

LECTURE NOTES
ON

Web Designing
2301CS202

B.Tech. 2nd Semester
(Darshan Institute of Engineering and Technology)

Prepared By

Vijay Shekhat
Assistant Professor
vijay.shekhat@darshan.ac.in
9558045778



Table of Content

Unit-1.....	4
Introduction to Web Technology	4
1.1. Internet and Web	4
1.2. Client Server Architecture	5
1.3. Introduction to Web Technologies.....	7
1.4. Concept of Effective Web Design	8
1.5. Web Design Issues	8
1.6. Tips for Effective Navigation.....	10
Unit-2.....	11
HTML	11
2.1. Introduction to HTML	11
2.2. Basic HTML Tags	12
2.3. <basefont>(Not supported in HTML5)	21
2.4. Anchor Tag.....	21
2.5. HTML Images	22
2.6. Table	24
2.7. HTML Forms.....	31
2.8. <!--comment-->	36
2.9. Other Tags	36
2.10. Meta Tag.....	38
2.11. Multimedia Tags	39
2.12. HTML5.....	41
2.13. Elements of HTML5	42
2.14. Other HTML5 Tags	43
2.15. HTML5 Attribute	44
2.16. HTML5 Input Types.....	49
2.17. HTML5 Form Validation.....	53
Unit-3.....	54
CSS	54
3.1. Introduction to CSS.....	54
3.2. Basic Syntax and Structure of CSS	54
3.3. Types of CSS.....	55
3.4. Targeting using CSS.....	58
3.5. Background Properties	61
3.6. Text Property	64

3.7.	Font Property.....	68
3.8.	Box Model.....	72
3.9.	Styling List.....	73
3.10.	Position Property	75
Unit-4.....		77
CSS3		77
4.1.	Introduction to CSS3.....	77
4.2.	Animations.....	77
4.3.	Tooltips	79
4.4.	Styling Images	80
4.5.	Media Query	82
4.6.	Wildcard Selector	86
4.7.	Variables	87
4.8.	Gradients	88
4.9.	Pseudo class and Elements.....	90
Unit-5.....		93
Bootstrap		93
5.1.	Introduction to Bootstrap.....	93
5.2.	Breakpoints.....	95
5.3.	Containers.....	95
5.4.	Bootstrap Grid System.....	96
5.5.	Utility Classes.....	99
5.6.	Typography	107
5.7.	Bootstrap Components	113

Unit-1

Introduction to Web Technology

1.1. Internet and Web

1.1.1. Internet

- The Internet is a massive network of networks, a networking infrastructure. It connects millions of computers together globally, forming a network in which any computer can communicate with any other computer as long as they are both connected to the Internet.
- Information that travels over the Internet does so via a variety of languages known as protocols. So, we can say that Internet is network of computer which connect to together and any computer communicate with any other computer.

1.1.2. WWW (World Wide Web)

- The World Wide Web, or simply Web, is a way of accessing information over the medium of the Internet. It is an information-sharing model that is built on top of the Internet.
- The Web uses the HTTP protocol, only one of the languages spoken over the Internet, to transmit data. The Web also utilizes browsers, such as Internet Explorer or Firefox, to access Web documents called Web pages that are linked to each other via hyperlinks. Web documents also contain graphics, sounds, text and video.

1.1.2.1 History of WWW

- Tim Berners-Lee, in 1980 was investigating how computer could store information with random links. In 1989, while working at European Particle Physics Laboratory, he proposed to idea of global hypertext space in which any network-accessible information could be referred to by single “universal Document Identifier”. After that in 1990, this idea expanded with further program and knows as World Wide Web.

1.1.2.2 Concept of WWW

- WWW is stands for World Wide Web.
- The World Wide Web (WWW) is a global information medium which users can read and write via computer connected to the internet.
- The Web, or World Wide Web, is basically a system of Internet servers that support specially formatted documents. The documents are formatted in a markup language called HTML (Hypertext Markup Language) that supports links to other documents, as well as graphics, audio, and video files.
- In short, World Wide Web (WWW) is collection of text pages, digital photographs, music files, videos, and animations you can access over the Internet.
- Web pages are primarily text documents formatted and annotated with Hypertext Markup Language (HTML). In addition to formatted text, web pages may contain images, video, and software components that are rendered in the user's web browser as coherent pages of multimedia content.
- The terms Internet and World Wide Web are often used without much distinction. However, the two are not the same.

- The Internet is a global system of interconnected computer networks. In contrast, the World Wide Web is one of the services transferred over these networks. It is a collection of text documents and other resources, linked by hyperlinks and URLs, usually accessed by web browsers, from web servers.
- There are several applications called Web browsers that make it easy to access the World Wide Web; For example: Firefox, Microsoft's Internet Explorer, Chrome Etc.
- Users access the World-Wide Web facilities via a client called a browser, which provides transparent access to the WWW servers.

1.1.2.3 Different between Internet and WWW

- The Web is a Portion of The Internet. The Web is just one of the ways that information can be disseminated over the Internet. The Internet, not the Web, is also used for email, which relies on SMTP, Usenet news groups, instant messaging and FTP. So the Web is just a portion of the Internet.

1.2. Client Server Architecture

- The Client-server model is a distributed application structure that partitions task or workload between the providers of a resource or service, called servers, and service requesters called clients.
- In the client-server architecture, when the client computer sends a request for data to the server through the internet, the server accepts the requested process and deliver the data packets requested back to the client. Clients do not share any of their resources. Examples of Client-Server Model are Email, World Wide Web, etc
- Client: It is a computer (Host) i.e., capable of receiving information or using a particular service from the service providers (Servers).
- Servers: It is a remote computer which provides information (data) or access to particular services.

1.2.1. Communication between Web Browser and Web Server

- There are few steps to follow to interacts with the servers a client.
- User enters the URL (Uniform Resource Locator) of the website or file. The Browser then requests the DNS (DOMAIN NAME SYSTEM) Server.
- DNS Server lookup for the address of the WEB Server.
- DNS Server responds with the IP address of the WEB Server.
- Browser sends over an HTTP/HTTPS request to WEB Server's IP (provided by DNS server).
- Server sends over the necessary files of the website.
- Browser then renders the files and the website is displayed. This rendering is done with the help of DOM (Document Object Model) interpreter, CSS interpreter and JS Engine collectively known as the JIT or (Just in Time) Compilers.

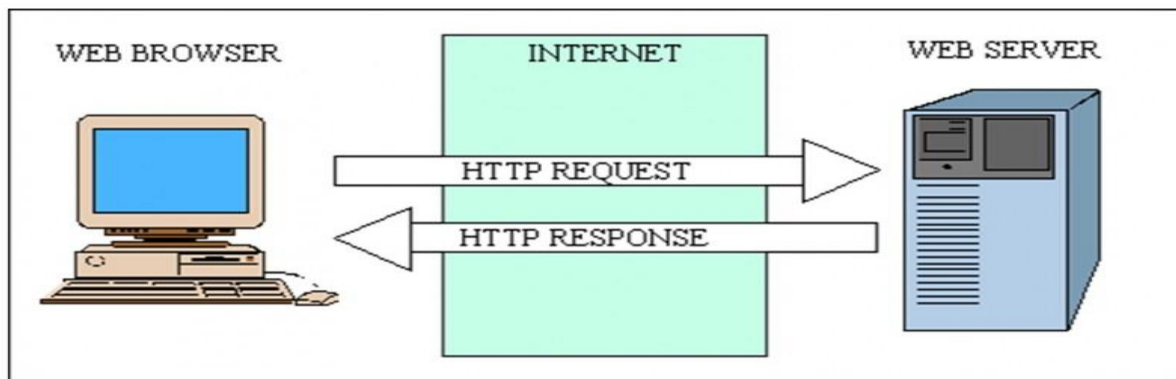


Figure 1.2.1 Communication between Web Browser and Web Server

1.2.2. Advantages of Client-Server model

- Centralized system with all data in a single place.
- Cost efficient requires less maintenance cost and Data recovery is possible.
- The capacity of the Client and Servers can be changed separately.

1.2.3. Disadvantages of Client-Server model

- Clients are prone to viruses, Trojans and worms if present in the Server or uploaded into the Server.
- Server are prone to Denial of Service (DOS) attacks.
- Data packets may be spoofed or modified during transmission
- Phishing or capturing login credentials or other useful information of the user are common and MITM (Man in the Middle) attacks are common.

1.2.4. HTTP Protocol: Request and Response

- HTTP stands for Hypertext Transfer Protocol.
- HTTP is based on the client-server architecture model and a stateless request/response protocol that operates by exchanging messages across a reliable TCP/IP connection.
- An HTTP "client" is a program (Web browser) that establishes a connection to a server for the purpose of sending one or more HTTP request messages. An HTTP "server" is a program (generally a web server like Apache Web Server) that accepts connections in order to serve HTTP requests by sending HTTP response messages.
- Errors on the Internet can be quite frustrating — especially if you do not know the difference between a 404 error and a 502 error. These error messages, also called HTTP status codes are response codes given by Web servers and help identify the cause of the problem.
- For example, "404 File Not Found" is a common HTTP status code. It means the Web server cannot find the file you requested. The file -- the webpage or other document you try to load in your Web browser has either been moved or deleted, or you entered the wrong URL or document name.
- HTTP is a stateless protocol means the HTTP Server doesn't maintain the contextual information about the clients communicating with it and hence we need to maintain sessions in case we need that feature for our Web-applications
- HTTP header fields provide required information about the request or response, or about the object sent in the message body. There are four types of HTTP message headers:
 - General-header: These header fields have general applicability for both request and response messages.

- Request-header: These header fields have applicability only for request messages.
- Response-header: These header fields have applicability only for response messages.
- Entity-header: These header fields define Meta information about the entity-body.
- As mentioned, whenever you enter a URL in the address box of the browser, the browser translates the URL into a request message according to the specified protocol; and sends the request message to the server.
- For example, the browser translated the URL `http://www.test101.com/doc/index.html` into the following request message:

```
GET /docs/index.html HTTP/1.1
Host: www.test101.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
```
- Here, Step by step communication between client and server mention into following figure.

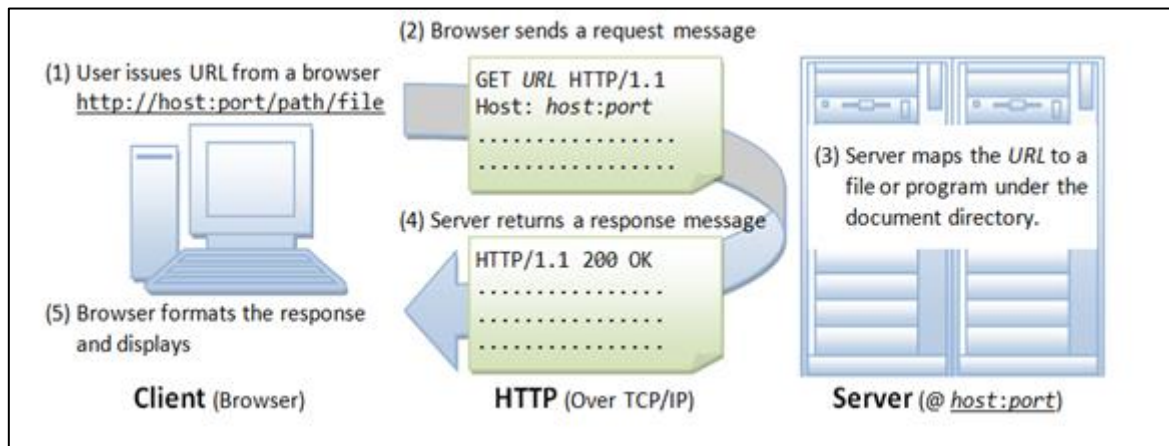


Figure 1.2.4 Step by step communication between client and server using Http

1.3. Introduction to Web Technologies

- **HTML:** - HTML stands for Hypertext Markup Language. It is used to design the front-end portion of web pages using a markup language. HTML is the combination of Hypertext and Markup language. Hypertext defines the link between the web pages. The markup language is used to define the text documentation within the tag which defines the structure of web pages.
- **CSS:** - *Cascading Style Sheets fondly referred to as CSS is a simply designed language intended to simplify the process of making web pages presentable. CSS allows you to apply styles to web pages. More importantly, CSS enables you to do this independent of the HTML that makes up each web page.*
- **JavaScript:** - JavaScript is a famous scripting language used to create magic on the sites to make the site interactive for the user. It is used to enhancing the functionality of a website to running cool games and web-based software.
- **Bootstrap:** - Bootstrap is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites. Bootstrap is completely free to download and use! The primary

purpose of adding it to a web project is to apply Bootstrap's choices of color, size, font and layout to that project. As such, the primary factor is whether the developers in charge find those choices to their liking. Once added to a project, Bootstrap provides basic style definitions for all HTML elements. The result is a uniform appearance for prose, tables and form elements across web browsers. In addition, developers can take advantage of CSS classes defined in Bootstrap to further customize the appearance of their contents. For example, Bootstrap has provisioned for light- and dark-colored tables, page headings, more prominent pull quotes, and text with a highlight.

- **Material Design:** - Material is a design system created by Google to help teams build high-quality digital experiences for Android, iOS, Flutter, and the web. Material Design is inspired by the physical world and its textures, including how they reflect light and cast shadows. Material surfaces reimagine the mediums of paper and ink.

1.4. Concept of Effective Web Design

- It's a good idea to first think about and design your site. That way, you'll give yourself direction and you'll need to reorganize less later.
- Consider the following step to design a website effectively
- Figure out why you're creating this site. What do you want to convey?
- Think about your audience. How can you tailor your content to appeal to this audience? For example, should you add lots of graphics or is it more important that your page download quickly?
- How many pages will you need? What sort of structure would you like it to have? Do you want visitors to go through your site in a particular direction, or do you want to make it easy for them to explore in any direction?
- Sketch out your site on paper.
- Devise a simple, consistent naming system for your pages, images and other external files.

1.5. Web Design Issues

1.5.1. Browser & Operating Systems

- Web pages are written using different HTML tags and viewed in browser window.
- The different browsers and their versions greatly affect the way a page is rendered, as different browsers sometimes interpret same HTML tag in a different way.
- Different versions of HTML also support different sets of tags.
- The support for different tags also varies across the different browsers and their versions.
- Same browser may work slightly different on different operating system and hardware platform.
- To make a web page portable, test it on different browsers on different operating systems.

1.5.2. Bandwidth and Cache

- Users have different connection speed, i.e., bandwidth, to access the Web sites.
- Connection speed plays an important role in designing web pages, if user has low bandwidth connection and a web page contains too many images, it takes more time to download.
- Generally, users have no patience to wait for longer time than 10-15 seconds and move to other site without looking at contents of your web page.
- Browser provides temporary memory called *cache* to store the graphics.
- When user gives the URL of the web page for the first time, HTML file together with all the graphics files referred in a page is downloaded and displayed.

1.5.3. Display Resolution

- Display resolution is another important factor affecting the Web page design, as we do not have any control on display resolution of the monitors on which user views our pages.
- Display or screen resolution is measured in terms of pixels and common resolutions are 800 X 600 and 1024 X 786.
- We have three choices for Web page design.
 - Design a web page with fixed resolution.
 - Make a flexible design using HTML table to fit into different resolution.
 - If the page is displayed on a monitor with a higher resolution, the page is displayed on left-hand side and some part on the right-hand side remains blank. We can use cantered design to display page properly.

1.5.4. Look & Feel

- Look and feel of the website decides the overall appearance of the website.
- It includes all the design aspects such as
 - Web site theme
 - Web typography
 - Graphics
 - Visual structure
 - Navigation etc...

1.5.5. Page Layout and Linking

- Website contains of individual web pages that are linked together using various navigational links.
- Page layout defines the visual structure of the page and divides the page area into different parts to present the information of varying importance.
- Page layout allows the designer to distribute the contents on a page such that visitor can view it easily and find necessary details.

1.5.6. Locating Information

- Webpage is viewed on a computer screen and the screen can be divided into five major areas such as center, top, right, bottom and left in this particular order.
- The first major area of importance in terms of users viewing pattern is the center, then top, right, bottom and left in this particular order.

