**Introducing HTML**

0.1

* HTML (HyperText Markup Language) provides essential structure and semantics of the web pages for accessibility and seo(search engine optimization)- if we take away HTML nothing is left, no webpage, no content nothing. For proper website we need HTML
* html is not only for readable to the user but also conveyed the structure of the document, the relationship of the content with other and allow the user to link other pages and sites. It does everything bz it is a markup language.
* HTML and css work closely together. Xhtml, html and html 5 all are different versions of html. Html 5 is the latest one. HTML uses simple commands tags to define the various parts of a web page.
* CSS-Cascading Style Sheets. CSS is a styling language. We use it to make HTML. Controls the visual presentation of page elements, it is responsible for page layout, color, typography and element dimensions, how page looks- page designs-if we take away css, the page is simply uninstalled. CSS for short, give you creative control over the layout and design of your web pages. CSS isn’t anything without HTML. HTML provides web pages with content and meaningful structure.
* JS- page behaviour- if we take away JS, no function perform

0.2

* <p lang="en"></p>
* <p> is opening tag </p> closing tag, lang - attribute, "en"- value which is always in inverted commas.
* Child tag always close before parent tag. for ex. \*<p>Sandhya is a good girl<em>She does her work<strong>carefully</strong></em></p>

0.3 HTML 5

0.4

* <https://webplatform.github.io/docs/html/>
* https://developer.mozilla.org/en-US/docs/Web/HTML/Reference

0.5 Code Editor- brackets.io

**Basic Page structure**

0.1

< ! doctype html > it shows the browser and user which version of html is used...

< html >

The <html> tag appears once at the beginning of a web page and again (with an added forward slash) at the end: </html>. This tag tells a web browser that the information contained in this document is written in HTML, as opposed to some other language. All the contents of a page, including other tags, appear between the opening and closing <html> tags.

If you were to think of a web page as a tree, the <html> tag would be its root. Springing from the trunk are two branches that represent the two main parts of any web page: the head and the body.

< meta charset="utf-8">Every tag does not required a closing tag same as with meta tag. It needs only opening tag. "utf-8" is a standard charset.>

< meta name="describtion" content="A page for exploring basic HTML documents">

< title >we have two version of region</title>

< head > this makes the document better but not define. Head represent functioning content.

The head of a web page contains the title of the page. It may also include other, invisible information, like a page description, that browsers and search engines use. You surround the head section with opening and closing <head> tags

</head>

< body > visual elements are in the body.

Within the <body> tag, you commonly find tags like these:

1. **<p>** (opening paragraph tag), and where it ends with a **</p>** (closing paragraph tag).
2. The **<strong>** tag marks text as important. When you surround some text with it and its partner tag, **</strong>**, you get boldface type. The HTML snippet <strong> Warning!</strong> tells a web browser to strongly emphasize the word “Warning!”
3. The **<a>** tag, **or anchor** tag, creates a hyperlink in a web page. When clicked, a hyperlink—or link—can lead anywhere on the Web. You tell the browser where the link points by putting a web address inside the <a> tags. For instance, you can type <a href="http://www.missingmanuals.com">Click here!</a>.

</body>

< /html >

0.3

<html lang="en"> by putting lang attribute to html tag, it show that this HTML document is going to be use English as a first language. It is nice to write this but it is not mandatory or can say it is not required.

0.4

0.5 **Content Models**

**There are seven main models**

1. **Metadata content** - metadata is a content that sets up the presentation or behavior of the rest of the content, or that sets up the relationship of the document with other documents or that conveys other "out of band" information. ..base, link, meta, noscript, script, style, title.

2. **Embedded content**- is any content that import other resources into the document...audio, convas, iframe, embed, img, math, svg, video, object

3. **interactive content**- is any content that specifically intented for some type of user interaction....a, audio\*,button, embed, iframe, image\*, input\*, keygen, label, object\*, select, textarea, video\*....here \*under certain circumstances

4. **Heading content** - defines the heading of the section...h1, h2, h3, h4, h5, h6

5. **Phrasing content**- this is the text of the document...a\*, abbr, area\*, audio, b, bdi, bdo, br, button, convas, cite, code, data, date, datalist, del\*, dfn, em, embed, iframe, img, input, ins\*, kbd, keygen, label, map\*, mark, math, meter, noscript, object, output, progress, q, ruby, s, samp, script, select, small, span, strong, sub, sup, svg, textarea, time, u, var, video, text\*, wbr...here \*under certain circumstances

6. **Flow content**-this contains the maturity of elements in html 5. An element that would be included in the normal flow of the document... a, abbr, address, area\*, article, aside, audio, b, bdi, bdo, blockquote, br, button, convas, cite, code, data, date, datalist, del, dfn, div, dl, em, embed, fieldset, figure, footer, form, h1, h2, h3,h4,h5,h6, header, hr, i, iframe, img, input, ins, kbd, keygen, label, main ,map , mark, math, meter, noscript, object, ol, output, p, pre, progress, q, ruby, s, samp, script, section, select, small, span, strong, style\*, sub, sup, svg, textarea, time, u, ul, var, video, text\*, wbr...here \*under certain circumstances

7. **Sectioning content**- it defines the scope of header and footer….article, aside, nav, section

**Formatting Page Content**

**0.1**

HTML is a markup language that means, it use tags to identify content on the page…if there is no element is used, the browser shows the test by default setting.

