**Day 2:**

**Introduction to Web Development**

**The World Wide Web**

The Web:

* + Collection of HTML documents
    - HTML: HyperText Markup Language
    - Basis for almost every Web page
  + HTML documents are linked together using hyperlinks
  + Invented in 1990
  + Has over 30 billion Web pages.

**TIM Berners-Lee**

* First internet site and first navigator were invented in 1990 by TIM Berners-Lee
* He is the initiator of the 3 main web technologies: Web addresses, HTTP and HTML
* Founder and president of W3SC since 1994

**W3C: WWW (World Wide Web) Consortium:** It is independent organization that issues standards for the web.

**WWW: Main Components:**

* + HTTP: The main protocol of the Web
    - State less protocol
  + Servers: Computers that host the files that make up the Web
  + Internet: The world’s largest computer network
  + Browser: A program that runts on Client computer to display files found on the web

**HTML: (Hypertext Markup Language )**

* + **Text Content**: What you see
  + **Markup**: What it looks like
  + **References to ther documents**: images or videos
  + **Hypertext**: Links to other pages

**HTML 5**

* + Fundamentally , the fifth version of HTML language
  + Referes also to the combination of three technologies:
  + HTML5, CSS3 and Javascript
  + HTML: Structure of the Web pages
  + CSS: Style and presentation of the Web pages
  + Javascript: Behaviour of Web elements.

**HTTP:**

* + Hyper TExt Transfer Protocol:
  + Application protocol for distributed , collaborative, hypermedia information System.
  + Foundation of data communication for the world wide web

**A Request-Repsonse protocol in the client-server computing model**

* + Request submitted by the client to ask the server for a resource or a computation.
  + Response sent by the server
  + Can contain requested content in the body.

## HTTP Request Method

**GET**

The GET method is used to retrieve information from the given server using a given URI. Requests using GET should only retrieve data and should have no other effect on the data.

**HEAD**

Same as GET, but it transfers the status line and the header section only.

**POST**

A POST request is used to send data to the server, for example, customer information, file upload, etc. using HTML forms.

**PUT**

Replaces all the current representations of the target resource with the uploaded content.

**DELETE**

Removes all the current representations of the target resource given by URI.

**CONNECT**

Establishes a tunnel to the server identified by a given URI.

**OPTIONS**

Describe the communication options for the target resource.

**TRACE**

Performs a message loop back test along with the path to the target resource.

**HTTP Request Header**

**GET /hello.htm HTTP/1.1**

**User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT)**

**Host: www.transflower.in**

**Accept-Language: en-us**

**Accept-Encoding HTTP REQUEST :Get**

**: gzip, deflate**

**Connection: Keep-Alive**

**HTTP Request with Posting Data to server**

**POST /cgi-bin/process.cgi HTTP/1.1**

**User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT)**

**Host: www.transflower.in**

**Content-Type: application/x-www-form-urlencoded**

**Content-Length: length**

**Accept-Language: en-us**

**Accept-Encoding: gzip, deflate**

**Connection: Keep-Alive**

**licenseID=string&content=string&/paramsXML=string**

**HTTP Status Code**

**The first line of HTTP Reponse is a status code.**

|  |  |  |
| --- | --- | --- |
| Code | Description | Signification |
| 200 | OK | It Works |
| 201 | Created | The Resouce was created |
| 304 | Not Modified | The client can use the cached version of the resource, as it was not mofified |
| 400 | Bad Request | The request of the client is badly written |
| 401 | Not Authorized | The Client must authenticate |
| 403 | Forbiden | The client is not authorized to use the resource |
| 404 | Not Found | The resource does not exist |
| 500 | Internal Server Error | There is a problem in the implementation of the service |

## HTTP Web Request Methods

## HTTP defined methods (or verbs) to indicate the desired action to be performed on the identified resource.

## GET:

## Returns a representation of the resource (JSON for example) and a reponse code 200 (OK) if all is well.

## In the case of an erros, returns a 404 (not found) or a 400 (bad request).

## GET must never update a resource.

## Example:

**GET** [**https://www.example.com/customers/5434**](https://www.example.com/customers/5434)

## DELETE:

## In the successful deletion , returns 200 (OK), as well as a content, such as the deleted element.

## IN ase of an unexisting resoucr, returns 204 )no content).

## Example:

**DELETE http://www.example.com/customers/12345**

**PUT:**

## Create /update a resource by indicating representation of the new resource, and the URI is that of the resource itself.