1.5.7. Making Design User-Centric

- It is very difficult for any Web designer to predict the exact behaviour of the Web site users.
- However, idea of general behaviour of common user helps in making design of the Web site user-centric.
- Users either scan the information on the web page to find the section of their interest or read the information to get details.

1.5.8. Sitemap

- Many a times Web sites are too complex as there are a large number of sections and each section contains many pages.
- It becomes difficult for visitors to quickly move from one part to other.

- Once the user selects a particular section and pages in that section, user gets confused about where he/she is and where to go from there.
- To make it simple, keep your hierarchy of information to few levels or provide the navigation bar on each page to jump directly to a particular section.

1.6. Tips for Effective Navigation.

- Navigation links are either text based, i.e. a word or a phrase is used as a link, or graphical, i.e. a image, i.e. a icon or a logo is used as a link.
- Navigation links should be clear and meaningful.
- It should be consistent.
- Link should be understandable.
- Organize the links such that contents are grouped logically.
- Provide search link, if necessary, usually on top of the page. Use common links such as 'About Us' or 'Contact Us'.
- Provide the way to return to first page.
- Provide the user with information regarding location
- Horizontal navigation bar can be provided on each page to directly jump to any section

Unit-2

HTML

2.1. Introduction to HTML

- HTML stands for Hyper Text Markup Language
- HTML is the standard markup language for creating Web pages.
- Most documents that appear on the World Wide Web were written in HTML.
- HTML was developed by Tim Berners-Lee in 1991.
- HTML uses tags.
- These tags are understood by browser such as Mozilla, google chrome, internet explorer etc.
- It is supported by all browsers.
- Browsers do not display the HTML tags, but use them to produce the content of the page
- We can apply this markup language to your pages to display text, images, sound and movie files, and almost any other type of electronic information.
- We use the language to format documents and link them together, regardless of the type of computer with which the file was originally created.
- **Structure of HTML program: -**

```
<html>
  <head>
    <title>Page Title</title>
  </head>
  <body>
    <h1>My First Heading</h1>
    <p>My first paragraph. </p>
  </body>
</html>
```

2.1.1. HTML Elements

- An element consists of three basic parts: an opening tag, the element's content, and finally, a closing tag.

```
<p> it is an Opening paragraph tag
      Element Content-paragraph words
</p> it is a Closing tag
```

- Every web page should have four critical elements: the html, head, title, and body elements.

1. `<html></html>`
2. `<head></head>`
3. `<title></title>`
4. `<body></body>`

`<html>` tag

- It represents the root of an HTML document.
- The HTML document itself begins with `<html>` and ends with `</html>`.
- This tag is the container for all other HTML elements.

`<head>` tag

- It includes title of the document, scripts, style, links to scripts, links to CSS files, meta tags etc.
- Tags included inside head tags are not displayed on browser window.

`<title>` tag

- Place the `<title>` tag within the `<head>` element to title your page.
- The words you write between the opening and closing `<title></title>` tags will be displayed at the top of a viewer's browser.

`<body>` tag

- The visible part of the HTML document is between `<body>` and `</body>`.
- `<body>` tag is used to contain a web page's content, including hyperlinks, images, tables, text, etc.
- It should have in every HTML document and a webpage should have a single body tag.

2.2. Basic HTML Tags

- A web browser read an HTML document top to bottom, left to right.
- Each time the browser finds a tag, it is displayed accordingly (paragraph look like paragraph, table look like table etc).
- Tags have 3 major parts: opening tag(s), content(s), and closing tag(s).

2.2.1. Heading Tag

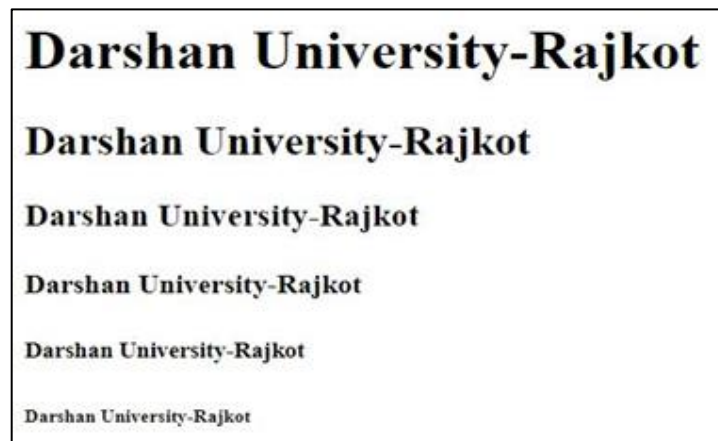
- Any document starts with a heading.
- HTML also has six levels of headings, which use the elements `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>`, and `<h6>`.

- While displaying any heading, browser adds one line before and one line after that heading.

Example: -

```
<html>
<body>
<h1>Darshan University-Rajkot</h1>
<h2>Darshan University-Rajkot</h2>
<h3>Darshan University-Rajkot</h3>
<h4>Darshan University-Rajkot</h4>
<h5>Darshan University-Rajkot</h5>
<h6>Darshan University-Rajkot</h6>
</body>
</html>
```

Output: -



2.2.2. Paragraph Tag

- The HTML <p> element defines a paragraph.
- Each paragraph of text should go in between an opening <p> and a closing </p> tag.
- Browsers automatically add some space (margin) before and after each <p> element.

Example: -

```
<p>Rajkot-Morbi Highway, <br>
    Hadala,<br>
    Darshan University-Rajkot
</p>
```

```
<p>Office No. 401, Lotus Arcade,<br>
    8-Royal Park, Near KKV Hall,<br>
    150 Feet Ring Road,<br>
    Rajkot - 360005, Gujarat (INDIA)
</p>
```

Output: -

Rajkot-Morbi Highway,
Hadala,
Darshan University-Rajkot

Office No. 401, Lotus Arcade,
8-Royal Park, Near KKV Hall,
150 Feet Ring Road,
Rajkot - 360005, Gujarat (INDIA)

2.2.3.
 tag

- The
 tag inserts a single line break.
- The
 tag is an empty tag which means that it has no end tag.
- Use the
 tag to enter line breaks, not to separate paragraphs.

2.2.4. <hr> tag

- It is useful when you are changing a topic or want to separate content on a page.
- It inserts horizontal line in HTML page.
- It does not require an end tag.

2.2.5. HTML Formatting Tag

- HTML Formatting is a process of arranging contents for better look.
- HTML formatting is used to design the page content part.
- HTML provides us ability to format contents without using CSS.
- There are many formatting tags in HTML which are listed below.

Table 3.4 HTML Formatting Tag

Tags	Description	Example
	Defines bold text	Bold Text

<code><i></code>	Defines italic text	<code><i>Italic Text</i></code>
<code><small></code>	Defines smaller text	<code><small>Small Text</small></code>
<code></code>	Defines important text	<code>Strong Text</code>
<code><sub></code>	The <code><sub></code> tag defines subscript text. Subscript text appears half a character below the baseline. Subscript text can be used for chemical formulas, like H ₂ O.	<code><p>H<sub>2</sub>O</p></code>
<code><sup></code>	The <code><sup></code> tag defines superscript text. Superscript text appears half a character above the baseline. Superscript text can be used for footnotes, like WWW	<code><p>WWW<sup>[1]</sup></p></code>
<code><mark></code>	Defines Highlighted text	<code><mark>Highlighted Text</mark></code>
<code></code>	Defines deleted text	<code><p>Price is 320 350 rs</p></code>
<code></code>	Defines emphasized text	<code>Emphasized Text</code>
<code><tt></code>	The <code><tt></code> tag defines teletype text	<code><p><tt>This text is teletype text</tt></p></code>
<code><blink></code>	The <code><blink></code> tag is used for blinking the text.	<code><blink>blinking text tag</blink></code>

2.2.6. List Tag

- Lists are used to group together related pieces of information.
- Lists are easy to read.
- There are three list types in HTML:
 1. Unordered list
 2. Ordered list
 3. Description list

Unordered list

- An unordered list is a collection of related items that have no sequence.
- Unordered list is created by `` tag.

- Each list item starts with the tag.
- Each item in the list is marked with a bullet.

Example: -

```
<ul>
  <li>B.Tech</li>
  <li>B.Sc.</li>
  <li>BCA</li>
  <li>BBA</li>
  <li>B.Com</li>
</ul>
```

Output:

- B.Tech
- B.Sc.
- BCA
- BBA
- B.Com

- **type attribute:-**(The type attribute is not supported in HTML5, instead of type you can use CSS property of list-style-type.)
- You can use type attribute for tag to specify the type of bullet you like.
- By default, it is a disc. Following are the possible options -

```
<ul type = "square">
```

```
<ul type = "circle">
```

Ordered List

- If you are required to put your items in a numbered list instead of bulleted, then HTML ordered list will be used.
- This list is created by using tag.
- The numbering starts at one and is incremented by one.

Example: -

```
<ol>
  <li>B.Tech</li>
  <li>B.Sc.</li>
  <li>BCA</li>
```



```
<li>BBA</li>
<li>B.Com</li>
</ol>
```

Output: -

```
1. B.Tech
2. B.Sc.
3. BCA
4. BBA
5. B.Com
```

- **type Attribute:** -By default, it is a number. Following are the possible options.

```
<ol type="1">
<ol type="A">
<ol type="a">
<ol type="I">
<ol type="i">
```

- **start Attribute:** -
- We can use start attribute for tag to specify the starting point of numbering we need.
- Start your ordered list on any number besides 1 using the start attribute. We can use start attribute for tag to specify the starting point of numbering we need.
- Start your ordered list on any number besides 1 using the start attribute.

Example: -

```
<ol start="4">
<li>B.Tech</li>
    <li>B.Sc.</li>
    <li>BCA</li>
    <li>BBA</li>
    <li>B.Com</li>
</ol>
```

Output: -

4. B.Tech
5. B.Sc.
6. BCA
7. BBA
8. B.Com

Definition(description) List

- HTML also supports description lists.
- A definition list is a list of terms, with a description of each term.
- Definition List makes use of following three tags.

<dl> – Defines the start of the list

<dt> – A term

<dd> – Term definition

</dl> – Defines the end of the list

Example: -

<dl>

<dt>HTTP</dt>

<dd>Hypertext Transfer Protocol</dd>

<dt>URL</dt>

<dd>Uniform Resource Locator</dd>

<dt>WWW</dt>

<dd>World Wide Web</dd>

<dt>DNS</dt>

<dd>Domain Name System</dd>

</dl>

Output: -

HTTP	Hypertext Transfer Protocol
URL	Uniform Resource Locator
WWW	World Wide Web
DNS	Domain Name System

Nested List: -

- A Nested list is a list within a list.
- You can also nest one list within another, so you could make an unordered list inside a numbered one.
- **Example:** You have an asked to create a list of two companies with their popular cars according to you. so you have to create a list of that company and a sub list to list the company's popular cars. Then you have to use the concept of nested list where two list tags are mostly used that is ordered list tag and unordered list tag.

```
<h3>list of cars</h3>
```

```
<ol>
```

```
    <li>maruti suzuki</li>
```

```
    <ul>
```

```
        <li>Dzire</li>
```

```
        <li>Alto</li>
```

```
        <li>Eco</li>
```

```
    </ul>
```

```
    <li>Toyota</li>
```

```
    <ul>
```

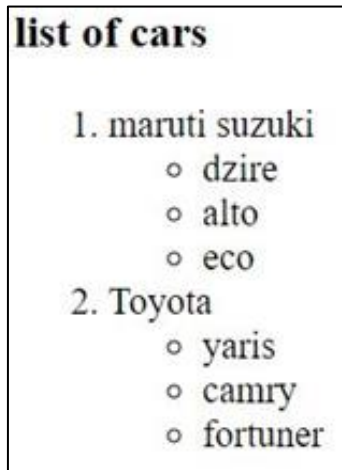
```
        <li>Yaris</li>
```

```
        <li>Camry</li>
```

```
        <li>Fortuner</li>
```

```
    </ul>
```

```
</ol>
```

Output: -

2.2.7. HTML Color & bgcolor Attribute(Not supported in HTML5)

- There are 2 different methods in HTML to set color and bgcolor.

2.2.8. Using color name

```
<body bgcolor="red">
```

```
<font color="red">
```

2.2.9. Using Hexadecimal value

```
<body bgcolor="#ffff00">
```

```
<font color="#ffff00">
```

2.2.10. HTML Font(Not supported in HTML5)

- The tag is used to add style, size, and color to the text on your site. Use the size, color, and face attributes to customize your fonts.
- Use a <basefont> tag to set all of your text to the same size, face, and color.

2.2.11. Font size

- We can change the size of text with the tag using the size attribute.
- The default font size is 3, and the largest font size that can be displayed in a browser is 7.

Example: -

```
<p><font size="5">Here is a size 5 font</font></p>
```

2.2.12. Font Color

- The Color is an attribute of tag, which specifies the text color.

Example:-

```
hex_number:-<font color="#990000">This text is hexcolor #990000</font><br />
```

```
color_name:-<font color="red">This text is red</font>
```

2.2.13. Font Face

- It specifies the font name of the text inside a Font tag.

Example:-

```
<font face="Arial"> Arial Font </font>
```

- When you specify a Font Face, the typeface you specify must be installed on the computer.
- The value of the face attribute can hold several font names separated by a comma.

Example:-

```
<font face="Sans serif","Comic Sans MS","Lucida Console">Sans serif Font</font>
```

- When your page is loaded, browser will display the first font face that it has available, otherwise second one and so on.
- If the specified fonts are not installed on your computer, then it will load default font face of web browser.
- Always use double quotes around the font names.

2.3. <basefont>(Not supported in HTML5)

- HTML <basefont> tag was used to specify the default value of font-size, color, and font-family for all content written within an HTML document.
- In HTML the closing tag </basefont> is not required

Example: -

```
<basefont size="2" color="green" face="arial">  
<p>This paragraph has had its font.</p>  
</basefont>
```

2.4. Anchor Tag

- HTML link is a text we can click on and jump to another document.
- Links allows user to click and move from one web page to another web page.
- Use the <a> tags to define the start and ending of an anchor.
- The text you place between the opening and closing tags will be shown as the link on a page.
- An Anchor tag have 3 important attributes:

2.4.1. href attribute

- The href attribute (hypertext reference) defines the target address of the document.
- Link to an absolute URL:
Example:- Darshan .
- Link to a relative URL:
Example:- Home .
- Link to a section within a URL:
Example:- Reference Section. .

- By default, links will appear as follows in all browsers:

An unvisited link is underlined and blue

A visited link is underlined and purple

An active link is underlined and red

2.4.2. Target attribute

- The target attribute defines whether to open the page in a separate window, or to open the link in the current browser window.

Table 3.9.2 Target Attribute

Attribute with Value	Description
target="_blank"	Opens the linked document in a new window or tab
target="_self"	Opens the linked document in the same frame as it was clicked (this is default)
target="_parent"	Opens the linked document in the parent frame
target="_top"	Opens the linked document in the full body of the window.
target="frameName"	Opens the linked document in the named iframe

2.4.3. name attribute

- the name attribute of the anchor tag can be used to enable users to “jump” to a specific point on a page.

2.5. HTML Images

- Use the tag to place an image on your web page.

2.5.1. Image src

- Above we have defined the src attribute.
- Src stands for source, the source of the image or more appropriately, where the picture file is located.

- There are two ways to define the source of an image. First you may use a standard URL. (src=http://www.Xyz.com/pics/htmlT/sunset.gif) As your second choice, you may copy or upload the file onto your web server and access it locally using standard directory tree methods. (src="../sunset.gif")
- The location of this picture file is in relation to your location of your .html file.

Table 3.10.1 Image src url type

Local SRC	Location Description
src="sunset.gif"	Picture file resides in same directory as .html file
src="../sunset.gif"	Picture file resides in parent directory as .html file
src="../pics/sunset.gif"	Picture file resides in the pics directory in a parent directory as .html file

- A URL cannot contain drive letters. Therefore, something like src="C://www/web/pics/" will not work. Pictures must be uploaded along with your .html file to your web server.