* <pre> this element is used for giving space or can say for writing poems</p>
* <h1>it is the biggest one</h1> there are 6 types of heading h1 to h6 <h6> this is the smallest one</p>
* There should be one h1 on single page . There should be strategy behind using heading.
* <i>italic</i>
* <b>bold</b>
* <em>emphasize</em>
* <strong>strongly emphasize on text</strong>
* <br> break, it don’t need closing tag

**0.2**

Heading help define the structure of the page and control the hierarchy of the content. You can use heading values ranging from a top-level heading of h1 all the way down to an h6. Use heading in an intelligent manner, according to the importance of the content.

**0.3 Formatting paragraph**

**0.4 Controlling the breaks <br> line break**

**0.5 Formatting page content by bold, italic, strong and emphasize**

**0.6 Displaying special characters**

&lang; for left angle bracket < , &rang; for right angle bracket >, &amp; for &, &copy; for copyright, &trade; for trademark,

**0.7 Controlling white space**

&nbsp; for non-breaking space. For ex Formula&nbsp;one

* It is not possible to memorize all character entity so use Wikipedia

List of XML and HTML character entity….

**0.8 Inserting** **image**-

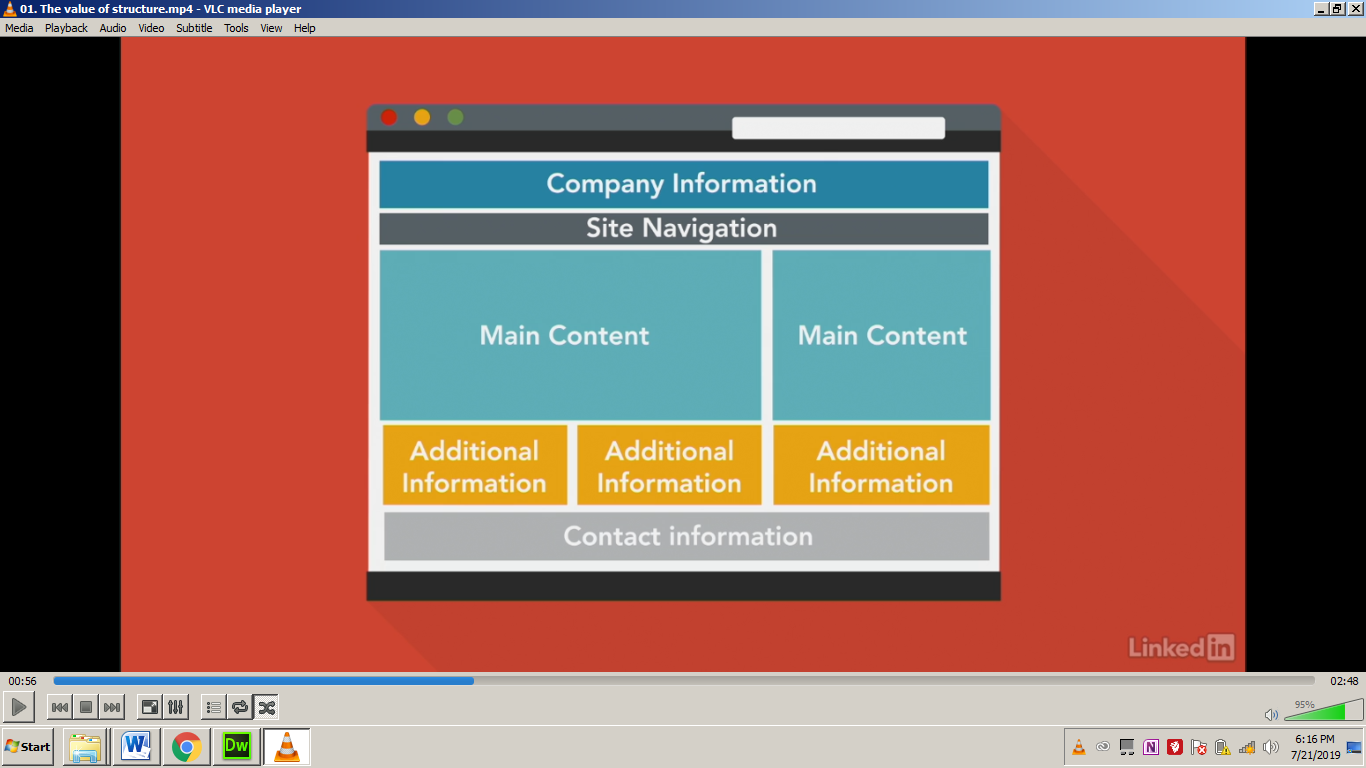
**Src**-it is attribute used to tell the browser where to find the image.

