Home > My courses > 121 - WEB101 / CCS3218 > 07 HTML Block, Inline Element and Forms Control > Lesson Proper for Week 7

Lesson Proper for Week 7

Block 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).

This are the common example of Block Element:

ª <div> - Document division.

ª <article> - Article content.

ª <p> - Paragraph.

ª <table> - Table.

ª <li> - List.

Inline Elements

An inline element does not start on a new line and it only takes up as much width as necessary.

This are the common example of Inline Element:

ª <b> - Makes the enclosed text into bold text.

ª <i> - Defines italic text

ª <u> - Displays the surrounded text underlined.

ª <strong> - Creates emphasis for the selected text, rendered in bold.

ª <sup> - above, this renders text in superscript Like this

<sup> - above, this renders text in superscript.

- <sub> - Renders the text in subscript, which is words under the normal text. Like this

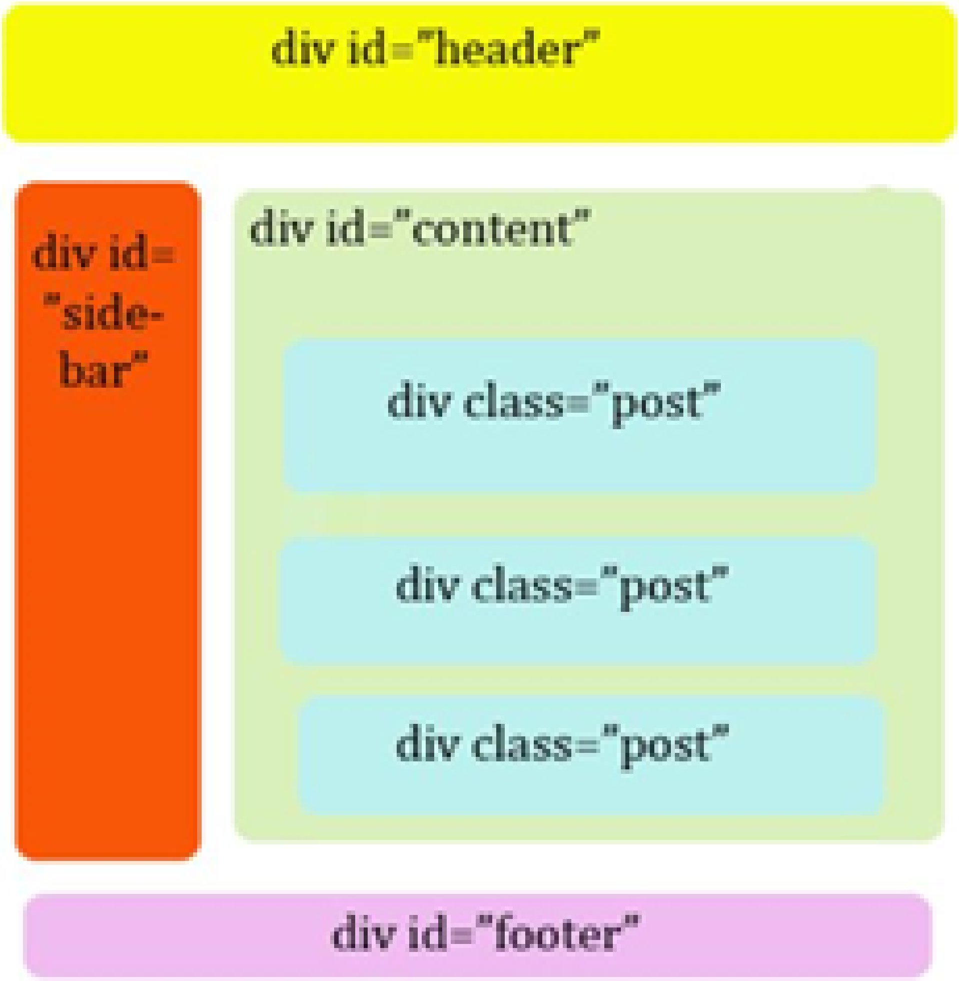
Grouping HTML Elements

There are two important tags which we use very frequently to group various other HTML tags

- <div> - tag is used to group the large section of HTML elements together.
- <span> - tag is used as a generic container of inline elements. It is used for styling purpose to the grouped inline elements (using class and id attribute or inline style).