## If the resource already exists, it will be mofified, if not , it will be created.

## Returns 201 (created) in case of the creation of a new resource, and 200 (OK) or 204 9no content ) in case of the creation of a new resource, and 200 (OK) or 204 (no content) in case of an update

## Idempotent ( executing several times the same request always gives the same result.

## Does not usually return content

## Equivalent in programming language to a : obj. attribute = calue

## Example:

**PUT /articles/1234**

**<article>**

**<title>red stapler</title>**

**<price currency="eur">12.50</price>**

**</article>**

## POST : create/update a resource using a handler

## Mainly enables:

## Annotating existing resources.

## Posting message in a forum

## Sending data to a process that will use it

## Adding data to database..

## The processing is deternied by the server

## User if we don’t know the exact URI of the resource

## Returs 201 if the entity was , 200 (OK) it the response contains an entity, and 204 if not. Must also returen a Location (URI of the resource)

## Non idempotent

## Equivalent in programming lauguage to a : obje.sert\_attibte(Vale)

## Example:

**POST /articles**

**<article>**

**<title>red stapler</title>**

**<price currency="eur">12.50</price>**

**</article>**

**201 Created**

**Location: /articles/63636**

# HTML (Hyper Text Markup Language)

* **HTML is used to describe web pages**
* **It is not a programming language**
* **Documents are save with extention .html or .htm.**
* **Uses markup tags to describe web pages.**
* **Not case sensitive**
* **Can have attribute which provide additional information about HTML elements on your page.**
* **<body bgcolor=”red”>**
* **<table border=”0”>**

|  |  |
| --- | --- |
| **POST /articles**  **<article>**  **<title>red stapler</title>**  **<price currency="eur">12.50</price>**  **</article>**  **201 Created**  **Location: /articles/63636** | HTML Head Section: contain information about the document. The browser does not display this information to user. Following tags can be in the the head section : <base>, <link>, <meta>, <script>, <style> and <title> |

**Document (Body) contents**

**Body Text**

* HTML truncates spaces in your text.
* Use <br> to insert new lines.
* Use <p> tag to create paragraphs

