new list, and other HTML elements, like images and links, etc.

## Horizontal Lists

HTML lists can be styled in many different ways with CSS.

One popular way, is to style a list to be displayed horizontally:

## Example

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      ul#menu li {
        display:inline;
      }
    </style>
  </head>
  <body>
    <h2>Horizontal List</h2>
    <ul id="menu">
      <li>HTML</li>
      <li>CSS</li>
      <li>JavaScript</li>
      <li>PHP</li>
    </ul>
  </body>
</html>
```

With a little extra style, you can make it look like a menu:

## Example

```
ul#menu {
  padding: 0;
}
ul#menu li {
  display: inline;
}
ul#menu li a {
  background-color: black;
  color: white;
  padding: 10px 20px;
  text-decoration: none;
  border-radius: 4px 4px 0 0;
}
ul#menu li a:hover {
  background-color: orange;
}
```

## CSS Layout – The Display Property

- The display property is the most important CSS property for controlling layout.
- The display property specifies if/how an element is displayed.
- Every HTML element has a default display value depending on what type of element it is.
- The default display value for most elements is block or inline.



## Block-level Elements

- A block-level element always starts on a new line and takes up the full width available (stretches out to the left and right as far as it can).
- The `<div>` element is a block-level element.

### Examples of block-level elements:

**`<div>`**

**`<h1>` – `<h6>`**

**`<p>`**

**`<form>`**

**`<header>`**

**`<footer>`**

**`<section>`**

## Inline Elements

- An inline element does not start on a new line and only takes up as much width as necessary.
- This is an inline **`<span>`** element inside a paragraph.

### Examples of inline elements:

**`<span>`**

**`<a>`**

**`<img>`**

## Display: none;

`display: none;` is commonly used with JavaScript to hide and show elements without deleting and recreating them.

## Override The Default Display Value

- As mentioned, every element has a default display value. However, you can override this.
- Changing an inline element to a block element, or vice versa, can be useful for making the page look a specific way, and still follow the web standards.
- A common example is making inline `<li>` elements for horizontal menus:

### Example

```
li {  
    display: inline;  
}
```

## Note!

- Setting the display property of an element only changes how the element is displayed, NOT what kind of element it is. So, an inline element with display: block; is not allowed to have other block elements inside it.
- The following example displays <span> elements as block elements:

### Example

```
span {  
    display: block;  
}
```

## Hide an Element – display:none or visibility: hidden?

- Hiding an element can be done by setting the display property to none.
- The element will be hidden, and the page will be displayed as if the element is not there:

### Example

```
h1.hidden {  
    display: none;  
}
```

visibility:hidden; also hides an element.

However, the element will still take up the same space as before. The element will be hidden, but still affect the layout:

### Example

```
h1.hidden {  
    visibility: hidden;  
}
```

## Quote of the day

If you can't fly, run. If you can't run, walk. If you can't walk, crawl. Whatsoever you do, don't stop moving.

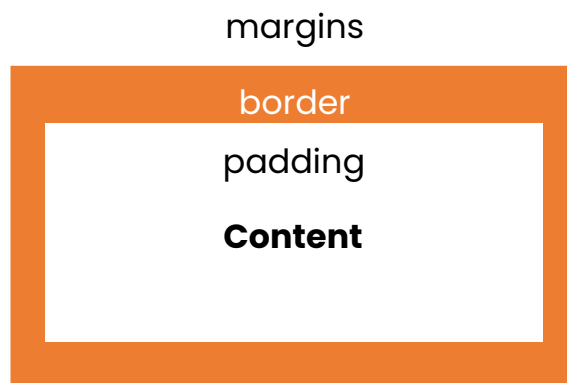
# Chapter 5: HTML / CSS Box Model, forms and Media

## CSS Box Model

Every elements in are represented as boxes

The CSS box model is essentially a box that wraps around every HTML elements. It consists of: margins, borders, padding, and the actual content.

The image below illustrates the box model:



### Explanation of the different parts:

- Content – The content is where text and images appear.
- Padding – The padding is the clear area inside the border around and before the content.
- Border – The border is the line or shape that separates the margins and padding and also surrounds the content.
- Margin – The margin is the clear area outside the border.

## The <form> Element

HTML forms are used to collect user input.

The <form> element defines an HTML form:

<form>

.

form elements

.

</form>

HTML forms contain form elements.

Form elements are different types of input elements, checkboxes, radio buttons, submit buttons, and more.

## The <input> Element

The <input> element is the most important form element.

The <input> element has many variations, depending on the type attribute.

Here are the types used in this chapter:

Type	Description
Text	Define normal text input
Radio	Defines Radio Button input (for selecting one of many choices)
Submit	Defines a submit button (For submitting the form)

### Text Input

<input type="text"> defines a one-line input field for text input:

#### Example