**Alt**-it allows you to pass along descriptive text representing the image.

<img=src”warrior-pose.jpeg” width=”200” height=”200” alt=”warrior-pose”>

**Structuring Content**

**0.1 standard layout for website**





**0.2** **Controlling document heading**

In this, headings are used systematically and how the content is organized has been taught in this video.

**0.3** **The nav element**

The nav element represent the section of a page that links to the other pages or to parts with in the page.

The <nav> element is used to contain primary navigation links.

**0.4** **The article element**

Article element is used to give structure to the content.

**0.5 The section element**

The <section> tag contains a grouping of related content, such as the chapter of a book. For example, you could divide the content of a home page into three sections: one for an introduction to the site, one for contact information, and another for latest news. The theme of each section should be identified, typically by including a heading (h1-h6 element) as a child of the section element.

**0.6** **The aside element**

The aside element is designed for side panel.

The <aside> tag holds content that is related to content around it. A sidebar in a print magazine is an example of the type of content that would go into an <aside>.

**0.7 The div element**

* The div element has no special meaning at all. It represents its children. It can be used with **class**, **lang**, and **title** attributes to markup semantics common to a group of consecutive elements.
* It is a enormous tag. It is used for grouping and structing content.
* <div class=”book”>
* **We can use 1 ID in one style sheet but class can be used as many as we required.**

**0.8 The header element**

The header element represents the introductory content for its nearest ancestor sectioning content or sectioning root element. A header typically contains a group of introductory or navigational aids.

**The footer element**

The footer element represents a footer for its nearest ancestor sectioning content or sectioning root element. A footer typically contains information about its section such as who wrote it, links to related documents , copyright data, and the like.

The <footer> tag contains information you’d usually place in a page’s footer, like a copyright notice, legal information, some site navigation links, and so on. You’re not limited to just a single <footer> per page, though; you can put a footer inside an <article>, for example, to hold related information like foot­notes, references, or citations.

**The main element**

The main element represents the main contents of the body of a document or application. The main content area consists of content that is directly related to or expands upon the central topic of the document or central functionality of an application.

The main content area of a document includes content that is unique to that document and excludes content that is repeated across a set of documents such as site navigation, links, copyright information, site logos and banners and search forms(unless the document or applications main function is that of a search form).

* It helps in semantic information…seo n all

**0.9 WAI-Aria in HTML**

<header role=”banner”>

<nav role=”navigation”>

<main role=”main”>

<article role=”article”>

<aside role=”complementary”>

<footer role=”contentinfo”>

* With this the page become more accessible…

1. Meaningful structure
2. Semantic(meaningful valid structure) meanings
3. consistency

**Creating Links**

* 1. **Exploring the anchor element**

For creating links we use **<a>** anchor element **</a>**

**href**= hypertext reference

**target**= target is used to open a page in a new window or tab or control where the page opens with in frameset. POINTS to remember- same site pe \_blank use nahi karna, one site se another site k link dene par \_blank use karna hai taki actual site apni identity na khoye.

Target=\_blank

\_parent framename

\_self

\_top

**rel** attribute describes the relationship of a target object to the link object.

rel=alternate next

author nofollow

bookmark noreferrer

download\* prefetch

help prev

license search

tag

<a href= “seed.html” target= “\_blank” rel= “next” title= “planting a seed”> Planting a seed</a>

Title is descriptive text.