**Comments in the HTML document**

* <!-- This a sample HTML comment 🡪

**Physical Character Effects:**

**Bold Font** : <b>… </b>, Italic: <i>…</i>, Underline: <u> …</u>

**Srikethrough**: <strike> or <s>

**Fixed width font:** <tt> - normal text in fixed -width font

**Subscript:** <sub>, Superscript: <sup>

**List of HTML Elements**

**Tag Short description**

**The root element**

**html** Contains the whole document

**Document metadata**

**head** Defines the document's header block

**title** Specifies the document's title

**base** Sets base URI to solve relative URIs

**link** Gives relational information for documents

**meta** Provides information for the document

**style** Contains presentational attributes

**Sections**

**body** Contains all renderable elements

**article** Contains distributable content

**section** Defines a section of the document

**nav** Defines a navigation section

**aside** Holds content only slightly related

**h1** Inserts a level 1 heading

**h2** Inserts a level 2 heading

**h3** Inserts a level 3 heading

**h4** Inserts a level 4 heading

**h5** Inserts a level 5 heading

**h6** Inserts a level 6 heading

**hgroup** Groups consecutive headings

**header** Contains the header of a section

**footer** Contains the footer of a section

**address** Provides contact information

**Grouping content**

**p** Inserts a paragraph

**hr**  Draws a horizontal line or rule

**pre** Defines a block of preformatted text

**blockquote** Block level quotation

**ol** Inserts an ordered list

**ul** Inserts an unordered list

**menu** Inserts a toolbar menu

**li** Defines a list item

**dl** Inserts a definitions list

**dt** Inserts a term in a list

**dd** Provides descriptions in a list

**figure** Marks its content as a reference

**figcaption** Provides a caption for a figure

**main** Acts as a main container for elements

**div** Defines a block of content

**Text-level semantics**

**a** Inserts links or bookmarks

**em** Indicates emphasis

**strong**  Indicates strong emphasis

**small**  Renders text in "small" font

**s** Content no longer accurate or relevant

**cite** Inserts a citation or reference

**q** Inserts an inline quotation

**dfn** Provides a definition for a term

**abbr** Explains abbreviations

**ruby** Inserts ruby annotated text

**rt** Provides a ruby annotation

**rp** Makes text to be ignored in ruby

**data** Provides a machine-readable version

**time**  Represents a date and/or time

**code** Represents computer code

**var** Indicates an instance of a variable

**samp** Contains a program's sample output

**kbd** Represents text entered by users

**sub** Defines subscript text

s**up** Defines superscript text

**i** Renders italic text

**b** Text in bold style

**u** Represents non-textual annotations

**mark** Marks text in another document

**bdi** Isolates for bidirectional formatting

**bdo** Overrides the bidirectional algorithm

**span** Assings attributes to text (inline)

**br** Forces a line break

**wbr** Represents a line break opportunity

**Edits**

**ins** Indicates inserted text

**del** Indicates deleted text

**Embedded content**

**picture** Inserts a multi-source image

**source** Specifies alternative media resources

**img** Inserts an image

**iframe** Inserts a frame inside a document

**embed** Integrates external applications

**object** Runs external applications

**param** Sets a parameter for an object

**video** Inserts videos in the document

**audio** Inserts audio files in the document

**track** Provides text tracks for a video

**map** Defines a client-side image map

**area** Defines sectors for image maps

**Tabular data**

**table** Inserts a table

**caption** Provides a caption for a table

**colgroup** Groups columns in a table

**col** Sets attributes for a table's columns

**tbody** Defines the body of a table

**thead** Defines the header of a table

**tfoot** Defines the footer of a table

**tr** Inserts a row in a table

**td** Inserts a regular cell in a table

**th** Inserts a header cell in a table

**Forms**

**form** Inserts a form

**label** Sets a label for a control

**input** Displays an input control

**button**  Creates a button control

**select** Creates a select control

**datalist** Provides suggestions for input fields

**optgroup** Groups options in a select control

**option** Inserts an option in a select control

**textarea** Creates a multiline text input

**output** Shows the output of a process

**progress** Shows a task's completion progress

**meter** Represents a measurement

**fieldset** Groups controls in a form

**legend** Assigns a caption for a fieldset

**Interactive elements**

**details** Provides collapsable information

**summary** Provides a summary for a details element

**dialog** Inserts a dialog box

**Scripting**

**script** Contains scripts

**noscript** Provides alternative content for scripts

**template** Defines a template for data to come

**slot** Placeholder for data in components

**canvas** Renders dynamic bitmap graphics

# CSS : Cascading Style Sheets

CSS is a language that describes the style of an HTML document.

CSS describes how HTML elements should be displayed.

* **CSS** stands for **C**ascading **S**tyle **S**heets
* CSS describes **how HTML elements are to be displayed on screen, paper, or in other media**
* CSS **saves a lot of work**. It can control the layout of multiple web pages all at once
* External stylesheets are stored in **CSS files**

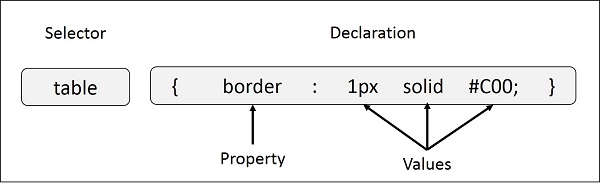
CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.

## Advantages of CSS

* **CSS saves time**
  + You can write CSS once and then reuse same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.
* **Pages load faster**
  + If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
* **Easy maintenance**
  + To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
* **Superior styles to HTML**
  + CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
* **Multiple Device Compatibility**
  + Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
* **Global web standards**
  + Now HTML attributes are being deprecated and it is being recommended to use CSS. So its a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.

A CSS comprises of style rules that are interpreted by the browser and then applied to the corresponding elements in your document. A style rule is made of three parts −

* **Selector**
  + A selector is an HTML tag at which a style will be applied. This could be any tag like <h1> or <table> etc.
* **Property**
  + A property is a type of attribute of HTML tag. Put simply, all the HTML attributes are converted into CSS properties. They could be *color*, *border* etc.
* **Value**
  + Values are assigned to properties. For example, *color*property can have value either *red* or *#F1F1F1* etc.



# CSS Basic Properties

List of basic CSS properties to work with while creating Rich Interactive HTML documents.