2.5.2. alt attribute

- The alt attribute specifies alternate text to be displayed if for some reason the browser cannot find the image.
- The value of the attribute can be read by screen readers.
- Syntax :-

2.5.3. Height and width

- This attribute is used to adjust the width and height of an image.

2.5.4. Vertically and Horizontally Align Images:-(Not supported in HTML5)

- Use the align and valign attributes to place images within your body, tables, or sections.

1. align (Horizontal)

1. right 2. left 3. center

2. valign (Vertical)

1. top 2. bottom 3. center

- Below is an example of how to align an image to the right of a paragraph

<p>This is paragraph 1, yes, it is...</p>

```

```

2.5.5. Images as Links

- Images are very useful for links and can be created with the HTML below.

```
<a href="http://www.xyz.com/"></a>
```

2.6. Table

- The HTML tables allow user to arrange data into rows and columns.
- The HTML tables are created using the `<table>` tag.
- The `<tr>` tag is used to create table rows and `<td>` tag is used to create data cells.
- Text (data) is displayed left aligned by default using tag `<td>`.
- We need to use border attribute to put a border across cells.

2.6.1. Table Heading

- Table heading can be defined using `<th>` tag. `<th>` tag is centered and bold by default.

Example: -

```
<html>
<head>      <title>HTML Table</title>
</head>
<body>
    <table border=1>
        <tr>
            <th>SRNO</th>
            <th>Browser Name</th>
            <th>Version</th>
        </tr>
        <tr>
            <td>1</td>
            <td>Google Chrome</td>
            <td>96</td>
        </tr>
        <tr>
            <td>2</td>
            <td>Mozilla Firefox</td>
            <td>95</td>
```



```

        </tr>
        <tr>
            <td>3</td>
            <td>Opera</td>
            <td>80</td>
        </tr>
        <tr>
            <td>4</td>
            <td>Internet Explorer</td>
            <td>15</td>
        </tr>
    </table>
</body>
</html>

```

Output: -

SRNO	Browser Name	Version
1	Google Chrome	96
2	Mozilla Firefox	95
3	Opera	80
4	Internet Explorer	15

2.6.2. Cellpadding and cellspacing attribute

- This two-attribute use to adjust space in our table.
- Cellpadding represents the distance between cell borders and content within the cell.
- Cellspacing is the amount of space in between the individual table cells.

Example: -

```

<h2>HTML Table Cellspacing Attribute</h2>
<table border=1 cellspacing="15">
    <caption>Browser Details</caption>
    <tr>
        <th>SRNO</th>
        <th>Browser Name</th>

```

```
        <th>Version</th>
    </tr>
    <tr>
        <td>1</td>
        <td>Google Chrome</td>
        <td>96</td>
    </tr>
    <tr>
        <td>2</td>
        <td>Mozilla Firefox</td>
        <td>95</td>
    </tr>
    <tr>
        <td>3</td>
        <td>Opera</td>
        <td>80</td>
    </tr>
    <tr>
        <td>4</td>
        <td>Internet Explorer</td>
        <td>15</td>
    </tr>
</table>
```

Output: -

Browser Details		
SRNO	Browser Name	Version
1	Google Chrome	96
2	Mozilla Firefox	95
3	Opera	80
4	Internet Explorer	15

- And now we will change the cellpadding of the table and remove the cellspacing from the previous example.

Example: -

<h2>HTML Table Cellpadding Attribute</h2>

<table border=1 cellpadding="10">

<caption>Browser Details</caption>

<tr>

<th>SRNO</th>

<th>Browser Name</th>

<th>Version</th>

</tr>

<tr>

<td>1</td>

<td>Google Chrome</td>

<td>96</td>

</tr>

<tr>

<td>2</td>

<td>Mozilla Firefox</td>

<td>95</td>

</tr>

<tr>

<td>3</td>

<td>Opera</td>

<td>80</td>

```

</tr>
<tr>
    <td>4</td>
    <td>Internet Explorer</td>
    <td>15</td>
</tr>
</table>

```

Output: -

Browser Details		
SRNO	Browser Name	Version
1	Google Chrome	96
2	Mozilla Firefox	95
3	Opera	80
4	Internet Explorer	15

2.6.3. Colspan and rowspan attribute

- We will use colspan attribute if you want to merge two or more columns into a single column.
- Similar way you will use rowspan if you want to merge two or more rows.

Example:-(rowspan)

<h2>Example of Table Rowspan</h2>

<table border=1 cellpadding="10">

<tr>

 <th rowspan="6">Browser Info</th>

</tr>

<tr>

 <th>SRNO</th>

 <th>Browser Name</th>

 <th>Version</th>

</tr>

<tr>

 <td>1</td>

```

        <td>Google Chrome</td>
        <td>96</td>
    </tr>
    <tr>
        <td>2</td>
        <td>Mozilla Firefox</td>
        <td>95</td>
    </tr>
    <tr>
        <td>3</td>
        <td>Opera</td>
        <td>80</td>
    </tr>
    <tr>
        <td>4</td>
        <td>Internet Explorer</td>
        <td>15</td>
    </tr>
</table>

```

Output: -

Example of Table Rowspan			
Browser Info	SRNO	Browser Name	Version
	1	Google Chrome	96
	2	Mozilla Firefox	95
	3	Opera	80
	4	Internet Explorer	15

Example:-(colspan)

```
<h2>Example of Table Colspan</h2>
```

```
<table border=1 cellpadding="10">
```

```
  <tr>
```

```
    <th colspan="3">Browser Info</th>
```

```
  </tr>
```

```
  <tr>
```

```
    <th>SRNO</th>
```

```
    <th>Browser Name</th>
```

```
    <th>Version</th>
```

```
  </tr>
```

```
  <tr>
```

```
    <td>1</td>
```

```
    <td>Google Chrome</td>
```

```
    <td>96</td>
```

```
  </tr>
```

```
  <tr>
```

```
    <td>2</td>
```

```
    <td>Mozilla Firefox</td>
```

```
    <td>95</td>
```

```
  </tr>
```

```
  <tr>
```

```
    <td>3</td>
```

```
    <td>Opera</td>
```

```
    <td>80</td>
```

```
  </tr>
```

```
  <tr>
```

```
    <td>4</td>
```

```
    <td>Internet Explorer</td>
```

```
    <td>15</td>
```

```
  </tr>
```

```
</table>
```

Output: -

Browser Info		
SRNO	Browser Name	Version
1	Google Chrome	96
2	Mozilla Firefox	95
3	Opera	80
4	Internet Explorer	15

2.6.4. Table Height and width

- We can specify table width or height in terms of pixels or in terms of percentage of available screen area.

2.6.5. Table Caption

- It is used to give title for table.
- It shows up at top of the table.

2.6.6. Table background

- We can set table background using one of the following two ways -
bgcolor attribute: -You can set background color for whole table or just for one cell.
background attribute: -You can set background image for whole table or just for one cell.
bordercolor: -You can also set border color also using bordercolor attribute.

2.7. HTML Forms

- A form will take input from the viewer and depending on your needs, you may store that data into a file or in database, place an order, gather user statistics, register the person to your web forum, or maybe subscribe them to your weekly newsletter.

Example: - Gmail Registration Form, Facebook Signup Form

- `<form>` is main tag to build a form. The `<form>` element is a container for different types of input elements, such as: text fields, checkboxes, radio buttons, submit buttons, etc.
- Following are attributes of `<form>`:
- Name: -The name attribute specifies the name of a form which is used to reference elements in a JavaScript.
- Action: -The required action attribute specifies where to send the form-data when a form is submitted.
`<form action="URL">`
- Value:-URL
Description:-Where to send the form data
- Method: -There are two types of method which is used when submitting the form data.
 1. GET
 2. POST
- GET: -The default method when submitting form data is GET. However, when GET is used, the submitted form data will be visible in the page address field. Only 2048 characters are submitted at a time when using GET method.
- POST: -The POST method does not display the submitted form data in the page address field. Unlimited characters send at a time.
- Target: - The target attribute specifies a name or a keyword that indicates where to display the response that is received after submitting the form.
`<form target="_blank|_self|_parent|_top|frameName">`

Table 3.11 HTML Form target attribute

Value	Description
<code>_blank</code>	The response is displayed in a new window or tab
<code>_self</code>	The response is displayed in the same frame (this is default)
<code>_parent</code>	The response is displayed in the parent frame
<code>_top</code>	The response is displayed in the full body of the window
FrameName	The response is displayed in a named frame

2.7.1. Types of Input

- The main tag for collecting information from the user is `<input>`.
- There are quite few different types of input to choose from:
- `<input type="text"/>` this is the default input type and accepts characters and numbers into a text box. It can also have a value attribute attached to it, which will give it an initial value.

Example: -

```
<label for="fname">First name: </label>
<input type="text" id="fname" name="fname">
```

- `<input type="password"/>` this is similar to the above text box but anything that is typed cannot be seen; instead an asterisk is printed to cover up the entry. As the name suggests, this is used for password entry.

Example: -

```
<label for="password">Password:</label>
<input type="password" id="password" name="pwd">
```

- `<input type="checkbox"/>` is used to define square boxes. It is a form element which allows user to select one more option from given options.

Example: -

```
<form method="post">
Select Your Hobbies <input type="checkbox" name="playing" value="playing">Playing
<input type="checkbox" name="reading" value="reading">Reading
<input type="checkbox" name="sleeping" value="sleeping">Sleeping
<input type="checkbox" name="dance" value="dance">Dance
</form>
```

- `<input type="radio"/>` Radio buttons are popular form of interaction. this is similar to checkbox but in group of radio buttons only one can be selected at a time. You may have seen them on quizzes, Questionnaires, Gender etc. For creating group in radio button, we need specify same value of name attribute in all radio buttons.

Example: -

```
<form method="post">
Gender <input type="radio" name="gender" value="male">Male
<input type="radio" name="gender" value="female">Female
</form>
```

- `<input type="file"/>` This will give a box to allow you to choose a file similar to when you open or save files usually on your machine. It can be used to select a file on the local machine for upload to a server.

Example: -

Select Your Photo: `<input type="file" name="image">`

- `<input type="submit"/>` this allows a form to be submitted. When pressed, the information will be passed on for processing, usually to a script mentioned in the action attribute option of the form.

Example: -

`<input type="submit" name="submit" value="click here">`

- `<input type="button"/>` this makes a button available.

Example: -

`<input type="button" id="btn" value="click here">`

- `<input type="reset"/>` this will reset the form to its initial state when selected.

Example: -

`<input type="reset" value="cancel">`

- `<input type="hidden"/>` this allows hidden data(not seen by the user) to be passed along with the form.

Example: -

`<input type="hidden" id="custId" name="custId" value="3487">`

2.7.2. Select Tag

- HTML `<select>` tag is used to create drop down list of options, which appears when the user clicks on form element, and it allows to choose one of the options.
- The `<option>` tag is used to define the possible options to choose from. The tag is put into the `<select>` tag.
- The first option from the list of options is selected by default. To change a predefined option, the `selected` attribute is used.

Example: -

`<form method="post">`

College Degree

```
<select name="degree">
<option>BTech</option>
<option>MCA </option>
<option>MBA</option>
</select>
</form>
```

2.7.3. Select Tag with optgroup tags

- The <optgroup> tag is used to group several options into one group.
- Using <optgroup> tag with <select> makes easier to access the dropdown list
- Especially if list has large number of options.
- The content of <optgroup> looks like heading in bold.
- The width of the list depends on the length of the text inside <option>.

Example: -

```
<form>
  <select id="tutorial_choice">
    <optgroup label="Web">
      <option value="html">HTML</option>
      <option value="css">CSS</option>
    </optgroup>
    <optgroup label="Database">
      <option value="sql">SQL</option>
      <option value="oracle">Oracle</option>
    </optgroup>
  </select>
</form>
```

2.7.4. Textarea

- The HTML <textarea> tag is used to define a multi-line text input control.
- It can hold unlimited number of characters and the texts are displayed in a fixed-width font (usually courier).
- The size of the HTML textarea is defined by <cols> and <rows> attribute, or it can also be defined through CSS height and width properties.
- It helps viewers to place their own comments onto forums.

Example: -

```
<form method="post">
<textarea rows="5" cols="20" name="comments" placeholder="Enter Comments
Here"></textarea>
</form>
```

2.7.5. <label>

- It is used to specify a label for an <input> element of a form. It adds a label to a form control such as text, email, password, textarea etc.

Syntax: - <label> form_content... </label>

- Set the id attribute inside the <input> element and specify its name for the for attribute inside the <label> tag.

Example: -

```
<label for="firstname">Firstname</label>
<input type="text" id="firstname" name="fname" placeholder="Firstname" size="15" required />
```

2.8. <!--comment-->

- Comments are not displayed in the browsers.
- We can use comments to explain our code.
- A comment is a way for you as the web page developer to control what lines of code are to be ignored by the web browser.
- Comment syntax may be a little complicated, there is an opening and a closing much like tags.

<!-- Opening Comment

--> Closing Comment

Example: -

```
<!--This is my banner image -->

```

2.9. Other Tags

2.9.1. span tag

- HTML tag is used as a generic container of inline elements.
- The tag is used to group inline-elements in a document, to change the color, font, background of a part of text using CSS.
- The tag does not have any default meaning.

Example:-

```
<p>My mother has <span style="color:lightblue">lightblue</span>eyes.</p>
```

- HTML is much similar as <div> tag, but <div> is used for block-level elements and tag is used for inline elements.

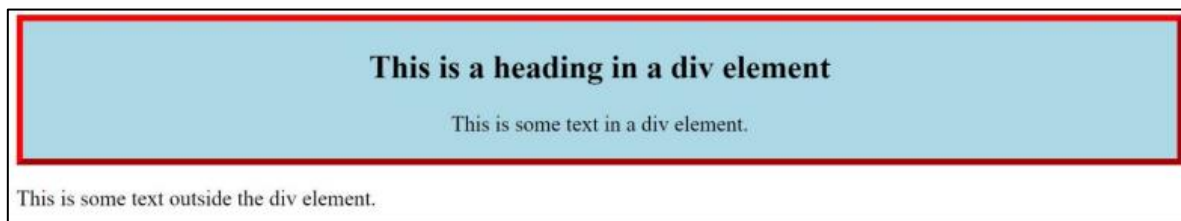
2.9.2. div tag

- The div tag is known as Division tag.
- The div tag is used in HTML to make divisions of content in the web page like (text, images, header, footer, navigation bar, etc).
- Div tag has both open(<div>) and closing (</div>) tag and it is mandatory to close the tag.
- The Div is the most usable tag in web development because it helps us to separate out data in the web page and we can create a particular section for particular data or function in the web pages.
- Div tag is Block level tag. It is a generic container tag.

Example:-

```
<!DOCTYPE html>
<html>
<head>
    <style>
        .myDiv
        {
            border: 5px outset red;
            background-color: lightblue;
            text-align: center;
        }
    </style>
</head>
<body>
    <div class="myDiv">
        <h2>This is a heading in a div element</h2>
        <p>This is some text in a div element.</p>
    </div>
    <p>This is some text outside the div element.</p>
</body>
</html>
```

Output: -



2.10. Meta Tag

- HTML <meta> tag is used to represent the metadata (information about data) about the HTML document.
- It specifies page description, keywords, copyright, language, author of the documents, etc.
- Google search for this meta tag in order to understand the information provided by the website.
- The metadata does not display on the webpage but it is used by browsers and other web services which scan the site or webpage to know about the webpage.
- With the help of meta tag, you can preview that how your webpage will render on the browser.
- The <meta> tag is placed within the <head> tag, and it can be used more than one times in a document.

Table 3.14 Meta Tag

Attribute	Value	Description
Charset	character_set	Specifies the character encoding for the HTML document
Name	application-name author description generator keywords viewport	Specifies a name for the metadata
http-equiv	content-security-policy content-type default-style refresh	Provides an HTTP header for the information/value of the content attribute
Content	Text	Gives the value associated with the http-equiv or name attribute

Scheme	Format/URI USA/Europe	Not supported in HTML5.Specifies a scheme to be used to interpret the value of the content attribute.
--------	--------------------------	---

2.11. Multimedia Tags

2.11.1. Audio Tag

- The <audio> tag is used to embed sound content in a document, such as music or other audio streams.
- The controls attribute adds audio controls, like play, pause, and volume.
- The <audio> tag contains one or more <source> tags with different audio sources. The browser will choose the first source it supports.
- The text between the <audio> and </audio> tags will only be displayed in browsers that do not support the <audio> element.
- There are three supported audio formats in HTML: MP3, WAV, and OGG.
- Before HTML5, audio files could only be played in a browser with a plug-in (like flash).