**Linking to pages within your site**

**<a href=”about.html” target=”\_blank” title=”branches of yoga”>Hatha Yoga</a>**

**<a href=”../about.html” target=”\_blank” title=”up one directory”>Gyan Yoga</a>**

**<a href=”../../about.html” target=”\_blank” title=”up two directory”>Bhakti Yoga</a>**

**Linking to external pages**

**<a href= "https://8limbsyoga.com/about-yoga/" target="\_blank" title="Yoga"> <b>Know about Yoga</b> </a>**

**Linking to downloadable resources**

**<a href=”\_assets/syntax.zip” title=”HTML syntax reference”> HTML Syntax references</a>**

**For reading**

**<a href=”\_assets/syntax.pdf” title=”HTML syntax reference”>HTML Syntax references</a>**

**For download(It is used for downloading)**

**<a href=”\_assets/syntax.pdf” title=”HTML syntax reference” download=”HTML\_syntax”> HTML Syntax reference! </a>**

**0.5 Linking to page regions**

**<h1 id=”#top”>Yoga</h1>**

**<h2 id=”#meaning”>Meaning of Yoga</h2>**

**<h2 id=”#limbs”>What are the limbs of Yoga</h2>**

**<h2 id=”#benefits”>Benefits of Yoga</h2>**

**<h2 id=”#branches”>Branches of Yoga</h2>**

**How to give link? For ex**

**<p><a href=”#top” title=”back to the top”>Back to top</a></p>**

**Creating Lists**

* 1. **Unordered lists- Grouping of list items in no specific order.**

**<ul>**

**<li>Item one</li>**

**<li>Item two</li>**

**<li>Item three**

**<ul>**

**sublist<li>item 3.1</li>**

**<li>item 3.2</li>**

**<li>item 3.3</li></ul></li>**

**<li>Item four</li>**

**</ul>**

**OUTPUT**

* **Item one**
* **Item two**
* **Item three**
* **Item 3.1**
* **Item 3.2**
* **Item 3.3**
* **Item four**

**Ordered List- Grouping of list items in a specific order**

**<ol start=”1” reversed>**

**<li>Item one</li>**

**<li>Item two</li>**

**<li>Item three**

**<ol type=”i”>**

**sublist<li>item 3.1</li>**

**<li>item 3.2</li>**

**<li>item 3.3</li></ol></li>**

**<li>Item four</li>**

**</ol>**

**OUTPUT**

1. **Item one**
2. **Item two**
3. **Item three**
   * 1. **Item 3.1**
     2. **Item 3.2**
     3. **Item 3.3**
4. **Item four**
   1. **Definition list- Grouping of terms and description**

**<dl>description list</dl>**

**<dt>term</dt>**

**<dd>description</dd>**

**For ex**

**<dl>**

**<dt>Html</dt>**

**<dd>Hypertext markup language</dd>**

**</dl>**

**HTML and CSS**

**CSS- Cascading style sheet**

**<style> </style>**

**<!\_ \_ <style> this way we can off the style sheet and ON the another one.**

**</style>\_ \_>**

**0.2 Inline Style**

**<body>**

**<h2 style=”color:red;”**

**font-weight:”normal;”> Style me using the style element </h2>**

**</body>**

* 1. **Embedded styles**

**<html>**

**<head>**

**<meta charset=”utf-8”>**

**<title>embedded styles</title>**

**<style>**

**h2.alert {**

**color:red;**

**font-weight:normal;**

**}**

**</style>**

**</head>**

**<body>**

**<h1> sandhya </h1>**

**<h2 class=”alert”> she is beautiful </h2>**

**</body>**

**</html>**

* 1. **Controlling typography**
  2. **Adding color 0-255 Red, Green, Blue**

**body{**

**width:80%;**

**margin:0 auto;**

**font:100% Arial, sans-serif;**

**}**

**h1 {color: red;**

**font-family: Cambria, "Hoefler Text", "Liberation Serif", Times, "Times New Roman", "serif";**

**font-size: 2em;**

**font-weight: normal;**

**font-style: italic;**

**}**

**.paragraph{ color: #880808;**

**line-height: 1.6;**

**text-align: justify;**

**width: 70%;**

**margin: 0;**

**margin-bottom: 1em;**

* 1. **External style sheet**

**Basic Scripting**

**Javascript**

* **Validating forms**
* **Remote Scripting**
* **Special Effects**
* **Improving navigation**
  1. **The script element**

**In html document, if we want to add styling to the document we use style element likewise if we want to add scripting to the document we use applinamed scripting element**

**<script type=”text/javascript”>**

**//tabbled panels**

**</script>**

**Points to remember**

       Margin

Padding

Content

Content

**Cascading- It means that styles can fall from one style sheet to another, enabling multiplestyle sheets to be used on one HTML document.**

**Void/Self closing document- Void elements don’t have closing tags and don’t wrap any content because they are the content.**

**\* Previous xhtml required self close <hr/> but html 5 does not.**

**<area>, <base>, <br>, < col>, <embed>, <hr>, <img>, <input>, <keygen>, <link>, <menuitem>, <meta>, <source>, <track>, <wbr>**

**DOM (Document Object Model)**

**Body**

**h1----------------------P-----------------------UL**

**siblings siblings**

**li li li (Children of UL)**

**Text Editor**

**HTML, Notepad, Visual Studio, Sublime Text, Brackets, Atom**

**HTML Comment**

**<!\_\_// Intro\_\_>**

**External CSS**

* **A separate css file with a.css file extension.**
* **Always reference within the <head>**
* **Uses the <link> and two attributes, rel and href**