Div Element

This is the very important block level tag which plays a big role in grouping various other HTML tags and applying CSS on group of elements. Even now <div> tag can be used to create webpage layout where we define different parts (Left, Right, Top etc.) of the page using <div> tag. This tag does not provide any visual change on the block but this has more meaning when it is used with CSS.



The **<div>** tag defines a division or a section in an HTML document. The **<div>** tag is used as a container for HTML

The **<div>** tag defines a division or a section in an HTML document. The **<div>** tag is used as a container for HTML elements. With the div tag, you can group large sections of HTML elements together and format them with CSS.

Syntax of Div Element:

<div> content of div here......</div>

NOTE:

*	**The HTML <div> element is found within the <body> tag.**

*	**The <div> tag is generally used to group elements together.**

Span Element

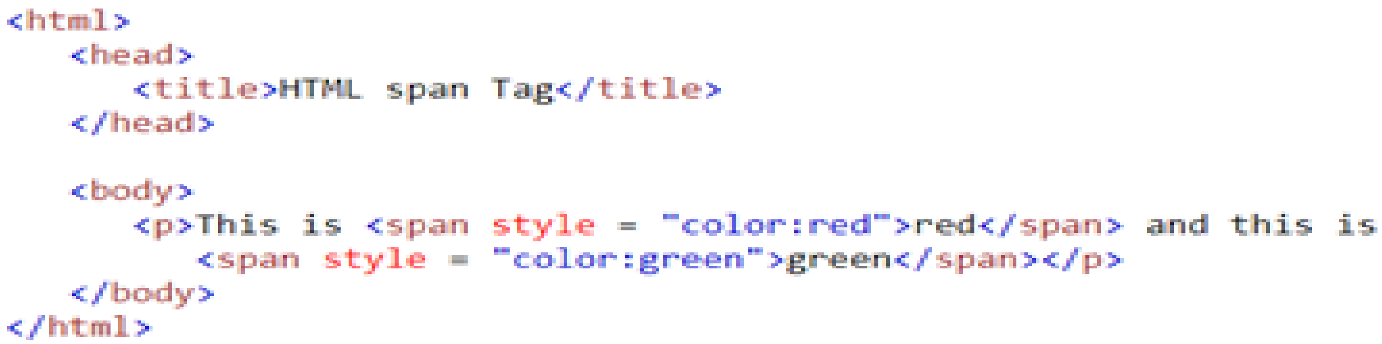
The HTML <span> is an inline element and it can be used to group inline-elements in an HTML document. This tag also does not provide any visual change on the block but has more meaning when it is used with CSS.

Syntax of Span Element:

<span> content of div here......</span>

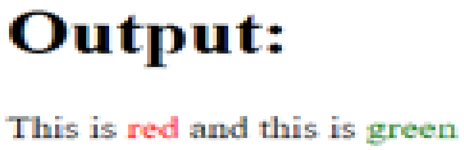
The difference between the <span> tag and the <div> tag is that the <span> tag is used with inline elements whereas the <div> tag is used with block-level elements.

Example of Span Element:



```
<html>
   <head>
      <title>HTML span Tag</title>
   </head>

   <body>
      <p>This is <span style = "color:red">red</span> and this is
         <span style = "color:green">green</span></p>
   </body>
</html>
```