Table 3.15.1 Multimedia Tag

Attribute	Value	Description
autoplay	autoplay	Specifies that the audio will start playing as soon as it is ready
controls	controls	Specifies that audio controls should be displayed (such as a play/pause button etc)
loop	loop	Specifies that the audio will start over again, every time it is finished
muted	muted	Specifies that the audio output should be muted

Example: -

```
<audio controls>
```

```
  <source src="darshanUniversity.mp3" type="audio/mpeg">
```

Your browser does not support the html audio tag.

```
</audio>
```

2.11.2. Video Tag

- The <video> tag is used to embed video content in a document, such as a movie clip or other video streams.
- The controls attribute adds video controls, like play, pause, and volume.
- It is a good idea to always include width and height attributes.
- The <source> element allows you to specify video files which the browser may choose from.
- The browser will use the first recognized format.
- The text between the <video> and </video> tags will only be displayed in browsers that do not support the <video> element.
- To start a video automatically use the autoplay attribute.
- In HTML5, there are 3 supported video formats: MP4, WebM, and Ogg.
- Before HTML5, a video could only be played in a browser with a plug-in (like flash).

Table 3.15.2 Video Tag

Attribute	Value	Description
autoplay	autoplay	Specifies that the video will start playing as soon as it is ready
controls	controls	Specifies that video controls should be displayed (such as a play/pause button etc).
height	pixels	Sets the height of the video player
width	Pixels	Sets the width of the video player
loop	Loop	Specifies that the video will start over again, every time it is finished
muted	Muted	Specifies that the audio output of the video should be muted
poster	URL	Specifies an image to be shown while the video is downloading, or until the user hits the play button

Example: -


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

2.12. HTML5

2.12.1. Introduction to HTML5

- It stands for Hypertext markup language version 5.
- HTML5 is the latest version of HTML.
- HTML5 is cooperation between the World Wide Web Consortium (W3C) and the Web Hypertext Application Technology Working Group (WHATWG).

2.12.2. What is new in HTML5

- Support multimedia without flash player.
- So, we can include audio, video in our web page without installing flash player.
- We create drawing in our webpage using canvas without graphics software.
- We can trace user's location.
- HTML5 coding structure is user friendly
- HTML5 program is run in latest version of Google chrome, Mozilla Firefox, Opera, Internet explorer 9.0

Structure of HTML5: -

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Page title will go here</title>
</head>
<body>
  This is test page
</body>
</html>
```

DOCTYPE declaration: -

- It is an instruction to the web browser about what version of HTML the page is written in.

Syntax:-<!DOCTYPE html>

- This declaration is not case sensitive. So, we can write it in capital or small letter.

2.13. Elements of HTML5

- A semantic element clearly describes its meaning to both the browser and the developer.
- Semantic elements = elements with a meaning.
- Examples of non-semantic elements: <div> and - Tells nothing about its content.
- Examples of semantic elements: <form>, <table>, and <article> - Clearly defines its content.
- Many semantic elements which is used to develop any webpages.
- Semantic elements which are explained below

<header>

- The <header> element is generally found at the top of a document, a section, or an article.
- It contains the main heading.
- We can have several <header> element in one document.
- It also contains website name, tagline, navigation links, search text box.

<section>

- The <section> element is defined as a block of related elements.
- <section> is a grouping of content.
- A home page could normally be split into sections for introduction, content, and contact information.

<article>

- This tag represents an independent content.
- The content within the <article> tag is independent from the other content of the site.
Ex:-Newspaper news, comment.

<figure>

- It is used to add photos in a document.
- Whenever a programmer calls an external image file inside the FIGURE Element, it automatically adjusts the paragraph text and figure alignment.
- Here is the Syntax for FIGURE Element

<figure>

</figure>

<footer>

- It can contain information about the author, copyright information and contact information.

- We may have several <footer> elements in one document.

<nav>

- The <nav> elements define the set of navigation links.
- Websites typically have sections dedicated to navigational links, which enables user to navigate the site.
- These links can be placed inside a nav tag.

<dialog>

- It is used to create popup dialog on a web page.
- This tag accepts single attribute open which is used to specify the dialog element is active.

<aside>

- The <aside> tag is used to describe the main object of the web page in a shorter way like a highlighter.
- The <aside> tag contains mainly author information, links, and content.
- The <aside> content could be placed as a sidebar in an article.

2.14. Other HTML5 Tags

2.14.1. <fieldset>


- The HTML <fieldset> tag is used for grouping related form elements.
- The use of this tag is optional while creating an HTML form but using <fieldset>, it is easy to understand the purpose of grouped elements of form.
- The <legend> tag is used with the <fieldset> element as a first child to define the caption for the grouped related fields.

Example: -

```
<form>
  <fieldset>
    <legend>User basic information:</legend>
    <label>First Name</label><br>
    <input type="text" name="fname"><br>
    <label>Last Name</label><br>
    <input type="text" name="lname"><br>
    <label>Enter Email</label><br>
    <input type="email" name="email"><br><br>
  </fieldset>
```

</form>

Output: -



2.14.2. <legend>

- HTML <legend> tag is used to insert a title or caption to its parent element such as <fieldset>.
- The <legend> element must be the first child of <fieldset> element.
- By using <legend> tag with <form> elements, it is easy to understand the purpose of grouped form elements.

2.15. HTML5 Attribute

- There are two types of attributes used in HTML5
 1. Input attribute
 2. Form attribute

2.15.1. Input Attribute

autofocus

- The autofocus attribute specifies that the input field should automatically get focus when the page loads.

Example: -

```
<input type="text" id="fname" name="fname" autofocus>
```

Formaction

- The formaction attribute is used to send the data through the URL, as we do it with action attribute of <form> element.
- The formaction attribute overrides the action attribute of the form element.
- The formaction attribute is used with the following type:

Type="submit"

Example: -

```
<form action="autofocus.html">
```

```
    First name: <input type="text" name="fname"><br>
```

```
    Last name: <input type="text" name="lname"><br>
```

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

```
<input type="submit" formaction="novalidate.html" value="Submit to another page">
```

```
</form>
```

Formtarget

- The formtarget attribute specifies a name or a keyword that indicates where to display the response that is received after submitting the form.
- The formtarget attribute overrides the target attribute of the <form> element.
- The formtarget attribute can be used with type="submit" and type="image".

Example: -

```
<form action="welcome.html">
```

```
    First name: <input type="text" name="fname"><br>
```

```
    Last name: <input type="text" name="lname"><br>
```

```
    <input type="submit" value="Submit as normal">
```

```
    <input type="submit" formtarget="_blank" value="Submit to a new window/tab">
```

```
</form>
```

List

- List attribute refer datalist element.
- Datalist contains predefined option for an input element.

```
<input list="datalist_id">
```

What does <input list=""> do?

- Specifies the id of a <datalist> element which provides a list of autocomplete suggestions for the input field.

Example: -

```
<form action="darshanUni.html" method="get">
```

```
<input list="browsers" name="web">
```

```
<datalist id="browsers">
```

```
    <option value="Internet Explorer">
```

```
    <option value="Firefox">
```

```
    <option value="Chrome">
```

```
    <option value="Opera">
```

```
    <option value="Safari">
```

```
</datalist>
```

```
<input type="submit">
```

```
</form>
```

Min and max

- The min and max attributes specify the minimum and maximum values for an <input> element.
- The min and max attributes work with the following input types: number, date, datetime-local, month, time and week.

Example: -

```
<form action="darshanUni.html">
    <label for="datemax">Enter a date before 1980-01-01:</label>
    <input type="date" name="datemax" max="1979-12-31"><br><br>
    <label for="datemin">Enter a date after 2000-01-01:</label>
    <input type="date" name="datemin" min="2000-01-02">
    <input type="submit" value="Submit">
</form>
```

Pattern

- The pattern attribute is used for checking the exact match against predefined regular expression.
- The following input type is support by the pattern attribute:

Type="text"

Type="search"

Type="url"

Type="tel"

Type="email"

Type="password"

Example: -

```
<form action="/action_page.php">
    <label for="country_code">Country code:</label>
    <input type="text" name="country_code" pattern="[A-Za-z]{3}" title="Three letter country
code"><br><br>
    <input type="submit" value="Submit">
</form>
```

placeholder

- The placeholder attribute is used for a short hint for specific input field.
- The following input type is support by the placeholder attributes:

Type="text"

Type="search"

Type="url"

Type="tel"

Type="email"

Type="password"

Example: -

```
<input type="text" name="fname" placeholder="Enter Your First name"><br>
```

```
<input type="text" name="lname" placeholder="Enter Your Last name">
```

Required

- The required attribute is used for a required input field. Without filling the input field user can't submit the form.

Example:-

```
<input type="text" id="username" name="username" required>
```

- The following input type is support by the required attributes:

Type="text"

Type="search"

Type="url"

Type="tel"

Type="email"

Type="password"

Type="date"

Type="number"

Type="checkbox"

Type="radio"

Type="file"

Step

- The step attribute is used for mathematical term matching case.
- We put the value 4, it will accept (-8,-4, 0, 4, 8, 12).
- Step attribute can be used together with the max and min attributes.
- The following input type is support by the required attributes:

Type="number"

Type="range"

Type="date"

Type="datetime"

Type="datetime-local"

Example: -

```
<input type="number" name="points" step="3">
```

Multiple

- It allows user to enter more than one value.
- The multiple attributes work with the following input types: email, and file.
- It is also worked with <select> element.

Example:

```
<input type="file" id="files" name="files" multiple>
```

autocomplete

- The input autocomplete attribute specifies whether a form or an input field should have autocomplete on or off.
- Autocomplete allows the browser to predict the value. When a user starts to type in a field, the browser should display options to fill in the field, based on earlier typed values.
- The autocomplete attribute works with <form> and the following <input> types: text, search, url, tel, email, password, range and color.

Example: -

```
<form autocomplete="on">
    <label for="fname">First name:</label>
    <input type="text" name="fname" autocomplete="off"><br>
    <label for="lname">Last name:</label>
    <input type="text" name="lname"><br>
    <input type="submit" value="Submit">
</form>
```

2.15.2. Form Attribute

- The Form Attributes gives some extra control on Form element.

Autocomplete

- If auto completion is on, it will auto complete the form and if auto completion is off, the user has to fill the form field manual.
- It is possible to have auto complete “on” and “off” for the form, and “off” and “on” for specific input fields.

The auto complete attribute works with <form> and the following <input> types:

Text	Search
url	url
tel	password
date	color

Example: -

```
<form autocomplete="on">
    First name:<input type="text" name="fname"><br>
    Last name: <input type="text" name="lname"><br>
    E-mail: <input type="email" name="email"><br>
    <input type="submit">
</form>
```


novalidate

- Where novalidate attribute is used to send the information for not validating the form field. It specifies that form data shouldn't be validated.

Example: -

```
<form action="welcome.html" novalidate>
<input type="email" name="email"><br><br>
  <input type="submit">
</form>
```

2.16. HTML5 Input Types

- HTML5 input elements introduced several new values for the type attribute.

datetime

- The HTML `<input type="datetime">` gives a control for entering a date and time (hour, minute, second, and fraction of a second) as well as a timezone.
- This feature is out of date.
- This feature has been removed from WHATWG HTML, and is no longer supported in browsers.

Example: -

```
<form>
<label for="meetingtime">Meeting Time:</label>
<input type="datetime" id="meetingtime" name="meetingtime">
<input type="submit">
</form>
```

datetime-local

- The `<input>` tag with a `type="datetime-local"` attribute creates an input field that accepts a date and time value which does not include the timezone.
- Clicking the calendar icon opens a datetime picker.
- The `-local` suffix does not mean the time is local to the user, instead it is a time without a timezone.
- The UI of this control varies from browser to browser. Browser support is limited.

A similarly named `<input type="datetime">` is obsolete.

Example: -

```
<form>
<label for="birthdaytime">Birthday (date and time):</label>
<input type="datetime-local" id="birthdaytime" name="birthdaytime">
</form>
```

date

- The date is the value of the type attribute of an <input> element. It creates a calendar that allows a user to choose the date.
- The resulting value includes the day, month, and year.

Example: -

```
<form>
  <label for="birthday">Birthday:</label>
  <input type="date" name="birthday">
</form>
```

- You can also use the min and max attributes to add restrictions to dates.

Month

- The <input> tag with a type="month" attribute creates a text field that accepts a month and year value.
- Inside this field is a calendar icon. Clicking the calendar icon opens a month/year picker.

Example: -

```
<form>
  <input type="month" name="birthmonth"><br /><br />
  <input type="submit" value="Submit">
</form>
```

Week

- The <input type="week"> defines a week and year control.

Example: -

```
<form>
  <label for="week">Select a week:</label>
  <input type="week" id="week" name="week">
  <input type="submit">
</form>
```

Time

- The <input type="time"> defines a control for entering a time.

Example: -

```
<form>
  <label for="appt">Select a time:</label>
```

```
<input type="time" id="appt" name="appt">
<input type="submit">
</form>
```

number

- This accepts only numerical value.
- The step attribute specifies the accuracy, defaulting to 1.

Example: -

```
<form>
  <label for="quantity">Quantity (between 1 and 5):</label>
  <input type="number" name="quantity" min="1" max="5">
</form>
```

range

- The `<input type="range">` defines a control for entering a number whose exact value is not important (like a slider control). Default range is 0 to 100.
- However, you can set restrictions on what numbers are accepted with the min, max, and step attributes:

Example: -

```
<form>
  <label for="vol">Volume (between 0 and 50):</label>
  <input type="range" name="vol" min="0" max="50">
</form>
```

Email

- This accepts only email value.
- This type is used for input fields that should contain an e-mail address.
- If you try to submit a simple text, it forces to enter only email address in email@example.com format

Example: -

```
<form>
  <label>Enter your Email-address</label>
  <input type="email" name="email" required>
  <input type="submit">
</form>
```

url

- This accepts only URL value.
- This type is used for input fields that should contain a URL address. if you try to submit a simple text, it forces to enter only URL address.

Example: -

```
<form>
    <label>Enter your website URL: </label>
    <input type="url" name="website" placeholder="http://www.darshan.ac.in"><br>
    <input type="submit" value="send data">
</form>
```

color

- The `<input type="color">` is used for input fields that should contain a color.
- Depending on browser support, a color picker can show up in the input field.

Example: -

```
<form>
    <label for="favcolor">Select your favorite color:</label>
    <input type="color" id="favcolor" name="favcolor">
</form>
```

tel

- The `<input type="tel">` is used for input fields that should contain a telephone number.

Example: -

```
<form>
    <label for="phone">Enter your phone number:</label>
    <input type="tel" name="phone" pattern="[0-9]{3}-[0-9]{2}-[0-9]{3}">
</form>
```

Search

- The `<input type="search">` creates an input field which allows a user to enter a search string.
- These are functionally symmetrical to the text input type, but may be styled differently.

Example: -