**<link href="styles.css" type="text/css" media="all" rel="stylesheet" />**

* **Use lower case – index.html**
* **Don’t use spaces or symbol = @,&**
* **Use dashes=\_,-**

**If style is saved in a folder then**

**<link href=”css/styles.css” rel=”styles.css>**

**For ex:- .P {**

**Color:blue**

**}**

**Here, P is selector**

**{ - declaration blocks**

**}**

**Color is property**

**Blue is value**

* **The same class can be used multiple times per page.**
* **Use whitespace to make the css easier to read.**

**ID Selector**

* **Ids can only be used once per page. IDs should have unique value.**
* **The value is the selector started with a # symbol.**
* **Multiples Ids can’t be used in the same Html element.**

**Default font size is 16 px.**

**CSS**

* **Cascading style sheet-Different language from HTML.**
* **Stylesheet language with its own syntax rules.**
* **Type selectors match the html by using the element name**
* **This way we can organize project**

**Project-name**

**!\_ \_css/**

**!\_ \_styles.css**

**!\_ \_images/**

**!\_ \_profile.jpg**

**!\_ \_index.html**

**Multiple classes**

* **Separate multiple classes with a space.**
* **Apply different styles to each class separately.**
* **Combine classes, with no space to select both**

**.style {**

**Color:blue;**

**}**

**.name {**

**Font-size:16px;**

**}**

**.style.name {**

**Background:green;**

**}**

**Pseudo-class Selectors**

**Descendent selectors**

* **Use descendent selectors, separated by a space, to match the descendent elements**

**<header> main h2{**

**<h1>header1</h1> color: blue;**

**<h2>header2</h2> }**

**<main>**

**<section> Output**

**<h2>section2</header> header1**

**</section> header2**

**<article> section2**

**<h2>article2</header> article2**

**</article>**

**</main>**

**The more selector we use the more specific the parent become.**

**For ex:- section p a {**

**Color: red;**

**}**

**Grouping multiple selectors.**

**\*we can also target multiple elements by grouping them in one declaration block. This will apply the same style to every selector. Each selector is separated with a comma. It helps to create more sufficient CSS bz it reduces the number of declarations, it also easier to update one declaration block.**

**<main> h1,h2,h3 {**

**<section> color:green;**

**</section> }**

**<article>**

**</article>**

**</main>**

**Each selector in the group is independent of each other.**

**.class-name h1,h2 { .class-name h1{**

**Font-size: 12px; font-size: 12px;**

**} }**

**h2{**

**Font-size: 12px;**

**}**

**Pseudo-class selectors**

**a{**

**background: purple;**

**padding:16px;**

**}**

**a:link {**

**color:white;**

**}**

**a:visited {**

**background: light blue;**

**}**

* **When you visit any link then its color is changed with above class.**

**Hover is not only used for a tag it’s used in other also like P tag.**

**a:active {**

**border: 1px solid black;**

**}**

**a:hover {**

**background:none;**

**}**

**P:hover {**

**Background: white;**

**}**

* **Add comment in pc**

**Ctrl + / (forward slash)**

**Video 16**

**ID values can only be used once per page; use them for unique or global styles that are not repeated.**

**#global-footer {**

**background:white;**

**}**

**\*use IDs for page linking and class for styling.**

**17. CSS Comments**

**/\* \*/**

**Usage**

* **Leave notes for yourself and others**
* **Organize code blocks**
* **Comment out code to hide it temporarily**
* **Css comments are used to keep the things organized.**

**18. CSS Color value**

**color: black;**

**Hex code: is always start from**

**# followed by six characters**

**Color: #000000;**

**Color :rgb(0,0,0) RGB stands for red, green and blue values on scale between 0 and 255**

**Sources- colours.neilorangepeel.com , randoma11y.com**

**And coolers.co**

**20. CSS Inheritence**

**Styles can be inherited from ancestor (parent) to descendant (child)**

**Elements.**

**But few elements won’t be inherited from parent like link (<a> tag</a>) bz it has its own specific default style.**

**CSS Specificity**

**The selector with the highest specificity will be applied.**

* **Type selectors (h1) – lowest**
* **Class selectors (.example)**
* **ID selectors (# example) – highest**