```
<form>
  First name:<br>
  <input type="text" name="firstname"><br>
  Last name:<br>
  <input type="text" name="lastname">
</form>
```

This is how it will look like in a browser:

First name:

Last name:

#### Note:

The form itself is not visible. Also note that the default width of a text field is 20 characters.

### Radio Button Input

<input type="radio"> defines a radio button.

Radio buttons let a user select ONE of a limited number of choices:

#### Example

```
<form>
  <input type="radio" name="gender" value="male" checked> Male<br>
  <input type="radio" name="gender" value="female"> Female<br>
  <input type="radio" name="gender" value="other"> Other
</form>
```

This is how the HTML code above will be displayed in a browser:

- ☐ Male
- ☐ Female
- ☐ Other

## The `<select>` Element (Drop-Down List)

The `<select>` element defines a drop-down list:

### Example

```
<select name="cars">
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="fiat">Fiat</option>
  <option value="audi">Audi</option>
</select>
```

The `<option>` elements defines the options to select.

The list will normally show the first item as selected.

You can add a `selected` attribute to define a predefined option.

### Example

```
<option value="fiat" selected>Fiat</option>
```

## The `<textarea>` Element

The `<textarea>` element defines a multi-line input field (a text area):

### Example

```
<textarea name="message" rows="10" cols="30">
  The cat was playing in the garden.
</textarea>
```

## The Submit Button

`<input type="submit">` defines a button for submitting a form to a form-handler.

The form-handler is typically a server page with a script for processing input data.

The form-handler is specified in the form's action attribute:

### Example

```
<form action="action_page.php">  
    First name:<br>  
    <input type="text" name="firstname" value="Mickey"><br>  
    Last name:<br>  
    <input type="text" name="lastname" value="Mouse"><br><br>  
    <input type="submit" value="Submit">  
</form>
```

### The Action Attribute

The action attribute defines the action to be performed when the form is submitted. The common way to submit a form to a server, is by using a submit button.

Normally, the form is submitted to a web page on a web server.

In the example above, a server-side script is specified to handle the submitted form:

**<form action="action\_page.php">**

If the action attribute is omitted, the action is set to the current page.

### The Method Attribute

The method attribute specifies the HTTP method (GET or POST) to be used when submitting the forms:

**<form action="action\_page.php" method="get">**

**or:**

**<form action="action\_page.php" method="post">**

### When to Use GET?

You can use GET (the default method):

If the form submission is passive (like a search engine query), and without sensitive information.

When you use GET, the form data will be visible in the page address:

**action\_page.php?firstname=Mickey&lastname=Mouse**

### Note

GET is best suited to short amounts of data. Size limitations are set in your browser.

### When to Use POST?

You should use POST:

If the form is updating data, or includes sensitive information (password).  
POST offers better security because the submitted data is not visible in the page address.

## The Name Attribute

To be submitted correctly, each input field must have a name attribute.  
This example will only submit the "Last name" input field:

### Example

```
<form action="action_page.php">  
    First name:<br>  
    <input type="text" value="Mickey"><br>  
    Last name:<br>  
    <input type="text" name="lastname" value="Mouse"><br><br>  
    <input type="submit" value="Submit">  
</form>
```

## Grouping Form Data with <fieldset>

The <fieldset> element groups related data in a form.  
The <legend> element defines a caption for the <fieldset> element.

### Example

```
<form action="action_page.php">  
    <fieldset>  
        <legend>Personal information:</legend>  
        First name:<br>  
        <input type="text" name="firstname" value="Mickey"><br>  
        Last name:<br>  
        <input type="text" name="lastname" value="Mouse"><br><br>  
        <input type="submit" value="Submit">  
    </fieldset>  
</form>
```

## HTML Form Attributes

An HTML <form> element, with all possible attributes set, will look like this:

```
<form action="action_page.php" method="post" target="_blank" accept  
charset="UTF-8"  
enctype="application/x-www-form-urlencoded"                autocomplete="off"  
novalidate>  
.
```

## form elements

.

**</form>**

## Playing Videos in HTML

Before HTML5, there was no standard for showing videos on a web page.

Before HTML5, videos could only be played with a plug-in (like flash).

The HTML5 <video> element specifies a standard way to embed a video in a web page.

### The HTML <video> Element

To show a video in HTML, use the <video> element:

#### Example