* [**Text Properties**](http://web.simmons.edu/~grabiner/comm244/weekthree/css-basic-properties.html#text)
* [**List Properties**](http://web.simmons.edu/~grabiner/comm244/weekthree/css-basic-properties.html#list)
* [**Border Properties**](http://web.simmons.edu/~grabiner/comm244/weekthree/css-basic-properties.html#border)
* [**Font Properties**](http://web.simmons.edu/~grabiner/comm244/weekthree/css-basic-properties.html#font)

**Text Properties**

|  |  |  |
| --- | --- | --- |
| **Property** | **Description** | **Values** |
| **color** | Sets the color of a text | RGB, hex, keyword |
| **line-height** | Sets the distance between lines | normal, number, length, % |
| **letter-spacing** | Increase or decrease the space between characters | normal, length |
| **text-align** | Aligns the text in an element | left, right, center, justify |
| **text-decoration** | Adds decoration to text | none, underline, overline, line-through |
| **text-indent** | Indents the first line of text in an element | length, % |
| **text-transform** | Controls the letters in an element | none, capitalize, uppercase, lowerca |

**List Properties**

|  |  |  |
| --- | --- | --- |
| **Property** | **Description** | **Values** |
| **list-style** | Sets all the properties for a list in one declaration | list-style-type, list-style-position, list-style-image,inherit |
| **list-style-image** | Specifies an image as the list-item marker | URL, none, inherit |
| **list-style-position** | Specifies where to place the list-item marker | inside, outside, inherit |
| **list-style-type** | Specifies the type of list-item marker | none, disc, circle, square, decimal, decimal-leading-zero,  armenian, georgian, lower-alpha, upper-alpha, lower-greek,  lower-latin, upper-latin, lower-roman, upper-roman, inherit |

**Border Properties**

|  |  |  |
| --- | --- | --- |
| **Property** | **Description** | **Values** |
| **border** | Sets all the border properties in one declaration | border-width, border-style, border-color |
| **border-bottom** | Sets all the bottom border properties in one declaration | border-bottom-width, border-bottom-style, border-bottom-color |
| **border-bottom-color** | Sets the color of the bottom border | border-color |
| **border-bottom-style** | Sets the style of the bottom border | border-style |
| **border-bottom-width** | Sets the width of the bottom border | border-width |
| **border-color** | Sets the color of the four borders | color\_name, hex\_number, rgb\_number, transparent, inherit |
| **border-left** | Sets all the left border properties in one declaration | border-left-width, border-left-style, border-left-color |
| **border-left-color** | Sets the color of the left border | border-color |
| **border-left-style** | Sets the style of the left border | border-style |
| **border-left-width** | Sets the width of the left border | border-width |
| **border-right** | Sets all the right border properties in one declaration | border-right-width, border-right-style, border-right-color |
| **border-right-color** | Sets the color of the right border | border-color |
| **border-right-style** | Sets the style of the right border | border-style |
| **border-right-width** | Sets the width of the right border | border-width |
| **border-style** | Sets the style of the four borders | none, hidden, dotted, dashed, solid, double, groove, ridge, inset, outset, inherit |
| **border-top** | Sets all the top border properties in one declaration | border-top-width, border-top-style, border-top-color |
| **border-top-color** | Sets the color of the top border | border-color |
| **border-top-style** | Sets the style of the top border | border-style |
| **border-top-width** | Sets the width of the top border | border-width |
| **border-width** | Sets the width of the four borders | thin, medium, thick, length, inherit |

**Font Properties**

|  |  |  |
| --- | --- | --- |
| **Property** | **Description** | **Values** |
| **font** | Sets all the font properties in one declaration | font-style, font-variant, font-weight, font-size/line-height, font-family, caption, icon, menu, message-box, small-caption, status-bar, inherit |
| **font-family** | Specifies the font family for text | family-name, generic-family, inherit |
| **font-size** | Specifies the font size of text | xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger, length, %, inherit |
| **font-style** | Specifies the font style for text | normal, italic, oblique, inherit |
| **font-variant** | Specifies whether or not a text should be displayed in a small-caps font | normal, small-caps, inherit |
| **font-weight** | Specifies the weight of a font | normal, bold, bolder, lighter,  100, 200, 300, 400, 500, 600, 700, 800, 900, inherit Careful, many of these are not supported! |

**CSS 3 Properties**

CSS3 is collaboration of CSS2 specifications and new specifications, we can call this collaboration is **module**.

Some of the modules are shown below −

* **Selectors**
* **Box Model**
* **Backgrounds**
* **Image Values and Replaced Content**
* **Text Effects**
* **2D Transformations**
* **3D Transformations**
* **Animations**
* **Multiple Column Layout**
* **User Interface**

**Animation Properties**

**Property Description**

**animation**  Specifies the keyframe-based animations.

**animation-delay** Specifies when the animation will start.

**animation-direction** Specifies whether the animation should play in reverse on alternate cycles or not.

**animation-duration** Specifies the number of seconds or milliseconds an animation should take to complete one cycle.

**animation-fill-mode** Specifies how a CSS animation should apply styles to its target before and after it is

executing.