**For ex- <h1 class=.example id=#example> Selector specificity </h1>**

**Here, Id selector will be applied. It will overwrite both type and class selectors.**

**21. font and font-family property**

**Typography: the study of the design and use of type for communication.**

* **Serif and sans serif typefaces**

1. **Serif typefaces**

* **Serif typefaces have small decorative lines.**

Georgia

Rockwell

Times New Roman

1. **Sans-serif typefaces**

* **Sans-serif typefaces have no decorative lines.**

Verdana

Arial

Hervetica

1. **Script and decorative typefaces**

* **Script typefaces have a hand-lettered looks.**

**Brush Script std**

**Arizonia**

**Brush Script MT**

1. **Decorative typefaces and distinct and ornamental.**

**Copperplate Gothic Bold**

**Papyrus**

1. **Monospace typefaces**

**In monospace typeface, each character uses the same amount of horizontal space.**

**Courier New**

Sans **means** “without,” so sans serif **fonts** are those without the **decorative lines** at the end of each character. These **fonts** tend to look more modern, and are extremely readable. This is why they are popularly used for digital content!

**Typeface is a set of fonts, designed with common characteristics.**

**22. Web fonts and Google fonts**

**Web safe fonts-** commonly preinstalled on devices and computers

Limited options

Have no control over what users have installed.

**Web fonts-** not required to be installed on devices and computers

**23. Font size: px, em and rem**

There are different units for specifying the font size. Some are relative value and some are absolute value.

Relative values are calculated based on the nearest ancestor (parent) element.

Absolute values are not affected by ancestor elements.

Px:

* Measure screen in pixels.
* Absolute value is great for accuracy.
* Browser default – 16px

Font size of em and rem have confusion.

24. Practicing with web fonts and font-size

Reference site for fonts- <https://fonts.google.com/>

**25.The font-style and font-weight properties**

**1. Font-weight**

* **Font-weight is the thickness or boldness**
* **Number values are 100, 200, 300, 400, 500, 600, 700, 800 and 900**
* **100 is the lightest or thinnest and 900 is the darkest or thickest among all.**
* **Keywords can also used for font-weight numbers-**

**Normal = 400;**

**Bold = 700**

**2. Font-style**

* **Used to add or remove an italic style.**
* **Three values: italic, oblique, and normal.**

**26. The color, line-height, and text properties**

**The color – The color property changes the text color and uses a variety of values.**

**h1{**

**color: black; - keywords**

**color: #000000; - hex code**

**color: rgb (0,0,0); - rgb**

**Line-height – Line height property sets the height of the space between two lines of text.**

**It is closely related to font-size.**

**If line the height set same as font-size value, there will be no space between the lines, which would make hard to read. So, line height value should always be larger than the font-size.**

**Line height also accepts unit list value for ex line-height: 1.5;**

**Text-decoration – The text- decoration property can be used to remove or add an underline above, below, or through the text.**

**text-decoration – overline;**

**text-decoration – underline;**

**text-decoration – line-through;**

**text-decoration – none;**

**Text-transform – The text transform property specifies the letter casing.**

**Text-Transform – Capitalize;**

**TEXT-TRANSFORM – UPPERCASE;**

**text-transform – lowercase;**

**text-transform – none;**

**Text-align – The text-align property can be used to center align text.**

**You can add it to the HTML element itself or the parent element.**

**28. There are two types of element – Block and Inline**

**Block elements**

* **Height = content**
* **Width = 100% of the parent container/container**
* **Element start on a new line**
* **Block element can wrap other block and inline element.**
* **For ex. <div></div>, <p></p>, <h1> all headings </h1>**

**Inline elements**

* **Height and width = content**
* **Height and width have not effect on inline element. (to solve this, need to set as display:inline-block;)**
* **Elements align left, in a line.**
* **Inline elements can only nest other inline elements (except <a> tags in html5)**
* **Ex <a>, <span>, <Strong>**

**Some common use display values are block, inline, inline-block and none.**

**29. The Box Model**

* **width and height: sets specific size of the content box**
* **padding: space inside of the element**
* **margin: space outside of the element**
* **border displays between the padding and margin**
* **The width and height property are used to change the default size of block and inline elements**

**Inline element Block element**

**Span { div {**

**Width: 100px; width: 100px;**

**Height: 100px; height: 100px;**

**display: block; }**

**or**

**display: inline-block;**

**}**

* **Inline element requires the display property.**

**30. Margin and page layouts**

* **By default, block elements stack on top of each other.**
* **Negetive margin values are used to move elements outside of the stacking position.**
* **For ex margin: -20px 0 0 -20px; it will push the box from stacking position.**
* **Use margin to center align content blocks**