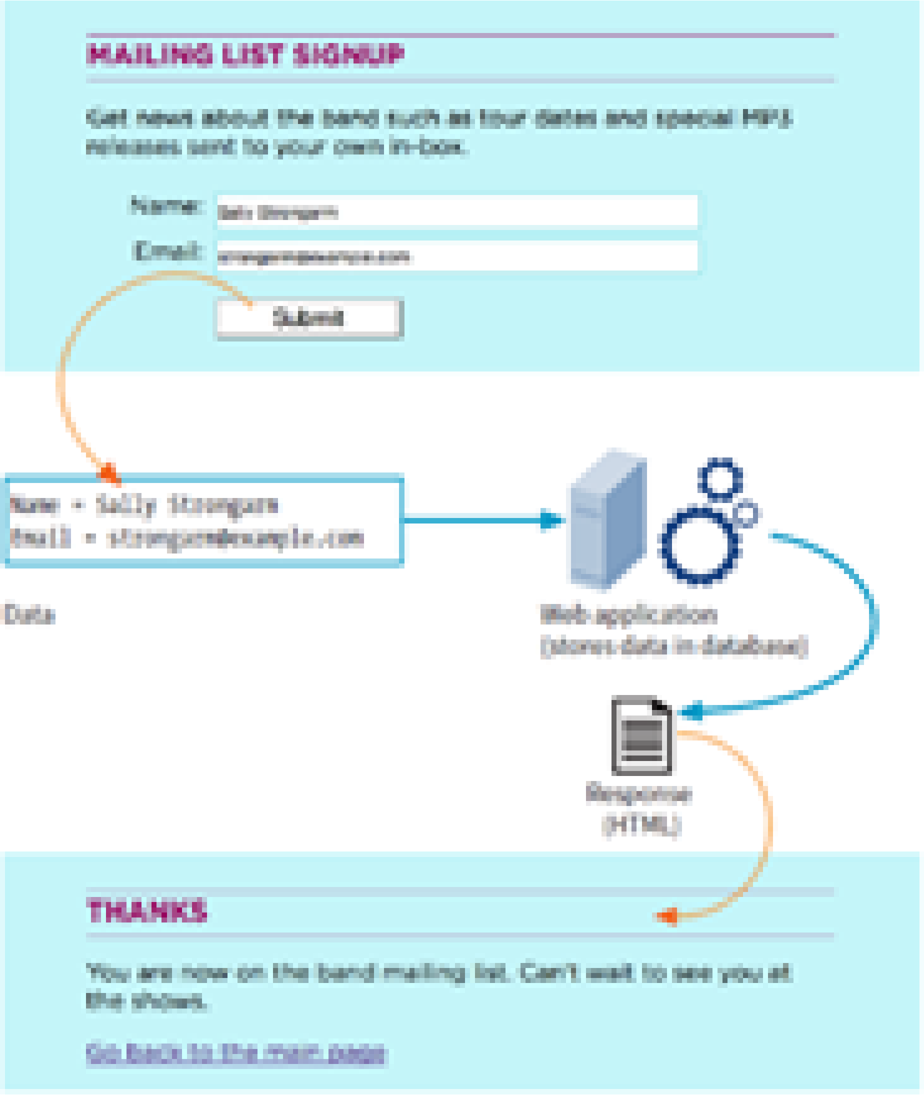
Output:

This is red and this is green

The Form Element

Forms are added to web pages with the **form** element. The form element is a container for all the content of the **form**, including some number of form controls, such as text-entry fields and buttons. It may also contain block elements **(h1, p,** and **lists,** for example). However, it may not contain another **form** element.

Web page forms have three important parts: a **<form>** tag, form **input** elements, and a **submit** button.



MAILING LIST SIGNUP

Get news about the band such as tour dates and special MP3 releases sent to your own in-box.

Name: Sally Strongarm

Email: strongarm@example.com

Submit

```
Name = Sally Strongarm
Email = strongarm@example.com
```

THANKS

You are now on the band mailing list. Can't wait to see you at the shows.

Go back to the main page

FIGURE 9-1. What happens behind the scenes when a web form is submitted.