**animation-iteration-count** Specifies the number of times an animation cycle should be played before stopping.

**animation-name** Specifies the name of @keyframes defined animations that should be applied to the selected element.

**animation-play-state** Specifies whether the animation is running or paused.

**animation-timing-function** Specifies how a CSS animation should progress over the duration of each cycle.

**Background Properties**

**Property Description**

**background** Defines a variety of background properties within one declaration.

**background-attachment** Specify whether the background image is fixed in the viewport or scrolls.

**background-clip** Specifies the painting area of the background.

**background-color** Defines an element's background color.

**background-image** Defines an element's background image.

**background-origin** Specifies the positioning area of the background images.

**background-position** Defines the origin of a background image.

**background-repeat** Specify whether/how the background image is tiled.

**background-size** Specifies the size of the background images.

**Border Properties**

**Property Description**

**border** Sets the width, style, and color for all four sides of an element's border.

**border-bottom**  Sets the width, style, and color of the bottom border of an element.

**border-bottom-color** Sets the color of the bottom border of an element.

**border-bottom-left-radius** Defines the shape of the bottom-left border corner of an element.

**border-bottom-right-radius** Defines the shape of the bottom-right border corner of an element.

**border-bottom-style** Sets the style of the bottom border of an element.

**border-bottom-width** Sets the width of the bottom border of an element.

**border-color**  Sets the color of the border on all the four sides of an element.

**border-image**  Specifies how an image is to be used in place of the border styles.

**border-image-outset**  Specifies the amount by which the border image area extends beyond the border box.

**border-image-repeat** Specifies whether the image-border should be repeated, rounded or stretched.

**border-image-slice**  Specifies the inward offsets of the image-border.

**border-image-source** Specifies the location of the image to be used as a border.

**border-image-width** Specifies the width of the image-border.

**border-left**  Sets the width, style, and color of the left border of an element.

**border-left-color**  Sets the color of the left border of an element.

**border-left-style**  Sets the style of the left border of an element.

**border-left-width** Sets the width of the left border of an element.

**border-radius**  Defines the shape of the border corners of an element.

**border-right** Sets the width, style, and color of the right border of an element.

**border-right-color** Sets the color of the right border of an element.

**border-right-style** Sets the style of the right border of an element.

**border-right-width** Sets the width of the right border of an element.

**border-style**  Sets the style of the border on all the four sides of an element.

**border-top**  Sets the width, style, and color of the top border of an element.

**border-top-color**  Sets the color of the top border of an element.

**border-top-left-radius** Defines the shape of the top-left border corner of an element.

**border-top-right-radius**  Defines the shape of the top-right border corner of an element.

**border-top-style**  Sets the style of the top border of an element.

**border-top-width** Sets the width of the top border of an element.

**border-width** Sets the width of the border on all the four sides of an element.

**Color Properties**

**Property Description**

**colo**r Specify the color of the text of an element.

**opacity** Specifies the transparency of an element.

**Dimension Properties**

**Property Description**

**height** Specify the height of an element.

**max-height** Specify the maximum height of an element.

**max-width** Specify the maximum width of an element.

**min-height** Specify the minimum height of an element.

**min-width** Specify the minimum width of an element.

**width** Specify the width of an element.

**Generated Content Properties**

**Property Description**

**content**  Inserts generated content.

**quotes**  Specifies quotation marks for embedded quotations.

**counter-reset**  Creates or resets one or more counters.

**counter-increment** Increments one or more counter values.

**Flexible Box Layout**

**Property Description**

**align-content** Specifies the alignment of flexible container's items within the flex container.

**align-items** Specifies the default alignment for items within the flex container.

**align-self** Specifies the alignment for selected items within the flex container.

**flex**  Specifies the components of a flexible length.

**flex-basis** Specifies the initial main size of the flex item.

**flex-direction** Specifies the direction of the flexible items.

**flex-flow** A shorthand property for the flex-direction and the flex-wrap properties.

**flex-grow** Specifies how the flex item will grow relative to the other items inside the flex container.

**flex-shrink** Specifies how the flex item will shrink relative to the other items inside the flex container.

**flex-wrap** Specifies whether the flexible items should wrap or not.

**justify-content** Specifies how flex items are aligned along the main axis of the flex container after any flexible lengths and auto margins have been resolved.