```
<form>
    <label>Search here:</label>
    <input type="search" name="q">
    <input type="submit" value="search">
</form>
```

2.17. HTML5 Form Validation

- Form validation is a “technical process where a web-form checks if the information provided by a user is correct.”
- The form will either alert the user that something is not in correct format and need to fix to proceed, or the form will be validated and the user will be able to continue with their process.
- Form can be validated both in Client-Side as well as Server-Side, it is recommended to validate the form in both the side.
- Form validation generally performs two functions.
 - Basic Validation
 - Emptiness
 - Length Validation etc.....
 - Data Format Validation: -Secondly, the data that is entered must be checked for correct form and value.
 - Email Validation
 - Mobile Number Validation
 - Enrollment Number Validation etc....
- We can use required attribute in order to stop user sending empty data to server.
`<input type="text" name="txtName" required/>`
- We can use pattern attribute in order to force some format on user before sending the data to server.
`<input type="text" name="txtName" pattern="[0-9]{10}" />`
- We can use title attribute for custom error message.
`<input type="text" name="txtName" pattern="[0-9]{10}" title="Please enter 10-digit mobile number" required/>`

Unit-3

CSS

3.1. Introduction to CSS

- CSS stands for Cascading Style Sheets.
- It is a simple design language intended to simplify the process of making web pages presentable.
- CSS defines layout of HTML documents. For example, CSS covers Fonts, colors, margins, lines, height, width, background images, advanced positions and many other things.
- CSS defines How HTML elements are to be displayed.
- Styles are normally saved in external .css files.

3.1.1. Why CSS

- HTML was not containing tags for formatting a web page!
- HTML was created to describe the content of a web page, like:
`<h1>this is a heading</h1>`
`<p>this is a paragraph. </p>`
- Development of large websites, where fonts and color information were added to every single page, became a long and expensive process.
- To solve this problem, the World Wide Web Consortium (W3C) created CSS.

3.1.2. Advantages

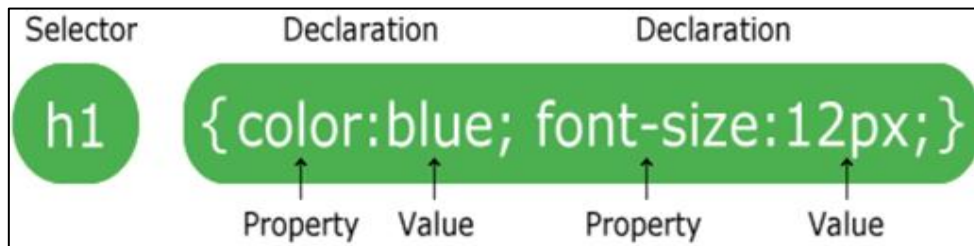
- Improves Website Presentation
- External CSS makes Updates Easier and Smoother
- External CSS helps Web Pages Load Faster

3.1.3. Disadvantages

- Browser Dependent
- Difficult to retrofit in old websites

3.2. Basic Syntax and Structure of CSS

- A CSS rule-set consists of a selector and a declaration block.
- The selector points to the HTML element you want to style.
- The declaration block contains one or more declarations separated by semicolons.
- Each declaration includes a CSS property name and a value, separated by a colon.
- A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.
- To make the style definitions more readable, you can describe one property on each line.
- **Syntax:** -



Example: -

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

3.3. Types of CSS

- Cascading Style Sheet (CSS) is used to set the style in web pages which contain HTML elements.
- It sets the background color, font-size, font-family, color, etc property of elements in web pages.
- There are three types of CSS which are given below:
 - Inline CSS
 - Internal or Embedded CSS
 - External CSS

3.3.1. Inline CSS

- It is possible to place CSS right in your HTML code, and this method of CSS usage is referred to as inline css.
- Inline CSS is used to style a specific HTML element.
- Inline CSS has the highest priority out of external, internal, and inline CSS.
- This means that you can override styles that are defined in external or internal by using inline CSS.
- If you want to add a style inside an HTML element all you have to do is specify the desired CSS properties with the style HTML attribute.
- Managing a website may difficult if we use only inline CSS. However, Inline CSS in HTML is useful in some situations.

Example:-

```
<!DOCTYPE html>
<html>
<body>
  <h1 style="color:red;">DARSHAN UNIVERSITY-RAJKOT</h1>
  <p style="color:blue;font-size:24px">Rajkot-Morbi Highway,<br>
    Hadala,<br>
    Rajkot-363650,GUJARAT(INDIA)<br>
    Email:-info@darshan.ac.in<br>
    Phone No:-9727747310
  </p>
</body>
</html>
```

Output:-**DARSHAN UNIVERSITY-RAJKOT**

Rajkot-Morbi Highway,
Hadala,
Rajkot-363650,GUJARAT(INDIA)
Email:-info@darshan.ac.in
Phone No:-9727747310

Drawbacks: -

- We have to define the style over and over again instead of defining it only once.
- We want to change a specific style; we have to change it in many places in our document instead of changing it to one.

3.3.2. Internal CSS

- This can be used when a single HTML document must be styled uniquely.
- The CSS rule set should be within the HTML file in the head section i.e the CSS is embedded within the HTML file.
- Using Internal CSS for multiple web pages is time-consuming because we require placing the style on each web page.

Example: -

```
<!DOCTYPE html>
<html>
<head>
<style>
    h1
    {color:red;}
    p
    {
        color:blue;
        font-size:24px;
    }
</style>
</head>
<body>
<h1>DARSHAN UNIVERSITY-RAJKOT</h1>
<p>Rajkot-Morbi Highway,<br>
Hadala,<br>
Rajkot-363650,GUJARAT(INDIA)<br>
Email:-info@darshan.ac.in<br>
Phone No:-9727747310
</p>
</body>
</html>
```


Output:-

DARSHAN UNIVERSITY-RAJKOT

Rajkot-Morbi Highway,
Hadala,
Rajkot-363650,GUJARAT(INDIA)
Email:-info@darshan.ac.in
Phone No:-9727747310

3.3.3. External CSS

- External CSS contains separate CSS file which contains only style property with the help of tag attributes (For example class, id, heading, ... etc).
- CSS property written in a separate file with .css extension and should be linked to the HTML document using link tag. This means that for each element, style can be set only once and that will be applied across web pages.

Example: -

- The file given below contains CSS property. This file saves with .css extension. For Ex: darshanuni.css

```
h1
{
  color: red;
}
p
{
  color: blue;
  font-size:24px;
}
```

- Below is the HTML file that is making use of the created external style sheet.
 - link tag is used to link the external style sheet with the html webpage.
 - href attribute is used to specify the location of the external style sheet file.

```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" href="darshanuni.css"/>
</head>
<body>
<h1>DARSHAN UNIVERSITY-RAJKOT</h1>
  <p>Rajkot-Morbi Highway,<br>
  Hadala,<br>
  Rajkot-363650,GUJARAT(INDIA)<br>
  Email:-info@darshan.ac.in<br>
  Phone No:-9727747310
</p>
```

</body>

</html>

Output: -**DARSHAN UNIVERSITY-RAJKOT**

Rajkot-Morbi Highway,
Hadala,
Rajkot-363650,GUJARAT(INDIA)
Email:-info@darshan.ac.in
Phone No:-9727747310

Note: -

- As Inline has the highest priority, any styles that are defined in the internal and external style sheets are overridden by Inline styles.
- Internal or embedded stands second in the priority list and overrides the styles in the external style sheet.
- External style sheets have the least priority. If there are no styles defined either in inline or internal style sheet then external style sheet rules are applied for the HTML tags.

3.4. Targeting using CSS

3.4.1. Element Selector

- The element selector select HTML elements based on the element name.

Example: -

```
p
{
  color: blue;
  font-size: 20px;
}
<h1>Darshan University-Rajkot</h1>
<p>Rajkot-Morbi Highway,<br>
  Hadala,<br>
  Rajkot-363650,GUJARAT(INDIA)<br>
  Email:-info@darshan.ac.in<br>
  Phone No:-9727747310
</p>
```

Output: -

Darshan University-Rajkot
Rajkot-Morbi Highway,
Hadala,
Rajkot-363650, GUJARAT(INDIA)
Email:-info@darshan.ac.in
Phone No:-9727747310

3.4.2. ID selector

- ID selectors in CSS allow you to target elements (Tags) by their ID values.
- Each element can have only one ID. Each page can have only one element with that ID.
- ID selectors are unique, so you can apply only to the content of one element.
- To reference an ID, you precede the ID name with a hash mark (#).

Example: -

```
#para1
{
    color:blue;
}
<h1 id="para1">Darshan University-Rajkot</h1>
<h1>Hello DU Students</h1>
```

Output: -

Darshan University-Rajkot
Hello DU Students

3.4.3. Class selector

- With class selector we can define same style for different html element.
- The Class selector begins with a dot (.) and followed by a class name which you choose.

Example: -

```
.myClass
{
    color: blue;
}
<h1 class="myClass">Darshan Uni. Courses after 12th science:-B.Tech,B.Sc.</h1>
<h1>Darshan Uni. Courses after 12th commerce:-B.B.A,B.C.A.,B.Com</h1>
<h1 class="myClass">Darshan Uni. PG Courses:-M.C.A,M.Sc,M.B.A,,M.Tech</h1>
```

Output: -

Darshan Uni. Courses after 12th science:-B.Tech,B.Sc.
Darshan Uni. Courses after 12th commerce:-B.B.A,B.C.A.,B.Com
Darshan Uni. PG Courses:-M.C.A,M.Sc,M.B.A,,M.Tech

3.4.4. Assign Multiple Classes

- We can apply different class to same html element by giving space separated class names in the class attribute.

Example: -

```
.class1
{
    color:blue;
}
.class2
{
    text-align:center;
}
<h1 class="class1 class2">Darshan University-Rajkot</h1>
```

Output: -

Darshan University-Rajkot

3.4.5. Multiple Selection

- We can apply same css to multiple selectors using comma separated selector list.

Example: -

```
h1,p
{
    color:blue;
}
<h1>Darshan University-Rajkot</h1>
<p>Rajkot-Morbi Highway,<br>
    Hadala,<br>
    Rajkot-363650,GUJARAT(INDIA)<br>
    Email:-info@darshan.ac.in<br>
    Phone No:-9727747310
</p>
```

Output: -

Darshan University-Rajkot

Rajkot-Morbi Highway,
Hadala,
Rajkot-363650,GUJARAT(INDIA)
Email:-info@darshan.ac.in
Phone No:-9727747310

3.4.6. Multi-level Selection

- We can use hierarchical path to target html element by space separated element/class/id names.

Example: -

```
div h1
{
  color:blue;
}
<h1>Darshan Institute of Engineering & Technology-Rajkot</h1>
<div>
  <h1>Darshan University-Rajkot</h1>
</div>
```

Output: -

Darshan Institute of Engineering & Technology-Rajkot
Darshan University-Rajkot

3.5. Background Properties

- CSS background property is used to define the background effects on element.
- There are 5 CSS background properties that affects the HTML elements.
 - background-color
 - background-image
 - background-repeat
 - background-attachment
 - background-position
- background-color
 - The background-color property sets the background color of an element.
 - The background of an element is the total size of the element, including padding and border (but not the margin).

Example: -

```
body
{
```

```
background-color: coral;
}
```

- **background-image**

- The background-image property is used to set an image as a background of an element.
- By default, a background-image is placed at the top-left corner of an element, and repeated both vertically and horizontally.

Example: -

```
body
{
background-image: url("darshanuni.jpg");
}
```

Note:-The background image should be chosen according to text color. The bad combination of text and background image may be a cause of poor designed and not readable webpage.

- **background-repeat**

- By default, the background-image property repeats the background image horizontally and vertically. Some images are repeated only horizontally or vertically.
- The background looks better if the image repeated horizontally only.

Table 4.5.1 Background Attachment

Value	Description
No-repeat	The background image will display only once.
repeat-x	The background image is repeated only horizontally.
repeat-y	The background image is repeated only vertically.
repeat	The background image will be repeated vertically & horizontally. This is default.

Example: -

```
body
{
background-image: url("darshanuni.jpg");
background-repeat: repeat-x;
}
```

- **background-attachment**

- The background-attachment property is used to specify if the background image is fixed or scroll with the rest of the page in browser window.
- If you set fixed the background image then the image will not move during scrolling in the browser.

Table 4.5.2 Background Attachment

Value	Description
fixed	The background image will not scroll with the page
scroll	The background image will scroll with the page. This is default
local	The background image will scroll with the element's contents

Example: -

```
body
{
  background-image: url("img_tree.gif");
  background-repeat: no-repeat;
  background-attachment: fixed;
}
```

- background-position
 - The background-position property sets the starting position of a background image.
 - By default, the background image is placed on the top-left of the webpage.

Table 4.5.3 Background Position

Value	Description
top left top right top center center left center center center right bottom left bottom right bottom center	If you only specify one keyword, the second value will be "center"
x% y%	The first value will be in horizontal position and the second value is the vertical. If you only specify one value, the other value will be added automatic.
x-pos y-pos	The first value will be in horizontal position and the second value is the vertical. Units can be pixels. can mix % and positions

Example: -

<!DOCTYPE html>

```
<html>
<head>
<style>
  body
  {
    background-image: url("darshan_uni.png");
    background-color: #95e8e0;
    background-repeat:no-repeat;
    background-position:top center;
  }
</style>
</head>
<body>
  <h1>Darshan University UG Courses</h1>
  <ul>
    <li>B.Tech</li>
    <li>BCA</li>
    <li>B.Sc.(Information Technology)</li>
    <li>B.Sc.(Microbiology)</li>
    <li>BBA</li>
    <li>B.Com.</li>
  </ul>
</body>
</html>
```

Output: -



3.6. Text Property

- The CSS text properties allow you to control the appearance of text.
- It is possible to change the color of a text, increase or decrease the space between characters in a text, align a text, decorate a text, indent the first line in a text.
- There are several text properties which are listed below
 1. text-align
 2. text-decoration
 3. text-transform
 4. text-indent
 5. word-spacing

6. letter-spacing
 7. white-space
 8. line-height
- text-align
 - The text-align property is used to set the horizontal alignment of a text.
 - A text can be left or right aligned, centered and justified.

Table 4.6.1 Text-align

Value	Description
Left	Aligns the text to the left
Right	Aligns the text to the right
Center	Center the text
Justify	

- When the text-align property is set to "justify", each line has equal width and left and right margins are same (like in newspaper).

Example: -

```
p
{
    text-align: justify;
}
```

- text-decoration
 - The text-decoration property is used to add "decorations" to inline content.
 - It can add lines above, under, and through the text.

Table 4.6.2 Text-decoration

Value	Description
None	Default value. Specifies no line for the text-decoration
Underline	Specifies that a line will be displayed under the text
Overline	Specifies that a line will be displayed over the text
Line through	Specifies that a line will be displayed over the text

Example: -

```
h1
{
```

```
text-decoration: overline;
}
```

- **text-transform: -**
 - This CSS property allows us to change the case of the text. It is used to control the text capitalization.
 - This CSS property can be used to make the appearance of text in all-lowercase or all-uppercase or can convert the first character of each word to uppercase.

Table 4.6.3 Text-transform

Value	Description
capitalize	Transforms the first character of each word to uppercase
uppercase	Transforms all characters to uppercase
lowercase	Transforms all characters to lowercase
none	No capitalization. The text renders as it is. This is default

Example: -

```
p
{
text-transform: capitalize;
}
```

- **Word-spacing: -**
 - It is a CSS property to increases or decreases the white space between words.
 - Negative values are allowed.

Table 4.6.4 Word Spacing

Value	Description
Normal	Defines normal space between words (0.25em).This is default
length	Defines an additional space between words (in px, pt, cm, em, etc). Negative values are allowed.

Example: -

```
h1
{
word-spacing: 30px;
}
```

- Letter-spacing: -
 - The letter-spacing property increases or decreases the space between characters in a text.

Table 4.6.5 Letter Spacing

Value	Description
Normal	No extra space between characters. This is default
length	Defines an extra space between characters (negative values are allowed).

Example: -

```
h1
{
    letter-spacing: 8px;
}
```

- White-space: -
 - The CSS white space property is used to specify how to display the content within an element. It is used to handle the white spaces inside an element.

Table 4.6.5 White Space

Value	Description
Normal	Sequences of whitespace will collapse into a single whitespace. Text will wrap when necessary. This is default
nowrap	Sequences of whitespace will collapse into a single whitespace. Text will never wrap to the next line. The text continues on the same line until a tag is encountered
pre	Whitespace is preserved by the browser. Text will only wrap on line breaks. Acts like the <pre> tag in HTML
pre-line	Sequences of whitespace will collapse into a single whitespace. Text will wrap when necessary, and on-line breaks
pre-wrap	Whitespace is preserved by the browser. Text will wrap when necessary, and on-line breaks

Example: -

```
p
{
    white-space: pre;
}
```

- Line-height:-
 - The line-height attribute will set the height of the line in the page.

- Negative values are not allowed.

Example: -