```
<video width="320" height="240" controls>  
    <source src="movie.mp4" type="video/mp4">  
    <source src="movie.ogg" type="video/ogg">  
    Your browser does not support the video tag.  
</video>
```

#### How it Works

The controls attribute adds video controls, like play, pause, and volume.

It is a good idea to always include width and height attributes.

If height and width are not set, the browser does not know the size of the video. The effect will be that the page will change (or flicker) while the video loads.

Text between the <video> and </video> tags will only display in browsers that do not support the <video> element.

Multiple <source> elements can link to different video files. The browser will use the first recognized format.

### HTML <video> Autoplay

To start a video automatically use the autoplay attribute:

#### Example

```
<video width="320" height="240" autoplay>  
    <source src="movie.mp4" type="video/mp4">  
    <source src="movie.ogg" type="video/ogg">  
    Your browser does not support the video tag.  
</video>
```



## Note!

The autoplay attribute does not work in mobile devices like iPad and iPhone.

## HTML Video – Methods, Properties, and Events

HTML5 defines DOM methods, properties, and events for the <video> element.

This allows you to load, play, and pause videos, as well as setting duration and volume.

There are also DOM events that can notify you when a video begins to play, is paused, etc.

## HTML5 Video Tags

Tag	Description
<video>	Defines a video or movie
<source>	Defines multiple media resources for media elements, such as <video> and <audio>
<track>	Defines text tracks in media players

## Form Validation and Handler

### JavaScript Form Validation

HTML form validation can be done by a JavaScript.

If a form field (fname) is empty, this function alerts a message, and returns false, to prevent the form from being submitted:

### JavaScript Example

```
function validateForm() {  
    var x = document.forms["myForm"]["fname"].value;  
    if (x == null || x == "") {  
        alert("Name must be filled out");  
        return false;  
    }  
}
```

The function can be called when the form is submitted:

### HTML Form Example

```
<form name="myForm" action="demo_form.asp" onsubmit="return validateForm()" method="post">  
    Name: <input type="text" name="fname">  
    <input type="submit" value="Submit">
```

</form>

## HTML Form Validation

HTML form validation can be performed automatically by the browser:

If a form field (fname) is empty, the required attribute prevents this form from being submitted:

### HTML Form Example

```
<form action="demo_form.asp" method="post">
<input type="text" name="fname" required>
<input type="submit" value="Submit">
</form>
```

### Note!

HTML form validation does not work in Internet Explorer 9 or earlier.

## Data Validation

Data validation is the process of ensuring that computer input is clean, correct, and useful.

Typical validation tasks are:

- has the user filled in all required fields?
- has the user entered a valid date?
- has the user entered text in a numeric field?

Most often, the purpose of data validation is to ensure correct input to a computer application.

Validation can be defined by many different methods, and deployed in many different ways.

Server side validation is performed by a web server, after input has been sent to the server.

Client side validation is performed by a web browser, before input is sent to a web server.

## PHP 5 Form Handling

The PHP superglobals **\$\_GET** and **\$\_POST** are used to collect form-data.

### PHP - A Simple HTML Form

The example below displays a simple HTML form with two input fields and a submit button:

### Example

```
<html>
  <body>
    <form action="welcome.php" method="post">
```

```
Name: <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit">
</form>
</body>
</html>
```

When the user fills out the form above and clicks the submit button, the form data is sent for processing to a PHP file named "welcome.php". The form data is sent with the HTTP POST method.

To display the submitted data you could simply echo all the variables. The "welcome.php" looks like this:

```
<html>
<body>
    Welcome <?php echo $_POST["name"]; ?><br>
    Your email address is: <?php echo $_POST["email"]; ?>
</body>
</html>
```

The output could be something like this:

**Welcome John**

**Your email address is john.doe@example.com**

The same result could also be achieved using the HTTP GET method:

### Example

```
<html>
<body>
    <form action="welcome_get.php" method="get">
        Name: <input type="text" name="name"><br>
        E-mail: <input type="text" name="email"><br>
        <input type="submit">
    </form>
</body>
</html>
```

and "welcome\_get.php" looks like this:

```
<html>
<body>
    Welcome <?php echo $_GET["name"]; ?><br>
    Your email address is: <?php echo $_GET["email"]; ?>
</body>
</html>
```

## Quote of the Day!

A fish in an aquarium thinks it has it all until it gets to the ocean.