**order**  Specifies the order in which a flex items are displayed and laid out within a flex container.

**Font Properties**

**Property Description**

**font** Defines a variety of font properties within one declaration.

**font-family** Defines a list of fonts for element.

**font-size** Defines the font size for the text.

**font-size-adjust**  Preserves the readability of text when font fallback occurs.

**font-stretch** Selects a normal, condensed, or expanded face from a font.

**font-style** Defines the font style for the text.

**font-variant** Specify the font variant.

**font-weight** Specify the font weight of the text.

**List Properties**

**Property Description**

**list-style** Defines the display style for a list and list elements.

**list-style-image** Specifies the image to be used as a list-item marker.

**list-style-position** Specifies the position of the list-item marker.

**list-style-type** Specifies the marker style for a list-item.

**Margin Properties**

**Property Description**

**margin**  Sets the margin on all four sides of the element.

**margin-bottom** Sets the bottom margin of the element.

**margin-left** Sets the left margin of the element.

**margin-right** Sets the right margin of the element.

**margin-top** Sets the top margin of the element.

**Multi-column Layout Properties**

**Property Description**

**column-count**  Specifies the number of columns in a multi-column element.

**column-fill** Specifies how columns will be filled.

**column-gap** Specifies the gap between the columns in a multi-column element.

**column-rule** Specifies a straight line, or "rule", to be drawn between each column in a multi-column element.

**column-rule-color** Specifies the color of the rules drawn between columns in a multi-column layout.

**column-rule-style** Specifies the style of the rule drawn between the columns in a multi-column layout.

**column-rule-width** Specifies the width of the rule drawn between the columns in a multi-column layout.

**column-span** Specifies how many columns an element spans across in a multi-column layout.

**column-width** Specifies the optimal width of the columns in a multi-column element.

**columns**  A shorthand property for setting column-width and column-count properties.

**Outline Properties**

**Property Description**

**outline**  Sets the width, style, and color for all four sides of an element's outline.

**outline-color** Sets the color of the outline.

**outline-offset** Set the space between an outline and the border edge of an element.

**outline-style** Sets a style for an outline.

**outline-width** Sets the width of the outline.

**Padding Properties**

**Property Description**

**padding**  Sets the padding on all four sides of the element.

**padding-bottom** Sets the padding to the bottom side of an element.

**padding-left**  Sets the padding to the left side of an element.

**padding-right** Sets the padding to the right side of an element.

**padding-top** Sets the padding to the top side of an element.

**Print Properties**

**Property Description**

**page-break-after** Insert a page breaks after an element.

**page-break-before** Insert a page breaks before an element.

**page-break-inside** Insert a page breaks inside an element.

**Table Properties**

**Property Description**

**border-collapse** Specifies whether table cell borders are connected or separated.

**border-spacing** Sets the spacing between the borders of adjacent table cells.

**caption-side** Specify the position of table's caption.

**empty-cells** Show or hide borders and backgrounds of empty table cells.

**table-layout** Specifies a table layout algorithm.

**Text Properties**

**Property Description**

**direction** Define the text direction/writing direction.

**tab-size** Specifies the length of the tab character.

**text-align** Sets the horizontal alignment of inline content.

**text-align-last** Specifies how the last line of a block or a line right before a forced line break is aligned when text-

align is justify.

**text-decoration** Specifies the decoration added to text.

**text-decoration-color** Specifies the color of the text-decoration-line.

**text-decoration-line** Specifies what kind of line decorations are added to the element.

**text-decoration-style** Specifies the style of the lines specified by the text-decoration-line property

**text-indent** Indent the first line of text.

**text-justify** Specifies the justification method to use when the text-align property is set to justify.

**text-overflow** Specifies how the text content will be displayed, when it overflows the block containers.

**text-shadow** Applies one or more shadows to the text content of an element.

**text-transform** Transforms the case of the text.

**line-height** Sets the height between lines of text.

**vertical-align** Sets the vertical positioning of an element relative to the current text baseline.

**letter-spacing**  Sets the extra spacing between letters.

**word-spacing**  Sets the spacing between words.

**white-space** Specifies how white space inside the element is handled.

**word-break** Specifies how to break lines within words.

**word-wrap** Specifies whether to break words when the content overflows the boundaries of its container.

**Transform Properties**

**Property Description**

**backface-visibility** Specifies whether or not the "back" side of a transformed element is visible when facing the user.