```
p
{
    line-height:30px;
}
```

3.7. Font Property

- CSS Font property is used to control the look of texts. By the use of CSS font property, you can change the text size, color, style and more.
- There are some important font properties
 1. color
 2. Font-family
 3. font-size
 4. font-style
 5. font-weight
 6. font-variant
- *color*
 - The color property in CSS is used to set the color of HTML elements.
 - Typically, this property is used to set the font color of an element.

Example: -

```
h1
{
    color: #00ff00;
}
```
- *Font-family*
 - The font-family property should hold several font names.
 - If the browser does not support the first font, it tries the next font, and so on.
 - If the name of a font family is more than one word, it must be in quotation marks, like: "Times New Roman".
 - More than one font family is specified in a comma-separated list:

Example: -

```
h2
{
    font-family: Arial, "Times New Roman", sans-serif;
}
```
- *font-size*
 - The font-size property in CSS is used to set the font size of text in HTML document.
 - Its default value is medium and can be applied to every element.

Table 4.7.1 Font Size

Value	Description
xx-small	

x-small	Sets the size of the font to different sizes, from xx-small to xx-large
Small	
Medium	
Large	
x-large	
xx-large	
Smaller	Sets the font size smaller than parent element
Larger	Sets the font-size to a larger size than the parent element
%	Sets the font-size to a %

Example: -

```
p
{
    font-size: large;
}
```

- **font-style**

- The font-style property is mostly used to specify italic text.
- This property has three values:
 - normal - The text is shown normally
 - italic - The text is shown in italics
 - oblique - oblique is very similar to italic, but less supported.

Example: -

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

- **Font-weight**

- The font-weight property sets how thick or thin characters in text should be displayed.

Table 4.7.1 Font Weight

Value	Description
Normal	Defines normal character
Bold	Defines thick character

Bolder	Defines thicker character
Lighter	Defines lighter character
100 to 900	Defines from thin to thick characters. 400 is the same as normal, and 700 is the same as bold

Example: -

```
p
{
    font-weight: bold;
}
```

- **Font-variant:** -

- The font-variant property is used to converted all lowercase letters into uppercase letters. But the converted upper letters appear too small font-size than the original uppercase letters.

Table 4.7.2 Font Variant

Value	Description
normal	The browser displays a normal font. This is default
small-caps	The browser displays a small-caps font

Example:

```
p
{
    font-variant: small-caps;
}
```

Example: -

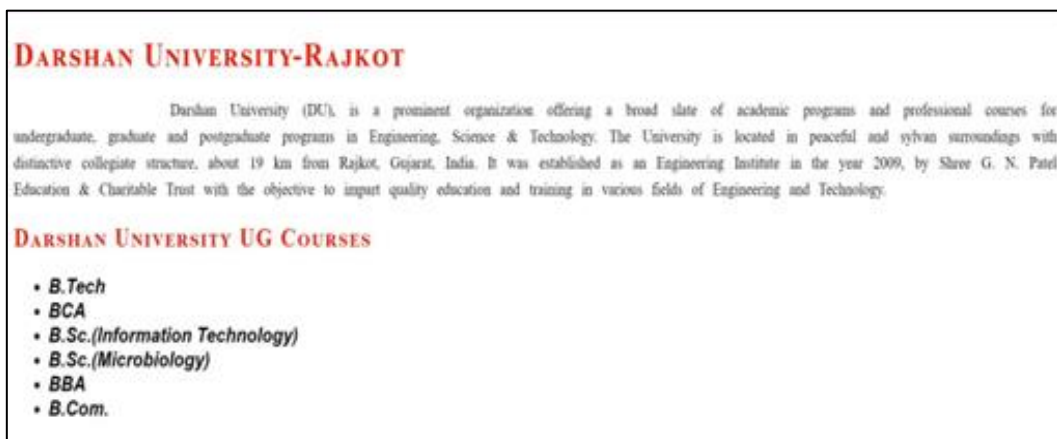
```
<!DOCTYPE html>
<html>
<head>
    <style>
        h1,h2
        {
            font-variant: small-caps;
            color:#ea1f0a;
        }
    </style>
    p
    {
        text-align: justify;
        text-indent:15%;
        word-spacing: 7px;
        line-height: 25px;
    }
</head>
<body>
```

```

    }
    ul li
    {
        font-size:20px;
        font-family:Arial,"Times New Roman",sans-serif;
        font-style: italic;
        font-weight: bold;
    }
</style>
</head>
<body>
    <h1>Darshan University-Rajkot</h1>
    <p>Darshan University (DU), is a prominent organization offering a broad slate of academic
    programs and professional courses for undergraduate, graduate and postgraduate programs in
    Engineering, Science & Technology. The University is located in peaceful and sylvan surroundings
    with distinctive collegiate structure, about 19 km from Rajkot, Gujarat, India. It was established as an
    Engineering Institute in the year 2009, by Shree G. N. Patel Education & Charitable Trust with the
    objective to impart quality education and training in various fields of Engineering and Technology.
    </p>
    <h2>Darshan University UG Courses</h2>
    <ul>
        <li>B.Tech</li>
        <li>BCA</li>
        <li>B.Sc.(Information Technology)</li>
        <li>B.Sc.(Microbiology)</li>
        <li>BBA</li>
        <li>B.Com.</li>
    </ul>
</body>
</html>

```

Output: -



3.8. Box Model

- All HTML elements can be considered as boxes. In CSS, the term "box model" is used when talking about design and layout.
- The CSS box model is essentially a box that wraps around HTML elements, and it consists of: margins, borders, padding, and the actual content.
- The box model allows us to place a border around elements and space elements in relation to other elements.
- The image below illustrates the box model:



Figure 4.8. Sample Diagram

- Explanation of the different parts:
 - Margin - Clears an area around the border. The margin does not have a background color, it is completely transparent
 - Border - A border that goes around the padding and content. The border is affected by the background color of the box
 - Padding - Clears an area around the content. The padding is affected by the background color of the box
 - Content - The content of the box, where text and images appear

Example: -

```
<!DOCTYPE html>
<html>
<head>
<style>
div
{
    background-color: lightgrey;
    width: 300px;
    border: 15px solid green;
    padding: 50px;
    margin: 20px;
}
</style>
</head>
```



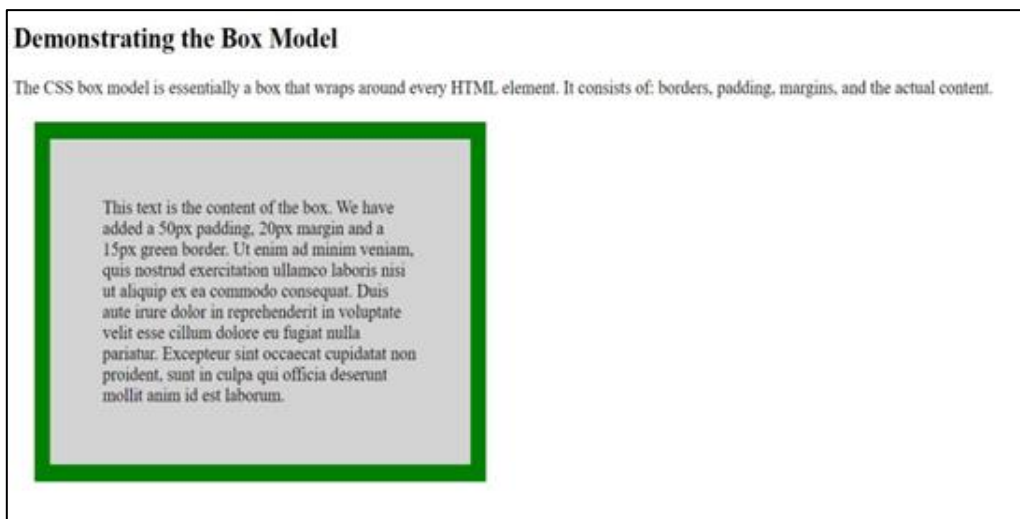
```
<body>
<h2>Demonstrating the Box Model</h2>
<p>The CSS box model is essentially a box that wraps around every HTML element. It consists of:
borders, padding, margins, and the actual content.
</p>
```

```
<div>
```

This text is the content of the box. We have added a 50px padding, 20px margin and a 15px green border. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

```
</div>
</body>
</html>
```

Output: -



3.9. Styling List

- The CSS list properties allow you to:
 - Set different list item markers for ordered lists
 - Set different list item markers for unordered lists
 - Set an image as the list item marker
- *list-style-type*
 - Specify all the list properties in one declaration.
Unordered list styles: square, circle, disc (default), and none
Ordered list styles: upper-alpha, lower-alpha, upper-roman, lower-roman, decimal (default), and none
Example: -


```
ol
{
    list-style-type: upper-roman;
}
```

```
ul
{
    list-style-type: circle;
}
```

- *list-style-image*

- Specify an image as the list-item marker in a list:

Example:

```
ul
{
    list-style-image:url("listArrow.gif");
}
ol
{
    list-style-image:url("listArrow2.gif");
}
```

- *list-style-position*

- With Specify that the list-item markers should appear inside the content flow (results in an extra indentation)

Example:

```
ul
{
    list-style-position:inside;
}
ol
{
    list-style-position:outside;
}
```

Note: "Outside" is actually the default setting for indentation.

Example: -

```
<!DOCTYPE html>
<html>
<head>
<style>
ul.ug
{
    list-style-image:url("darshanlogo.gif");
}
ol.pg
{
    list-style-type: upper-roman;
    list-style-position:inside;
}
</style>
</head>
```

```
<body>
<h2>Styling List</h2>
<h3>Darshan University UG Courses:</h3>
<ul class="ug">
    <li>B.Tech</li>
    <li>B.Sc.</li>
    <li>B.B.A</li>
    <li>B.C.A</li>
</ul>
<h3>Darshan University PG Courses:</h3>
<ol class="pg">
    <li>M.Tech</li>
    <li>M.Sc.</li>
    <li>M.B.A</li>
    <li>M.C.A</li>
</ol>
</body>
</html>
```

Output: -



Figure 4.9 StyleSheet

3.10. Position Property

- The position property in CSS defines where an element should be placed on a page.
- There are four different position values:
 - sticky
 - fixed
 - relative
 - absolute
- Fixed**
 - An element with fixed positioned will stays in the same place even if the page is scrolled.
 - The top, right, bottom, and left properties are used to position the element.

Example:

```
h1
{
    position:fixed;
    top:50px;
    left:100px;
}
```

- **Relative**

- Using relative method, we move element from its normal position.
- If we had a header that appears at the top of our page, we could use relative positioning to move it a bit to the right and down a couple of pixels.

Example:-

```
h3
{
    position:relative;
    top:15px;
    left:150px;
}
```

- **Absolute: -**

- An element with position: absolute is positioned relative to its parent. In the case when there is no positioned parent element, it will be positioned related directly to the HTML element
- The point of origin is the top-left of the browser's viewable area, so be sure you are measuring from that point.
- Element's final position is determined by the values of top, right, bottom, and left.

Example:

```
h1
{
    position:absolute;
    left:50px;
    top:100px;
}
```

- **Sticky: -**

- An element with position: sticky; is positioned based on the user's scroll position.
- A sticky element toggles between relative and fixed, depending on the scroll position.

Example: -

```
h1
{
    position:sticky;
    top:0px;
}
```

Note: -Without having defined the position method you cannot assign the alignment like bottom, top, left and right.

Unit-4

CSS3

4.1. Introduction to CSS3

- CSS3 is the latest standard for CSS.
- CSS3 is completely backwards-compatible with earlier versions of CSS.
- CSS3 has been split into "modules". It contains the "old CSS specification" (which has been split into smaller pieces). In addition, new modules are added.
- CSS3 Transitions are a presentational effect which allow property changes in CSS values, such as those that may be defined to occur on: hover or: focus, to occur smoothly over a specified duration – rather than happening instantaneously as is the normal behaviour.
- Transition effects can be applied to a wide variety of CSS properties, including background-color, width, height, opacity, and many more.
- Some of the most important CSS3 modules are:
 - CSS Animations and Transitions
 - Calculating Values With calc()
 - Advanced Selectors
 - Generated Content and Counters
 - Gradients
 - Web fonts
 - Box Sizing
 - Border Images
 - Media Queries
 - Multiple Backgrounds
 - CSS Columns

4.2. Animations

- CSS animations make it possible to animate transitions from one CSS style configuration to another.
- Animations consist of two components,
 - A style describing the CSS animation
 - A set of keyframes that indicate the start and end states of the animation's style, as well as possible intermediate waypoints.
- There are three key advantages to CSS animations over traditional script-driven animation techniques:
 - They're easy to use for simple animations; you can create them without even having to know JavaScript.
 - The animations run well, even under moderate system load. Simple animations can often perform poorly in JavaScript. The rendering engine can use frame-skipping and other techniques to keep the performance as smooth as possible.
 - Letting the browser control the animation sequence lets the browser optimize performance and efficiency by, for example, reducing the update frequency of animations running in tabs that aren't currently visible.
- To create a CSS animation sequence, you style the element you want to animate with the animation property or its sub-properties. This lets you configure the timing, duration, and other details of how the animation sequence should progress.

- Note: This does not configure the actual appearance of the animation, which is done using the @keyframes
- The sub-properties of the animation property are:
 - **animation-name:** Specifies the name of the @keyframes at-rule describing the animation's keyframes.
 - **animation-duration:** Configures the length of time that an animation should take to complete one cycle.
 - **animation-timing-function:** Configures the timing of the animation; that is, how the animation transitions through keyframes, by establishing acceleration curves. (Linear, ease, ease-in, ease-out etc...)
 - **animation-delay:** Configures the delay between the time the element is loaded and the beginning of the animation sequence.
 - **animation-iteration-count:** Configures the number of times the animation should repeat; you can specify infinite to repeat the animation indefinitely.
 - **animation-direction:** Configures whether or not the animation should alternate direction on each run through the sequence or reset to the start point and repeat itself. (Normal, reverse, alternate etc...)
 - **animation-fill-mode:** Configures what values are applied by the animation before and after it is executing.
 - **animation-play-state:** Lets you pause and resume the animation sequence.

4.2.1. Defining the animation sequence using keyframes

- Once you've configured the animation's timing, you need to define the appearance of the animation. This is done by establishing two or more keyframes using the @keyframes at-rule.
- Each keyframe describes how the animated element should render at a given time during the animation sequence.

Example: -

```
<style>
p
{
  animation-duration: 3s;
  animation-name: slidein;
}
@keyframes slidein
{
  from
  {
    margin-left: 100%;
    width: 300%;
  }
  to
  {
    margin-left: 0%;
    width: 100%;
  }
}
</style>
```

```
<body>
  <p>Hello World</p>
</body>
```

4.2.2. Specifying Intermediate steps

- We can specify intermediate steps using percentage of time in @keyframes, similar to the example given below.

Example: -

```
<style>
p
{
    animation-duration: 4s;
    animation-name: changeColors;
}
@keyframes changeColors
{
    0% {background-color: yellow;}
    25% {background-color: blue;}
    50% {background-color: green;}
    100% {background-color: red;}
}
</style>
<body>
  <p>Hello World</p>
</body>
```

4.3. Tooltips

- A tooltip is often used to specify extra information about something when the user moves the mouse pointer over an element.

Example:

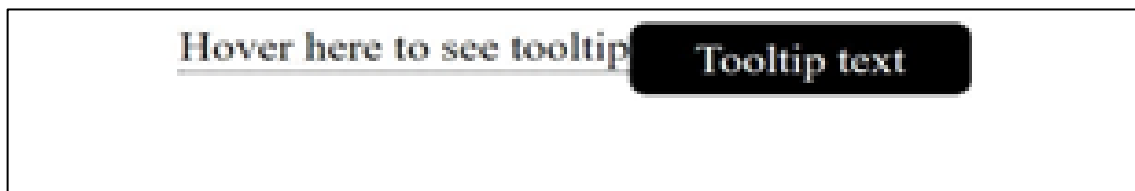
```
<!DOCTYPE html>
<html>
<style>
.tooltip
{
    position: relative;
    display: inline-block;
    border-bottom: 1px dotted black;
}
.tooltip .tooltiptext
{
    visibility: hidden;
    width: 120px;
    background-color: black;
    color: #fff;
    text-align: center;
```