**div {**

**width: 950px;**

**margin: 0 auto;**

**}**

* **Creating a content wrapper**

**Create a container for just the content**

**<section>**

**<div class=”content-wrap”>**

**Work Experience**

**</div>**

**</section>**

**CSS**

**.content-wrap {**

**Width: 950px;**

**Margin: 0 auto;}**

**31. practice done**

**32. Practicing with padding and spacing**

**33. Page layouts**

**Float – float is another property can be used how elements are rearranged.**

**Css– float : right;**

**Clearing floats- Floats must be cleared to return to the natural page flow.**

**Clear: both;**

**Need to understand from Jitu**

**34. Practicing with floats \_ little understand this too of last**

**Images can be resize with css but it is always good to crop in which size is required.**

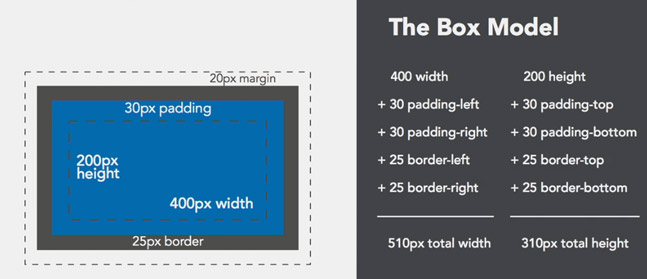
**Border-radius makes the pic look circular.**

**Css- border-radius: 50%; (for circle), 6px; (for slightly round corners)**

**35. The box model**

**The width, height, padding and border all contribute to the total size of the element.**

**Margin adds space around the element but does not affect the size.**

****

**Floating images and text**

* **If the text is longer than the image, the text will flow underneath.**
* **If we don’t want to flow the text underneath then we need to create a container to group the text content to keep it from flowing underneath.**
* **The box model fix need to understand from Jitu.**

**36. Practicing with columns**

**Second Phase**

**0.5 Basic and attribute selectors**

**Type attribute selector - having problem in its example.**

**0.5 Combinator selectors**

**Combinators are selectors that combine in various ways to make more specific based on Html structure.**

**Descendant selectors – are based on the relationship of the nested elements.**

**<div class=”ancestor”>**

**<div class=”parent”>**

**<div class=”child”></div>**

**<div class=”descendant”></div>**

**</div>**

**</div>**

**Need to understand this video from Jitu.**

* 1. **Pseudo class-selectors**

**A Pseudo-class is a keyword, added to a selector with the colon symbol. It’s used to specify a certain state.**

**We can use multiple classes by giving a space in them.**

***/\*applies to all the links\*/***

**a {**

**color:black;**

**}**

***/\*only applies on mouse over\*/***

**a:hover {**

**text-decoration:none;**

**}**

**We can also use pseudo classes for selecting a specific child element based to its order.**

**: first-child- selects the first child element of its parent.**

**: last-child- selects the last child element of its parent.**

**p: first-child {**

**color: red;**

**}**

**p: last-child {**

**color: blue;**

**}**

**<section>**

**<p> paragraph. </p> *<!\_ \_ this one will be red \_ \_!>***

**<p> paragraph. </p>**

**<p> paragraph. </p> *<!\_ \_ this one will be blue \_ \_!>***

**</section>**

**:first-of-type and :last-of-type**

**p: first-of-type {**

**color: red;**

**}**

**p: last-of-type {**

**color: blue;**

**}**

**<section>**

**<h1>heading</h1>**

**<p>paragraph.</p> *<!\_ \_ this one will be red \_ \_!>***

**<p>paragraph.</p>**

**<p> paragraph. </p> *<!\_ \_ this one will be blue \_ \_!>***

**<span>span element </span>**

**</section>**

**:nth-child ( ) is used to select one or more child elements based on the order within the parent container.**

**Elements are selected by passing in an argument in one of the three ways: keyword, number, or algebraic expression.**

**:nth-child (odd)**

**:nth-child (3)**

**:nth-child (2n+1)**

**\* rest video need to clear from Jitu**

**0.7 Pseudo element selector**

**0.8 Practicing with advanced selectors – done – Good video**

**0.9 Box model review**

**10 Float and display review**

**11 Horizontal navs with display properties**

**12 Horizontal navs with the float properties – It would be understand more with projects.**

**13 Practicing with the nav element – Practice done**

**14 Positioning – example of this video need to understand from Jitu**

**Position is used to arrange elements relative to the default page flow or browser viewport.**

**There are five values: relative, absolute, fixed, static and inherit.**

**Postion is also used with a combination of offset properties:**

**top, right, bottom, left.**

**.box{**

**position: relative;**

**top: 10px;**

**}**

**15 Practicing with fixed navigation-practice is left.**

**16 Practicing with positioning elements.**

**17 How to know when to use float, display or position?**

**Float**

* **If you have variable and flexible content (e.g. image surrounded by text or blog posts with different lengths ) then float is a good option.**
* **Float is also good for global or large page structures ( e.g. header, footer or sidebar)**

**Display**

* **Display can also be used on line-block on a page similar to float. Aligning page components ( make sure to account for the extra space)**
* **Aligning elements that need to be center aligned to the child element.**
* **Also, does not worry to change the page’s natural page flow.**

**Positioning**

* **Positioning can be used when positioned relative to the another element.**
* **It is also useful for the aligning elements outside of the document flow.**
* **Positioning elements to a specific spot in the document.**
* All three can’t be used on same element.
* float, display and position can't be used all together on the same element. If using float, then display is iqnored and if using display, then position is iqnored.