Syntax:

```
<form>
    //input controls e.g. textfield, textarea, radiobutton, button
</form>
```

Example:

```
<!DOCTYPE html>
```

```
<!DOCTYPE html>

<html>

<head>

<title>Mailing List Signup</title>

<meta charset="utf-8">

</head>

<body>

<h1>Mailing List Signup</h1>

<form action="/mailinglist.php" method="POST">

<fieldset>

<legend>Join our email list</legend>

<p>Get news about the band such as tour dates and special MP3 releases sent to your own in-box.</p>

<ol>

<li><label for="firstlast">Name:</label>

<input type="text" name="fullname" id="firstlast"></li>

<li><label for="email">Email:</label>

<input type="text" name="email" id="email"></li>

</ol>

<input type="submit" value="Submit">

</fieldset>

</form>

</body>

</html>
```

The action Attribute

The **action** attribute provides the location (URL) of the application or script that will be used to process the form. The **action** attribute in this example sends the data to a script called *mailinglist.php*:

```
<form action="/mailinglist.php" method="POST">...</form>
```

The *.php* suffix indicates that this form is processed by a script written in the PHP scripting language, but web forms

The *.php* suffix indicates that this form is processed by a script written in the PHP scripting language, but web forms may be processed by any of the following technologies:

* PHP (*.php*) is an open source scripting language most commonly used
* with the Apache web server. It is the most popular and widely supported
* forms processing option.
* Microsoft ASP (Active Server Pages; *.asp*) is a programming environment
* for the Microsoft Internet Information Server (IIS).
* Microsoft’s ASP.NET (Active Server Page; *.aspx*) is a newer Microsoft language
* that was designed to compete with PHP.
* Ruby on Rails. Ruby is the programming language that is used with the
* Rails platform. Many popular web applications are built with it.
* JavaServer Pages (*.jsp*) is a Java-based technology similar to ASP.
* Python is a popular scripting language for web and server applications.

The method Attribute

The **method** attribute specifies how the information should be sent to the server. Let’s use this data gathered from the sample form in FIGURE 9-1 as an example.

fullname = Sally Strongarm

email = strongarm@example.com

When the browser encodes that information for its trip to the server, it looks like this (see the earlier sidebar if you need a refresher on encoding):

fullname=Sally+Strongarm&email=strongarm%40example.com

There are only two methods for sending this encoded data to the server:

POST or **GET,** indicated by the **method** attribute in the **form** element. The method is optional and will default to **GET** if omitted. We’ll look at the difference between the two methods in the following sections.

An example uses the POST method, as shown here:

<form action="/mailinglist.php" method="POST">...</form>

The GET method

With the **GET** method, the encoded form data gets tacked right onto the **URL** sent to the server. A question mark character separates the URL from the following data, as shown here:

get http://www.bandname.com/mailinglist.php?name=Sally+Strongarm&email= strongarm%40example.com get http://www.bandname.com/mailinglist.php?name=Sally+Strongarm&email=strongarm%40example.com

GET is inappropriate if the form submission performs an action, such as deleting something or adding data to a database, because if the user goes back, it gets submitted again.

The POST method

When the form's method is set to POST, the browser sends a separate server request containing some special headers followed by the data. In theory, only the server sees the content of this request, and thus it is the best method for sending secure information such as a home address or other personal information. In practice, make sure HTTPS is enabled on your server so the user's data is encrypted and inaccessible in transit.

The POST method is also preferable for sending a lot of data, such as a lengthy text entry, because there is no character limit as there is for GET.

The GET method is appropriate if you want users to be able to bookmark the results of a form submission (such as a list of search results). Because the content of the form is in plain sight, GET is not appropriate for forms with private personal or financial information. In addition, GET may not be used when the form is used to upload a file.

Input Elements

Forms are made up of a variety of input elements. Some elements, such as text boxes, give users a way to add information in a free-form manner. Others, such as radio buttons, constrain what the user can submit. You can mix different types of input elements in a single form.

It is added to the form using the input element with its type attribute set to text, as shown here:

```
<li>

<label for="form-city">City:</label> <br />
<input type="text" name="city" value="Your Hometown" size="25" maxlength="50" id="form-city"/>

</li>
```

*   **name** attribute is required for identifying the variable name.

*   **id** attribute binds this control to its associated label (although it could also be referenced by style sheets and scripts).

*   **value** attribute specifies default text that appears in the field when the form is loaded. When you reset a form, it returns to this value.