```

        border-radius: 6px;
        padding: 5px 0;
        position: absolute;
        z-index: 1;
    }
    .tooltip:hover .tooltiptext
    {
        visibility: visible;
    }
</style>
<body style="text-align:center;">
<div class="tooltip">Hover here to see tooltip
    <span class="tooltiptext">Tooltip text</span>
</div>
</body>
</html>

```

Output:-



Example Explained: -

- Use a container element (like <div>) and add the "tooltip" class to it. When the user mouse over this <div>, it will show the tooltip text.
- The tooltip text is placed inside an inline element (like) with class="tooltiptext".
- The tooltip class use position:relative, which is needed to position the tooltip text (position:absolute).
- The tooltiptext class holds the actual tooltip text. It is hidden by default, and will be visible on hover.
- The: hover selector is used to show the tooltip text when the user moves the mouse over the <div> with class="tooltip".

4.4. Styling Images

- Styling images in CSS works exactly the same way as styling any element using the Box Model of padding, borders, and margins for the content.
- **Border-radius:** - Use the border-radius property to create rounded images.

Example: -

```

img
{
border-radius:50%;
}

```

- **Responsive image:** - Responsive images will automatically adjust to fit the size of the screen.

Example: -

```

img

```



```
{
max-width:100%;
height:auto;
}
```

- *center an image*: -To center an image, set left and right margin to auto and make it into a block element.

Example: -

```
img
{
display: block;
margin-left: auto;
margin-right: auto;
width: 50%;
}
```

- *Transparent Image*: The opacity property is used to set the image transparent. The opacity value lies between 0.0 to 1.0. The lower value, the more transparent.

Example: -

```
img
{
opacity: 0.5;
}
```

- *Image Filters*: -The CSS filter property adds visual effects (like blur and saturation) to an element.

Example: -

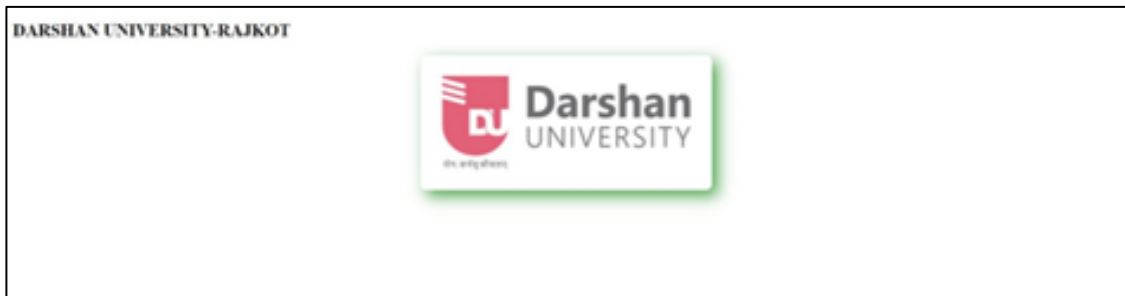
```
img
{
filter: grayscale(100%);
}
```

Example: -

```
<!DOCTYPE html>
<html>
<head>
<style>
img
{
border-radius: 8px;
opacity:0.7;
display: block;
margin-left: auto;
margin-right: auto;
max-width: 100%;
height: auto;
filter: drop-shadow(8px 8px 10px green);
}
</style>
```

```
</head>
<body>
  <h2>DARSHAN UNIVERSITY-RAJKOT</h2>
  
</body>
</html>
```

Output: -



4.5. Media Query

- Media queries are a way to target browser by certain characteristics, features, and user preferences, then apply styles or run other code based on those things.
- Perhaps the most common media queries in the world are those that target particular viewport ranges and apply custom styles, which is the whole idea of responsive design.

Syntax:-

```
@media [media-type] ([media-feature])
{
  /* Styles! */
}
```

- It consists of:
 - A media type, which tells the browser what kind of media this code is for (e.g. print, screen etc...).
 - A media feature, which is a rule, or test that must be passed for the contained CSS to be applied.
 - A set of CSS rules that will be applied if the test passes and the media type is correct.
- Media Types:
 - all
 - print
 - screen
 - speech
- Media Features:
 - width & height
 - min-width, min-height, max-width & max-height
 - orientation
- The viewport meta tag:
<meta name="viewport" content="width=device-width, initial-scale=1.0">
 - This is needed because by default, most mobile browsers lie about their viewport width.

- Non-responsive sites commonly look really bad when rendered in a narrow viewport, so mobile browsers usually render the site with a viewport width wider than the real device width by default (usually 960 pixels), and then shrink the rendered result so that it fits in the display.

Example: -

```
<style>
#div1
{
    background-color: red;
    width: 50%;
    float: left;
}
#div2
{
    background-color: yellow;
    width: 50%;
    float: left;
}
@media (max-width: 600px)
{
#div1
{
    width: 100%;
}
#div2
{
    width: 100%;
}
}
</style>
<body>
    <div id="div1">Hello</div>
    <div id="div2">World</div>
</body>
```

Note: you have to add viewport meta tag in the head section

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

- We can also use and/or logic in media query, for example
 - We can use and as a separator if we want two condition to satisfy before applying the CSS.
Example: -
@media screen and (min-width: 600px) and (max-width: 900px) {
body
{
color: blue;
}

- ```

 }

```
- We can use, (comma) as a separator if we want anyone of the condition to satisfy before applying the CSS.

Example:

```

 @media (min-width: 600px), (orientation: landscape) {
 body {
 color: blue;
 }
 }

```

*Example: -*

```

<!DOCTYPE html>
<html>
<head>
 <meta charset="utf-8">
 <meta name="viewport" content="width=device-width, initial-scale=1">
 <style type="text/css">
 body
 {
 font-size:20px;
 }
 #div1
 {
 background-color:red;
 width:25%;
 float:left;
 color:white;
 }
 #div2
 {
 background-color:yellow;
 width:25%;
 float:left;
 }
 #div3
 {
 background-color:green;
 width:25%;
 float:left;
 color:white;
 }
 }

 #div4
 {
 background-color:blue;
 width:25%;
 float:left;
 color:white;
 }

```

```
@media(min-width:600px) and (max-width: 900px)
{
 #div1
 {
 width:50%;
 }
 #div2
 {
 width:50%;
 }
 #div3
 {
 width:50%;
 }
 #div4
 {
 width:50%;
 }
}
@media(max-width:600px)
{
 #div1
 {
 width:100%;
 }
 #div2
 {
 width:100%;
 }
 #div3
 {
 width:100%;
 }
 #div4
 {
 width:100%;
 }
}
</style>
</head>
<body>
 <div id="div1">Web Designing</div>
 <div id="div2">C Lang.</div>
 <div id="div3">Maths</div>
 <div id="div4">Communication Skill</div>
</body>
</html>
```

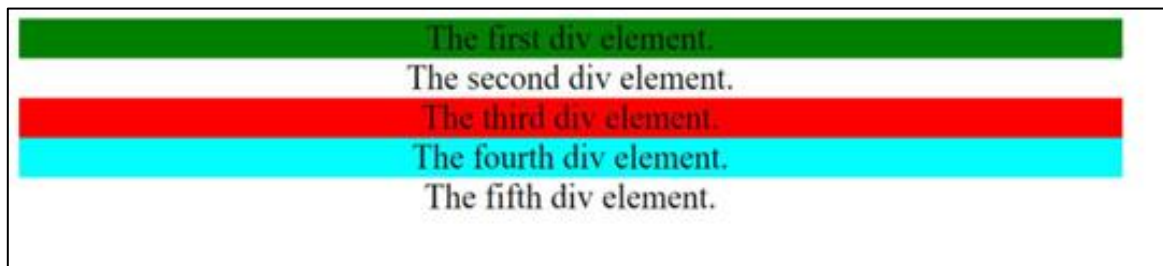
## 4.6. Wildcard Selector

- Wildcard selector is used to select multiple elements simultaneously.
- It selects similar type of class name or attribute and use CSS property. \* Wildcard also known as containing wildcard.
- Some of the wild card selectors are,
  - **[attribute\*="str"] Selector**
    - It will select all the elements with the given attribute containing the str.
  - **[attribute^="str"] Selector**
    - It will select all the elements with the given attribute starts with the str.
  - **[attribute\$="str"] Selector**
    - It will select all the elements with the given attribute ends with the str.

### Example

```
<style>
[class*="str"]
{
 background: green;
}
[class^="str"]
{
 background: red;
}
[class$="str"]
{
 background: cyan;
}
body
{
 width:60%;
 text-align:center;
}
</style>
</head>
<body>
<div class="my-strt">The first div element.</div>
<div class="second">The second div element.</div>
<div class="str-third">The third div element.</div>
<div class="end-str">The fourth div element.</div>
<div class="start">The fifth div element.</div>
</body>
```

### Output



## 4.7. Variables

- The variables in CSS are just like simple variables of any other programming language.
  - These variables are used to store values and have a scope in which the variables can be used.
  - A variable is defined by using two dashes (–) at the beginning and then the name which is case-sensitive.
  - The benefit of variables is that it allows the same values to be reused at multiple places and updated/modified from one place.
  - Also, the variable names are easier to understand and use, as compared to the values of colors, as it avoids of being copying & pasting the value of the colors over and over again.
- Syntax:** `--var( --custom-name, value );`
- The [var\(\) function](#) can be used to take the values of the variables in CSS.

**Parameters:** The variable `var()` accepts two parameters which are listed below:

- `--custom-name`: It is a required parameter that accepts the custom property name starting with two dashes.
- `value`: It is an optional parameter. It accepts a fallback value which is used when a custom property is invalid.

### 4.7.1. Working of CSS var() function

- The scope of the variables in CSS can either be local or global. We can utilize the global variables in the entire document whereas the local variable can only be used inside the selector where the variable is declared, within the scope.
- For creating the variables with global scope, we need to declare the variable inside the `:root` selector, where it compares for the document's root element.
- For creating the variable in the local scope, we can declare the variable inside the selector where it can be used within the scope.

**Example: -**

```
<style>
 :root
 {
 --bg-color: green;
 --txt-color: white;
 }
 body
 {
 background-color: var(--bg-color);
 }
 h1
```

```

 {
 color: var(--txt-color);
 }
 div
 {
 color: var(--txt-color);
 }
</style>
<body>
 <h1>DARSHAN UNIVERSITY-RAJKOT</h1>
 <div>We offers several courses like B.Tech,B.Sc,M.Sc,Diploma,B.B.A,B.C.A,M.C.A
</div>
</body>

```

*Output:-*



## 4.8. Gradients

- CSS gradients let you display smooth transitions between two or more specified colors.
- CSS defines two types of gradients:
  - Linear Gradients (goes down/up/left/right/diagonally)
    - To create a linear gradient you must define at least two color stops.
    - Color stops are the colors you want to render smooth transitions among.
    - You can also set a starting point and a direction (or an angle) along with the gradient effect.
  - Radial Gradients (defined by their center)
    - The radial-gradient() function sets a radial gradient as the background image.
    - A radial gradient is defined by its center.
    - To create a radial gradient you must define at least two color stops.

### 4.8.1. Linear Gradient

- To create a linear gradient you must define at least two color stops.
- Color stops are the colors you want to render smooth transitions among.
- You can also set a starting point and a direction (or an angle) along with the gradient effect.

**Syntax:-**background-image:linear-gradient(direction,color-stop1,color-stop2);

direction:-to bottom,to top,to right,to left,to bottom left,180deg etc.

**Example: -**

```

<style>
#grad1
{

```

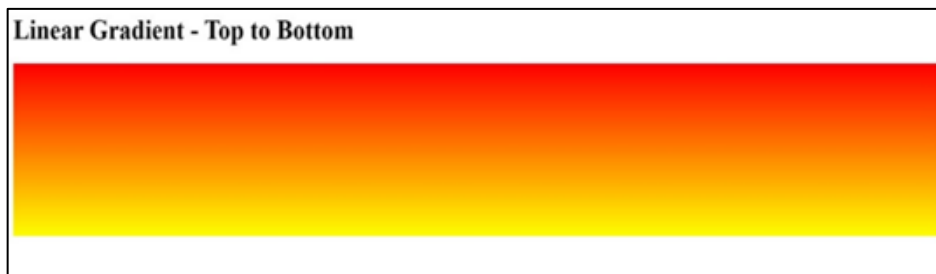


```

 height: 200px;
 background-image: linear-gradient(red, yellow);
 }
</style>
<body>
 <h1>Linear Gradient - Top to Bottom</h1>
 <div id="grad1"></div>
</body>

```

*Output: -*



#### 4.8.2. Radial Gradient

- To create a linear gradient you must define at least two color stops.
  - Color stops are the colors you want to render smooth transitions among.
  - You can also set a starting point and a direction (or an angle) along with the gradient effect.
- Syntax:-**background-image: radial-gradient(shape size at position, start-color, ..., last-color);
- shape: Ellipse (default), Circle
  - size: farthest-corner (default), closest-corner, farthest-side, closest-side
  - position: Center (default)

**Example: -**

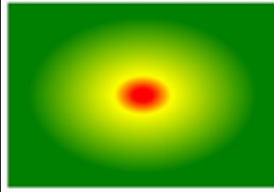
```

<style>
#grad1
{
 height: 150px;
 width: 200px;
 background-image: radial-gradient(red 5%, yellow 15%, green 60%);
}
</style>
<body>
<h1>Radial Gradient - Differently Spaced Color Stops</h1>
<div id="grad1"></div>
</body>

```

*Output: -*

### Radial Gradient - Differently Spaced Color Stops



## 4.9. Pseudo class and Elements

### 4.9.1. CSS Pseudo element

- CSS pseudo-elements are used to add special effects to some selectors.
- For example, it can be used to:

- Style the first letter, or line, of an element
- Insert content before, or after, the content of an element.

Syntax:-

Selector::pseudo element

{

Property: value;

}

- The most commonly used pseudo-elements are as follows
  - first-line:-Use this element to add special styles to the first line of the text in a selector.
  - first-letter:-Use this element to add special style to the first letter of the text in a selector.
  - before:-Use this element to insert some content before an element.
  - after:-Use this element to insert some content after an element.
  - selection:-The ::selection pseudo-element matches the portion of an element that is selected by a user. Color and background property is applied to this pseudo elements.

*Example: -*

```
<!DOCTYPE html>
<html>
<head>
<style>
p::first-letter
{
 color: #ff0000;
 font-size: xx-large;
}
p::first-line
{
 color: #0000ff;
 font-variant: small-caps;
}
::selection
{
```

```

 color: red;
 background: yellow;
 }
 h1::before
 {
 content: url(du.png);
 }
 /*h1::after
 {
 content: url(du.png);
 }*/
</style>
</head>
<body>
 <h1>Darshan University-Rajkot</h1>
 <p>You can combine the :: first-letter and ::first-line pseudo-elements to add a special effect to the
 first letter and the first line of a text!</p>
</body>

```

*Output:-*



#### 4.9.2. Pseudo class

- A pseudo-class is used to define a special state of an element.
- For example, it can be used to:
  - Style an element when a user mouse over it
  - Style visited and unvisited links differently
  - Style an element when it gets focus
- Some important pseudo classes are
  - :active: It is used to select and style the active link. A link becomes active when you click on it. The :active selector can be used on all elements, not only links.
  - :disabled: It targets a form element in the disabled state. An element in a disabled state can't be selected, checked, activated or gain focus.
  - :first-child: The :first-child pseudo-class matches a specified element that is the first child of another element.
  - :last-child: The :last-child selector matches every element that is the last child of its parent.
  - :focus: The :focus selector is used to select the element that has focus. The :focus selector is allowed on elements that accept keyboard events or other user inputs.
  - :visited: Use this class to add special style to a visited link.
  - :hover: Use this class to add special style to an element when you mouse over it.

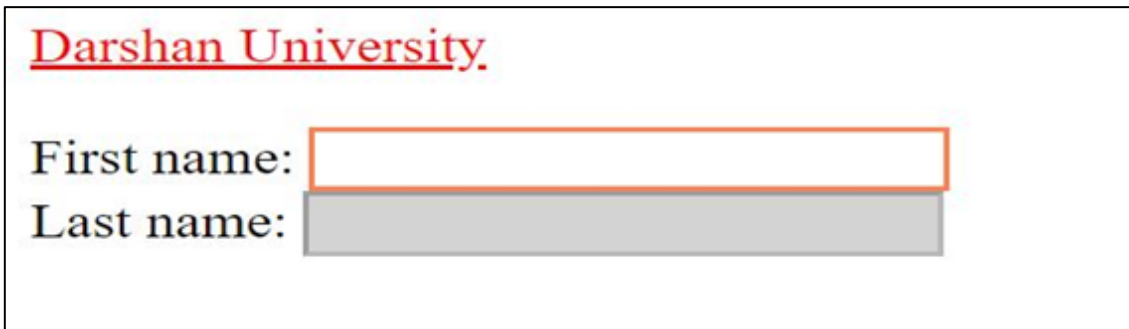
*Example:-*