**perspective**  Defines the perspective from which all child elements of the object are viewed.

**perspective-origin** Defines the origin (the vanishing point for the 3D space) for the perspective property.

**transform**  Applies a 2D or 3D transformation to an element.

**transform-origin** Defines the origin of transformation for an element.

**transform-style** Specifies how nested elements are rendered in 3D space.

**Transitions Properties**

**Property Description**

**transition** Defines the transition between two states of an element.

**transition-delay** Specifies when the transition effect will start.

**transition-duration** Specifies the number of seconds or milliseconds a transition effect should take to complete.

**transition-property** Specifies the names of the CSS properties to which a transition effect should be applied.

**transition-timing-function** Specifies the speed curve of the transition effect.

**Visual formatting Properties**

**Property** **Description**

**display** Specifies how an element is displayed onscreen.

**position**  Specifies how an element is positioned.

**top** Specify the location of the top edge of the positioned element.

**right** Specify the location of the right edge of the positioned element.

**bottom**  Specify the location of the bottom edge of the positioned element.

**left** Specify the location of the left edge of the positioned element.

**float** Specifies whether or not a box should float.

**clear** Specifies the placement of an element in relation to floating elements.

**z-index**  Specifies a layering or stacking order for positioned elements.

**overflow** Specifies the treatment of content that overflows the element's box.

**overflow-x** Specifies how to manage the content when it overflows the width of the element's content area.

**overflow-y** Specifies how to manage the content when it overflows the height of the element's content area.

**resize**  Specifies whether or not an element is resizable by the user.

**clip** Defines the clipping region.

**visibility**  Specifies whether or not an element is visible.

**cursor** Specify the type of cursor.

**box-shadow** Applies one or more drop-shadows to the element's box.

**box-sizing** Alter the default CSS box model.

**Advaned Web Programming**

**PHP (Personal Home Page)**

PHP is an HTML-embedded, server-side scripting language designed for web development. It is also used as a general purpose programming language. It was created by Rasmus Lerdorf in 1994 and appeared in the market in 1995. Much of its syntax is borrowed from C, C++, and Java.

PHP codes are simply mixed with HTML codes and can be used in combination with various web frameworks. Its scripts are executed on the server. PHP code is processed by a PHP interpreter. The main goal of PHP is to allow web developer to create dynamically generated pages quickly.

A PHP file consists of texts, HTML tags and scripts with a file extension of .php, .php3, or .phtml. You can create a login page, design a form, create forums, dynamic and static websites and many more with PHP.

PHP is a popular server-side language that is particularly good for web applications. Some of the largest companies and organizations from around the world utilize PHP for their operations. A large amount of web sites and applications are powered by PHP; therefore, an understanding of the PHP language is mandatory to fully understand and accept the power behind popular frameworks (such as Laravel, CodeIgniter or Symfony), and how popular websites and applications may be handling user's data.

**Key features of PHP**

* PHP stands for Hypertext Preprocessor.
* PHP is a server-side scripting language like ASP.
* PHP supports various databases like MySQL, Oracle, Sybase, Solid, PostgreSQL, Informix etc.
* PHP is an open source software and it is free to download and use.

**Advantages of PHP**

1) **Free of Cost:** PHP is open source and all its components are free to use and distribute.

2) **Platform independent:** PHP is platform independent and can be run on all major operating systems.

3) **Compatible with almost all servers:** PHP is compatible with almost all servers used today.

4) **Secure:** PHP has multiple layers of security to prevent threats and other malicious attacks.

5) **Easy to learn:** PHP has a very easy and understandable syntax. Its codes are based on C, C++ and embedded with HTML so it is very easy to learn for a programmer.

## The Language

PHP is an interpreted language. This means that you will write code statements (lines of code) and when a page is requested, the PHP ***interpreter*** will load your PHP code, parse it and then execute it. This differs from other languages, such as Java or C#, where the source code is compiled and then executed. This is useful for web development in the fact that you do not have to re-compile your source code for trivial code changes and the changes have immediate effect on all subsequent requests.

PHP is written as standard text files with the **.php** extension. PHP files are often saved within a folder in a web server's public directory (or a ***web root* directory**). On most systems this will either be named public or public\_html. For example, if a file was saved as index.php in a web root directory, a user could access it by typing .