*   **size** By default, browsers display a text-entry box that is 20 characters wide, but you can change the number of characters using the size attribute.

*   **maxlength** By default, users can type an unlimited number of characters in a text field regardless of its size

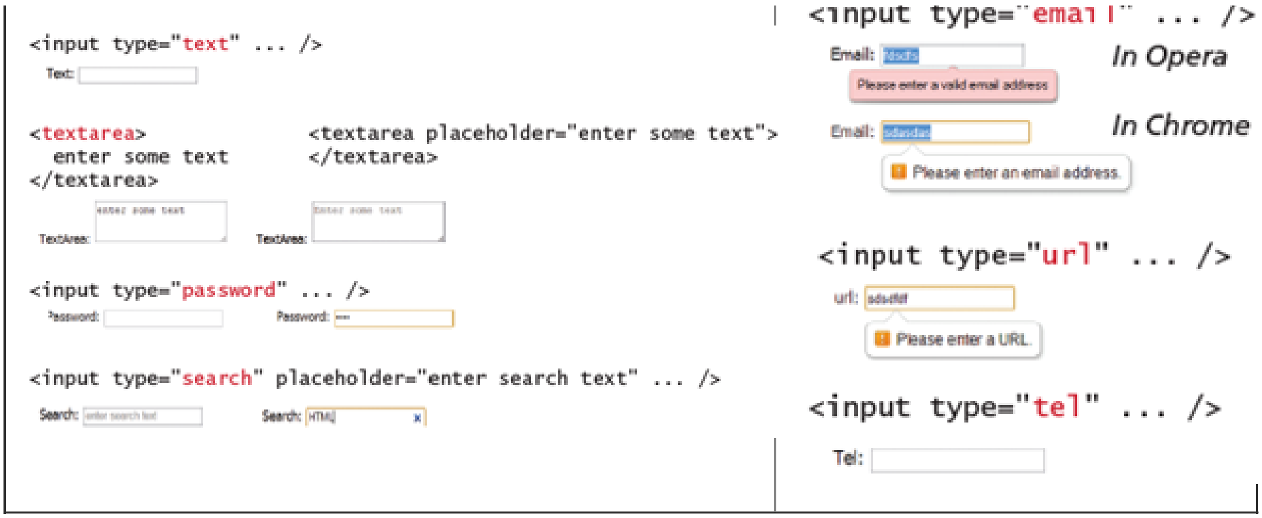
maxlength By default, users can type an unlimited number of characters in a text field regardless of its size (the display scrolls to the right if the text exceeds the character width of the box). You can set a maximum character limit using the maxlength attribute if the forms processing program you are using requires it.

Text Boxes

Text boxes are input fields designed specifically for users to type data into, such as typing a name or comment. Which element you use to collect text input depends on whether users are asked to enter a single line of text (input) or multiple lines (textarea).

TEXT INPUT CONTROLS	
TYPE	**DESCRIPTION**
text	Creates a single-line text entry box. **<input type="text" name="title" />**
textarea	Creates a multiline text entry box. (hint text that disappears once user begins typing into the field). **<textarea rows="3" ... />**
password	Creates a single-line text entry box for a password (which masks the user entry as bullets or some other character) **<input type="password" ... />**
search	Creates a single-line text entry box suitable for a search string. **<input type="search" ... />**
email	Creates a single-line text entry box suitable for entering an email address. **<input type="email" ... />**
tel	Creates a single-line text entry box suitable for entering a telephone. **<input type="tel" ... />**
url	Creates a single-line text entry box suitable for entering a URL. **<input type="url" ... />**

Example:



```
<input type="text" ... />

<textarea>
  enter some text
</textarea>

<textarea placeholder="enter some text">
</textarea>

<input type="password" ... />

<input type="search" placeholder="enter search text" ... />

<input type="email" ... />

<input type="url" ... />

<input type="tel" ... />
```

Please enter a valid email address

Please enter an email address.

Please enter a URL.