**18. Layers and the z-index property – the stacking context**

**19. Browser developer tools – inspect any webpage and test your front-end code. You can inspect any page. But these changes does not save as you refresh changes will disappear.**

**20. Debugging CSS-browser-inspect-css-**

**21. Resetting Stylesheet**

**Reset Stylesheet – Prewritten stylesheet containing rules that override all the default browser styles to an un-styled baseline.**

**Normalize- Prewritten stylesheet containing rules that aim to create consistent default styles, rather than removing them.**

**22. Icon fonts**

**23. The background property**

**Css**

**.background {**

**height: 400px;**

**background-color: black;**

**background-image: url (http://....**

**background-repeat: no-repeat;**

**background-position: 20px 2%;**

**Or (Keyword) - : top-right;**

**background-attachment: fixed;**

**background-size: 100%**

**Or (keyword): cover;**

**24. Background shorthand method**

**Selector {background: blue url(Image.jpg) no-repeat fixed 0px 0px;**

**The order of the values doesn’t matter, \*but make sure to separate the values with a space.**

**When using background-size, it must be included after background-position, with a forward slash (/).**

**Selector {background: blue url(image.jpg) no-repeat fixed 0px 0px / cover;**

**Add background-size after the shorthand declaration, if not using background-position.**

**Selector {**

**background: blue url(Image.jpg) no-repeat;**

**background-size: cover;**

**}**

**Make sure background-size after the shorthand declaration bz in this case it really does matter.**

**If the value of the property is not declared, the default value will be assumed.**

**Mixing shorthand and longhand values may result in values getting overwritten.**

**25. Alpha transparency and gradients-notes in diary.**

**26. Practicing with backgrounds and gradients-its yet to be done.**

**27. Introduction to responsive design – need to discuss with Jitu**

**28. Mobile friendly and mobile first - need to discuss with Jitu**

**29. Creating flexible and fluid layouts- need to discuss with Jitu**

**30. Introducing media queries - need to discuss with Jitu**

**31. Using media queries- need to discuss with Jitu**

**32. Testing responsive layouts- need to discuss with Jitu**

**33. Device emulation- need to discuss with Jitu**

**Points to remember**

**1. <div> It is use to make divisions. This is block element. It wrap inline and block element. We can put Paragraph tag in div but not div tag in paragraph (<p> </p>).**

**The <div> and <span> tags have been around for much of the life of the Web. They’ve traditionally been used to organize and group content that doesn’t quite lend itself to other HTML tags. Think of them as like empty vessels that you fill with content. A div is a block, meaning it has a line break before it and after it, while a span appears inline, as part of a paragraph. Otherwise, divs and spans have no inherent visual properties, so you can use CSS to make them look any way you want.**

**More often it’s used to group any number of other elements, so you can insert a headline, a bunch of paragraphs, and a bulleted list inside a single <div> block. The <div> tag is a great way to subdivide a page into logical areas, like a banner, footer, sidebar, and so on.**

**The <span> tag is used for inline elements: words or phrases that appear inside a larger paragraph or heading.**

**<div>Sample <span>test</span></div>**

**Output – Sample test**

**<div>Sample<p>test</p></div>**

**Output – Sample**

**test**

**Because P tag is also block element but it is used for writing test and inline items like strong, b etc. basically P only wrap inline elements. P can’t wrap block elements.**

**2. Font size is inherited property; means in between wherever font-size is define that further implemented.**

**1em (emphasize) = 16px , 1rem (root-emphasize) = 16px**

**em emphasize on parent tag and rem emphasize on root.**

**3. Percentage also works like em and follow parent tag, but there is one difference**

**4. Float is inline element. It works from left to right. When we use display: inline-block then content of element of parents float but with this 1 issue arises that background color displays between buttons. So for solving this issue we use float: left; and make another div container with style= “clear: both;” For ex**

**<div style="background: red;">**

**<div style=" background: green; float: left;">test 1</div>**

**<div style=" background: yellow; float: left;">test 2</div>**

**<div style=" background: brown; float: left;">test 3</div>**

**</div>**

**<div style="clear: both;"></div>**

**Clear: both is sibling of float element and last child of parent.**