**http://www.example.org or http://www.example.org/index.php.**

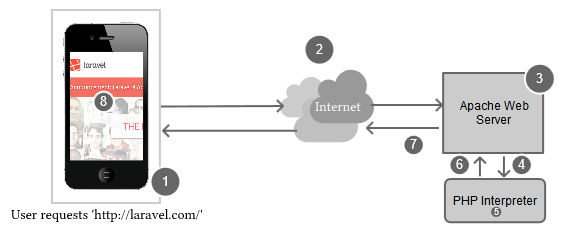
## The Request Life-cycle

What exactly is happening when a user types in the URL <http://example.org> ?

When a user types in **http://example.org** in a Web client (a browser, for instance), the client issues a GET request to the server. When Apache gets this request, it looks for a file named index.php (or index.html, remember the directory indexes from earlier?). If a file named index.php is found, Apache essentially says "Hey, this is a PHP file because it has the .php extension. I am going to give this to the PHP interpreter". After Apache decides that is is a PHP file, it gives it to the PHP interpreter. When PHP receives the file it reads through it and executes any PHP code it can find. After it is done with the file, the PHP interpreter gives the output of the code, if any, back to Apache. When Apache gets the output back from PHP, it sends that output back to a browser which renders it to the screen.

**The main goal of PHP is to generate some HTML document that a browser can render.**

However, modern applications built with client-side MVC frameworks often see the role of PHP change to just interacting with server-side data storage. Let's take another look at this process with a diagram. In this diagram, we will assume the user is going to the Laravel website at http://laravel.com/. The following figure has circled numbers that will highlight the various stages of the request. A step-by-step explanation of each step follows the figure.



**Steps**

1. The user enters **http://laravel.com** into their browser and taps/hits **enter**.
2. After the user has tapped/hit **enter**, the browser sends the page request over the Internet to the web server.
3. The web server gets the request and analyzes the request information. Apache realizes that we didn't specify a file, so it looks for a directory index and finds **index.php**.
4. Since Apache knows to send files that end with the **.php** file extension to the PHP interpreter, it asks PHP to execute the file.
5. In this step, PHP is executing the code contained in the **index.php** file from the request. During this step, PHP may interact with databases, the file system or make external API calls, amongst other things.
6. After PHP has finished executing the `index.php` file, it sends the output back to Apache.
7. Apache receives the output from PHP and sends it back over the Internet to a user's web browser. This is called the `web response`.
8. The user's web browser receives the response from the server, and renders the web page on a computer or device.

**<!DOCTYPE>**

**<html>**

**<body>**

**<?php**

**echo "<h2>Hello World PHP</h2>";**

**?>**

**</body>**

**</html>**

* **Performance**: Script written in PHP executes much faster then those scripts written in other languages such as JSP & ASP.
* **Open Source Software**: PHP source code is free available on the web, you can develop all the version of PHP according to your requirement without paying any cost.
* **Platform Independent**: PHP are available for WINDOWS, MAC, LINUX & UNIX operating system. A PHP application developed in one OS can be easily executed in another OS also.
* **Compatibility**: PHP is compatible with almost all local servers used today like Apache, IIS etc.
* **Embedded**: PHP code can be easily embedded within HTML tags and script.

**PHP installation**

To install PHP use **AMP (Apache, MySQL, PHP)** software stack. It is available for all operating systems. There are many AMP options available in the market that are given below:

* **WAMP** for Windows
* **LAMP** for Linux
* **MAMP** for Mac
* **SAMP** for Solaris
* **FAMP** for FreeBSD
* **XAMPP** (Cross, Apache, MySQL, PHP, Perl) for Cross Platform: It includes some other components too such as FileZilla, OpenSSL, Webalizer, Mercury Mail etc.

**Variable declaration in PHP**

**<?php**

**$title="Gerbera";**

**$price=15;**

**$quantity=2000;**

**echo "Flower  is: $title <br/>";**

**echo "Unit Price is: $price<br/>";**

**echo "Quantitis available are : $quantity <br/>";**

**?>**

**PHP function**

**<?php**

**function sayHello($name,$age){**

**echo "Hello  $name,  you are $age years old<br/>";**

**}**

**function sqaure($n){**

**return $n\*$n;**

**}**

**sayHello("Sameer",29);**

**sayHello("Vijay",27);**

**sayHello("Rani",24);**

**echo "Square of 4 is: ". sqaure (4);**

**?>**