Multiline text entry field

<textarea>. . . . .</textarea>

At times, you’ll want your users to be able enter more than just one line of text. For these instances, use the textarea element that is replaced by a multi-line, scrollable text entry box when displayed by the browser

<li>

<label for="form-entry">Official contest entry:</label> <br/>

<textarea name="contest_entry" rows="5" cols="100" id="form-entry"/>Tell us why you love the band in 50 words or less. Five winners will get backstage passes!</textarea>

</li>

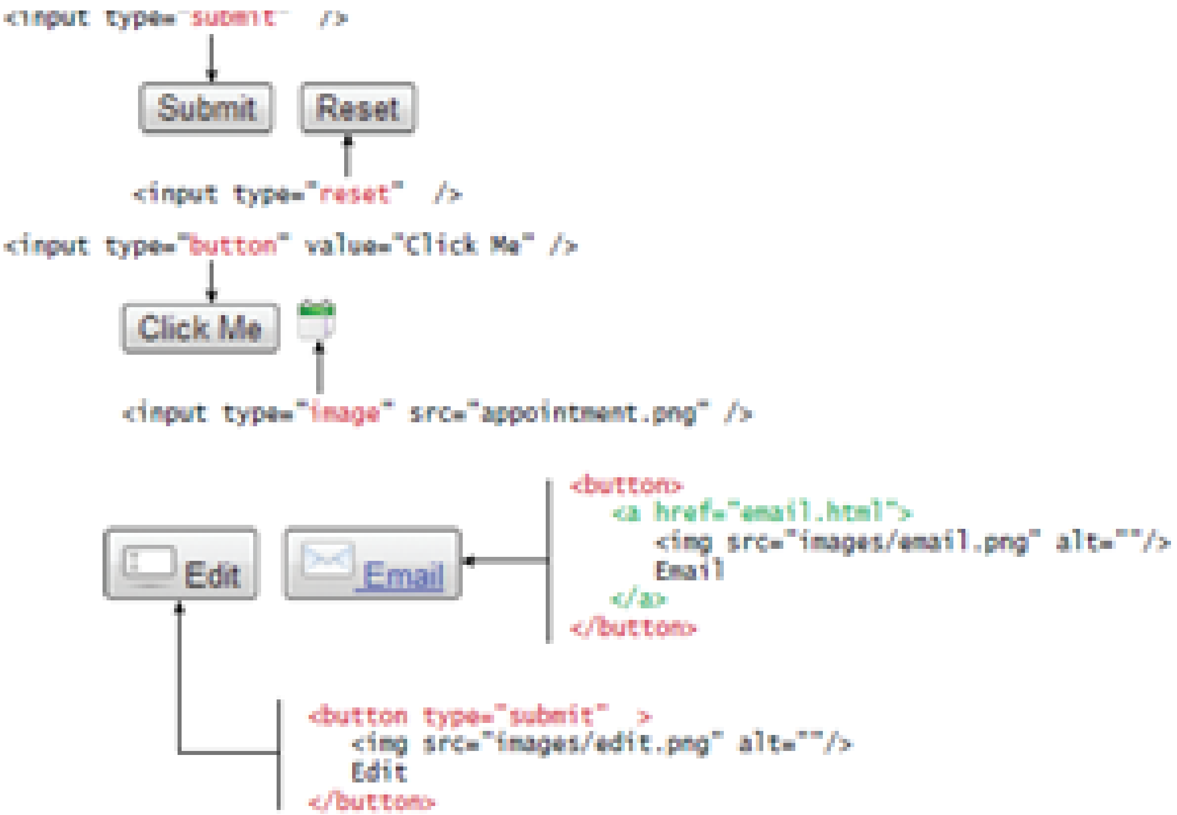
Unlike the empty input element, the textarea element has content between its opening and closing tags. The content of the textarea element is the intial content of the text box when the form is displayed in the browser.

In addition to the required name attribute, the textarea element uses the following attributes:

- **rows** Specifies the number of lines of text the area should display. Scrollbars will be provided if the user types more text than fits in the allotted space.
- **cols** Specifies the width of the text area measured in number of characters.

Submit and Reset Buttons

There are several kinds of buttons that can be added to web forms. The most fundamental is the submit button. When clicked or tapped, the submit button immediately sends the collected form data to the server for processing.



```
<input type="submit" />
<input type="reset" />
<input type="button" value="Click Me" />
<input type="image" src="appointment.png" />
<button>
   <a href="email.html">
      <img src="images/email.png" alt=""/>
      Email
   </a>
</button>
<button type="submit" >
   <img src="images/edit.png" alt=""/>
   Edit
</button>
```

FIGURE 4.23 Example button elements

A reset button returns the form controls to the state they were in when the form initially loaded. In other words, resetting the form doesn't simply clear all the fields.

BUTTON ELEMENTS	
TYPE	**DESCRIPTION**
<input type="submit">	Creates a button that submits the form data to the server.
<input type="reset">	Creates a button that clears any of the user's already entered form data.
<input type="button">	Creates a custom button. This button may require JavaScript for it to actually perform any action.
<input type="image">	Creates a custom submit button that uses an image for its display.
<button>	Creates a custom button. The <button> element differs from <input type="button"> in that you can completely customize what appears in the button; using it, you can, for instance, include both images and text, or skip server-side processing entirely by using hyperlinks. You can turn the button into a submit button by using the type="submit" attribute.

◀ Preliminary Activity for Week 7

Jump to...

Analysis, Application, and Exploration for Week 7 ▶

Navigation

- Home
- Dashboard
  - Site pages
  - My courses
    - 121 - CC106
    - 121 - BPM101 / DM103
    - 121 - OAELEC2
    - 121 - ITE3
    - 121 - MUL101
    - 121 - ITSP2B
    - 121 - WEB101 / CCS3218
      - Participants
      - Grades
      - General
      - 01 Introduction to Internet and World Wide Web
      - 02 Full Stack Development Overview
      - 03 The Basics of HTML
      - 04 Mark Up Text
      - 05 HTML Link, Image and Table
      - 06 - Preliminary Examination
      - 07 HTML Block, Inline Element and Forms Control
        - Preliminary Activity for Week 7
        - **Lesson Proper for Week 7**
        - Analysis, Application, and Exploration for Week 7
        - Generalization for Week 7
        - Evaluation for Week 7
        - Assignment for Week 7
      - 08 Introduction To Cascading Style Sheets
      - 09 The Font Style Properties
      - 10 The List and Table style properties
      - 11 Advanced CSS
      - 12 - Midterm Examination
      - 13 JavaScript Basics
      - 14 JavaScript Objects

- 15 Introduction to PHP
- 16 PHP Control Statement and Function
- 17 PHP Array and String

Courses

Fair Warning

NOTICE: Please be reminded that it has come to the attention of the Publishing Team of eLearning Commons that learning materials published and intended for ***free use only by students and faculty members within the eLearning Commons network were UNLAWFULLY uploaded in other sites without due and proper permission***.

PROSECUTION: Under Philippine law (Republic Act No. 8293), copyright infringement is punishable by the following: Imprisonment of between 1 to 3 years and a fine of between 50,000 to 150,000 pesos for the first offense. Imprisonment of 3 years and 1 day to six years plus a fine of between 150,000 to 500,000 pesos for the second offense.

COURSE OF ACTION: Whoever has maliciously uploaded these concerned materials are hereby given an ultimatum to take it down within 24-hours. Beyond the 24-hour grace period, our Legal Department shall initiate the proceedings in coordination with the National Bureau of Investigation for IP Address tracking, account owner identification, and filing of cases for prosecution.

2nd Semester Enrollment

visit www.bcp.edu.ph
Enrollment registration is now Ongoing
For 2nd Semester SY 2021 - 2022
We are accepting new students, returnees and transferees.
"Be trained to be the best,
Be linked to success"
bcp-inquire@bcp.edu.ph
(8)442-8601 | (8)518-8050

Activities

- Assignments
- Forums
- Quizzes
- Resources

Bestlink College of the Philippines
College Department

Powered byeLearning Commons