```
<style>
input:focus
{
 background-color: yellow;
}
a:visited
{
 color:red;
}
a:hover
{
 background-color:gold;
 color:blue;
}
a:active
{
 color:blue;
}
input:first-child
{
 border:2px solid coral;
}
input:disabled
{
 background-color:lightgrey;
}
</style>
<body>
Darshan University

<form>
 First name: <input type="text" name="firstname">

 Last name: <input type="text" name="lastname" disabled>
</form>
</body>
```

*Output: -*



**Darshan University.**

First name:

Last name:

# Unit-5

## Bootstrap

---

### 5.1. Introduction to Bootstrap

- Bootstrap is a free and open-source tool collection for creating responsive web applications.
- It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites.
- It solves many problems which we had once, one of which is the cross-browser compatibility issue.
- Nowadays, the websites are perfect for all the browsers (IE, Firefox, and Chrome) and for all sizes of screens (Desktop, Tablets, Phablets, and Phones). All thanks to Bootstrap developers Mark Otto and Jacob Thornton of Twitter, though it was later declared to be an open-source project.

#### 5.1.1. Advantages of Bootstrap

- *Easy to use:-* Anybody with just basic knowledge of HTML and CSS can start using Bootstrap
- *Responsive features:-* Bootstrap's responsive CSS adjusts to phones, tablets, and desktops
- *Mobile-first approach:-* In Bootstrap, mobile-first styles are part of the core framework
- *Browser compatibility:* Bootstrap 4 is compatible with all modern browsers (Chrome, Firefox, Internet Explorer 10+, Edge, Safari, and Opera) Download and Load Bootstrap
- There are two ways to start using Bootstrap on your own web site.
  - Include Bootstrap from a CDN
  - Download Bootstrap from [getbootstrap.com](https://getbootstrap.com)

#### 5.1.2. Bootstrap CDN

- If you don't want to download and host Bootstrap yourself, you can include it from a CDN (Content Delivery Network).
- `<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">`
- `<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IIRH9sENBOOLRn5q+8nbTov4+1p" crossorigin="anonymous"></script>`.

#### 5.1.3. Downloading Bootstrap from [getbootstrap.com](https://getbootstrap.com)

- If you want to download and host Bootstrap yourself, go to <https://getbootstrap.com>
- We can download compiled CSS & JS, Source Code or include it with package managers like npm.

#### 5.1.4. Important Globals

- *Add the HTML5 doctype:-*
  - Bootstrap requires the use of the HTML5 doctype. Without it, you'll see some incomplete styling, but including it shouldn't cause any considerable problems.  
`<!DOCTYPE html>`  
`<html lang="en">`  
...  
`</html>`

- **Responsive Meta tag:** -
  - Bootstrap is developed mobile first, a strategy in which we optimize code for mobile devices first and then scale up components as necessary using CSS media queries.
  - To ensure proper rendering and touch zooming for all devices, add the responsive viewport meta tag to your <head>.
 

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```
  - The width=device-width part sets the width of the page to follow the screen-width of the device (which will vary depending on the device).
  - The initial-scale=1 part sets the initial zoom level when the page is first loaded by the browser.
- **Box Sizing:** -
  - For more straightforward sizing in CSS, they switch the global box-sizing value from content-box to border-box. This ensures padding does not affect the final computed width of an element.
- **Reboot**
  - It is Explain as separate topic in next point please look in that topic.

#### 5.1.5. Reboot

- It's a reset CSS from bootstrap, included from bootstrap 4.
- Reboot builds upon Normalize, providing many HTML elements with somewhat opinionated styles using only element selectors.
- Additional styling is done only with classes. For example, we reboot some <table> styles for a simpler baseline and later provide .table, .table-bordered, and more.

#### 5.1.6. Starter Template using CDN (just for reference)

```
<!doctype html>
<html lang="en">
<head>
 <!-- Required meta tags -->
 <meta charset="utf-8">
 <meta name="viewport" content="width=device-width, initial-scale=1">

 <!-- Bootstrap CSS -->
 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">
 <title>Hello, world!</title>
</head>
<body>
 <!-- WRITE CODE HERE -->
 <h1>Hello, world!</h1>
 <!-- Bootstrap Bundle with Popper -->
```

```
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBOOLRN5q+8nbTov4+1p"
crossorigin="anonymous"></script>
</body>
</html>
```

## 5.2. Breakpoints

- Breakpoints are widths that determine how your responsive layout behaves across device or viewport sizes in Bootstrap.
- Breakpoints are the building blocks of responsive design. Use them to control when your layout can be adapted at a particular viewport or device size.
- Available Breakpoints

Table 5.2 Breakpoints

Point	Class infix	Dimensions
Extra small	None	<576px
Small	sm	≥576px
Medium	md	≥768px
Large	lg	≥992px
Extra Large	xl	≥1200px
Extra Extra Large	xxl	≥1400px

## 5.3. Containers

- Containers are a fundamental building block of Bootstrap that contain, pad, and align your content within a given device or viewport.
- Containers are required when using bootstrap default grid system.
- Containers can be nested; most layouts do not require a nested container.
- Bootstrap comes with three different containers:
- .container, which sets a max-width at each responsive breakpoint
  - **.container-fluid**, which is width: 100% at all breakpoints
  - **.container-{breakpoint}**, which is width: 100% until the specified breakpoint

Table 5.3 Containers

	Extra small <576px	Small ≥576px	Medium ≥768px	Large ≥992px	X-Large ≥1200px	XX-Large ≥1400px
.container	100%	540px	720px	960px	1140px	1320px
.container-sm	100%	540px	720px	960px	1140px	1320px
.container-md	100%	100%	720px	960px	1140px	1320px
.container-lg	100%	100%	100%	960px	1140px	1320px
.container-xl	100%	100%	100%	100%	1140px	1320px
.container-xxl	100%	100%	100%	100%	100%	1320px
.container-fluid	100%	100%	100%	100%	100%	100%

- *Example: -*

```
<div class="container">
 100% wide until small breakpoint
</div>
<div class="container-sm">
 100% wide until small breakpoint
</div>
<div class="container-md">
 100% wide until medium breakpoint
</div>
<div class="container-lg">
 100% wide until large breakpoint
</div>
<div class="container-xl">
 100% wide until extra large breakpoint
</div>
<div class="container-xxl">
 100% wide until extra extra large breakpoint
</div>
<div class="container-fluid">
 100% wide in all screen size
</div>
```

## 5.4. Bootstrap Grid System

- The Bootstrap Grid System is used for layout, specifically Responsive Layouts.
- Understanding how it works is vital to understanding Bootstrap.
- The Grid is made up of groupings of Rows & Columns inside Containers.
- There are three main components in Grid System
  - Container(s)
  - Row(s)
  - Column(s)



col-md-1	col-md-1	col-md-1	col-md-1	col-md-1	col-md-1	col-md-1	col-md-1	col-md-1	col-md-1	col-md-1	col-md-1
col-md-2		col-md-2		col-md-2		col-md-2		col-md-2		col-md-2	
col-md-3			col-md-3			col-md-3		col-md-3			
col-md-4				col-md-4				col-md-4			
col-md-6							col-md-6				
col-md-7							col-md-5				
col-md-8								col-md-4			
col-md-9										col-md-3	
col-md-10											col-md-2
col-md-11											col-md-1
col-md-12											
col-md-5					col-md-3			col-md-4			

Figure 5.4 Bootstrap Grid System

- Grid supports six responsive breakpoints; this means you can control container and column sizing and behaviour by each breakpoint.
- .containers centrally and horizontally pad your content, .container for a responsive pixel width, .container-fluid for width: 100% across all viewports and devices, or a responsive container (e.g., .container-md) for a combination of fluid and pixel widths.
- Rows are wrappers for columns, each column has horizontal padding (called a gutter) for controlling the space between them.
- Columns are incredibly flexible, there are 12 template columns available per row, allowing you to create different combinations of elements that span any number of columns. Column classes indicate the number of template columns to span (e.g., col-4 spans four). Widths are set in percentages so you always have the same relative sizing.

#### 5.4.1. Grid Classes

- The Bootstrap grid system has six classes:
  - .col- (extra small devices - screen width less than 576px)
  - .col-sm- (small devices - screen width equal to or greater than 576px)
  - .col-md- (medium devices - screen width equal to or greater than 768px)
  - .col-lg- (large devices - screen width equal to or greater than 992px)
  - .col-xl- (xlarge devices - screen width equal to or greater than 1200px)
  - .col-xxl- (xxlarge devices - screen width equal to or greater than 1400px)

##### 5.4.1.2. Basic Structure of a Bootstrap Grid

```
<!-- Control the column width, and how they should appear on different devices -->
<div class="row">
```

```
<div class="col-*-*"></div>
<div class="col-*-*"></div>
</div>
```

```
<!-- Or let Bootstrap automatically handle the layout -->
<div class="row">
 <div class="col"></div>
 <div class="col"></div>
 <div class="col"></div>
</div>
```

- **First example:** - create a row (<div class="row">). Then, add the desired number of columns (tags with appropriate .col-\*-\* classes). The first star (\*) represents the responsiveness: sm, md, lg, xl or xxl, while the second star represents a number, which should add up to 12 for each row.
- **Second example:** - instead of adding a number to each col, let bootstrap handle the layout, to create equal width columns: two "col" elements = 50% width to each col, while three cols = 33.33% width to each col. Four cols = 25% width, etc. You can also use .col-sm|md|lg|xl|xxl to make the columns responsive.
- **Three Equal Columns:** -

.col	.col	.col
------	------	------

- The following example shows how to create three equal-width columns, on all devices and screen widths.
- Example:
 

```
<div class="row">
 <div class="col">.col</div>
 <div class="col">.col</div>
 <div class="col">.col</div>
</div>
```

- **Responsive Columns:** -

.col-sm-3	.col-sm-3	.col-sm-3	.col-sm-3
-----------	-----------	-----------	-----------

- The following example shows how to create four equal-width columns starting at tablets and scaling to extra-large desktops. On mobile phones or screens that are less than 576px wide, the columns will automatically stack on top of each other.
- Example:
 

```
<div class="row">
 <div class="col-sm-3">.col-sm-3</div>
 <div class="col-sm-3">.col-sm-3</div>
 <div class="col-sm-3">.col-sm-3</div>
 <div class="col-sm-3">.col-sm-3</div>
</div>
```

- *Two Unequal Responsive Columns: -*

<code>.col-sm-4</code>	<code>.col-sm-8</code>
------------------------	------------------------

- The following example shows how to get two various-width columns starting at tablets and scaling to large extra desktops.
- Example:

```
<div class="row">
 <div class="col-sm-4">.col-sm-4</div>
 <div class="col-sm-8">.col-sm-8</div>
</div>
```

#### 5.4.2. Grid Options

- The following table summarizes how the Bootstrap grid system works across different screen sizes:

*Table 5.4.2 Grid Options*

	Extra small (<576px)	Small (≥576px)	Medium (≥768px)	Large (≥992px)	Extra Large (≥1200px)	Extra Extra Large (≥1400px)
Container max-width	None (auto)	540px	720px	960px	1140px	1320px
Class prefix	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-	.col-xxl-
No. of columns	12	12	12	12	12	12
Gutter width	1.5rem (.75rem on left and right)	1.5rem (.75rem on left and right)	1.5rem (.75rem on left and right)	1.5rem (.75rem on left and right)	1.5rem (.75rem on left and right)	1.5rem (.75rem on left and right)

## 5.5. Utility Classes

### 5.5.1. Colors

- The classes for text colors are: `.text-muted`, `.text-primary`, `.text-success`, `.text-info`, `.text-warning`, `.text-danger`, `.text-secondary`, `.text-white`, `.text-dark`, `.text-body` (default body color/often black) and `.text-light`

- Example:

```
<p class="text-primary">.text-primary</p>
<p class="text-secondary">.text-secondary</p>
<p class="text-success">.text-success</p>
<p class="text-danger">.text-danger</p>
<p class="text-warning bg-dark">.text-warning</p>
<p class="text-info bg-dark">.text-info</p>
<p class="text-light bg-dark">.text-light</p>
<p class="text-dark">.text-dark</p>
<p class="text-body">.text-body</p>
<p class="text-muted">.text-muted</p>
```

```
<p class="text-white bg-dark">.text-white</p>
<p class="text-black-50">.text-black-50</p>
<p class="text-white-50">.text-white-50</p>
```

### 5.5.2. Background color

- Similar to the contextual text color classes, set the background of an element to any contextual class.
- Example:

```
<p class="bg-primary">This text is important.</p>
<p class="bg-success">This text indicates success.</p>
<p class="bg-info">This text represents some information.</p>
<p class="bg-warning">This text represents a warning.</p>
<p class="bg-danger">This text represents danger.</p>
<p class="bg-secondary">Secondary background color.</p>
<p class="bg-dark">Dark grey background color.</p>
<p class="bg-light">Light grey background color.</p>
```

### 5.5.3. Border

- **Additive:-** Use border utilities to add an element's borders. Choose from all borders or one at a time.
 

```



```
- **Subtractive:-** Use border utilities to remove an element's borders. Choose from all borders or one at a time.
 

```



```
- **Border color:-** Change the border color using utilities built on our theme colors.
 

```



```
- **Border-width:-** Use .border-1 to .border-5 to change the width of the border.
 

```



```
- **Border-radius:-** Add classes to an element to easily round its corners.
 

```

```

```



```

#### 5.5.4. Display

- Display utility classes that apply to all breakpoints, from xs to xxl, have no breakpoint abbreviation in them. This is because those classes are applied from min-width: 0; and up, and thus are not bound by a media query. The remaining breakpoints, however, do include a breakpoint abbreviation.
- As such, the classes are named using the format:
  - .d-{value} for xs
  - .d-{breakpoint}-{value} for sm, md, lg, xl, and xxl.
- Where value is one of:
  - none
  - inline
  - inline-block
  - block
  - grid
  - table
  - table-cell
  - table-row
  - flex
  - inline-flex

#### 5.5.5. Flex

- Quickly manage the layout, alignment, and sizing of grid columns, navigation, components, and more with a full suite of responsive flexbox utilities. For more complex implementations, custom CSS may be necessary.

- Example:

```
<div class="d-flex p-2 bd-highlight">I'm a flexbox container!</div>
```

I'm a flexbox container!

```
<div class="d-inline-flex p-2 bd-highlight">I'm an inline flexbox container!</div>
```

I'm an inline flexbox container!

- Responsive variations also exist for .d-flex and .d-inline-flex.
  - .d-flex
  - .d-inline-flex
  - .d-sm-flex
  - .d-sm-inline-flex
  - .d-md-flex

- .d-md-inline-flex
- .d-lg-flex
- .d-lg-inline-flex
- .d-xl-flex
- .d-xl-inline-flex
- .d-xxl-flex
- .d-xxl-inline-flex

#### 5.5.6. Float

- These utility classes float an element to the left or right, or disable floating, based on the current viewport size using the CSS float property. !important is included to avoid specificity issues. These use the same viewport breakpoints as our grid system. Please be aware float utilities have no effect on flex items.
- Example:
 

```
<div class="float-start">Float start on all viewport sizes</div>

<div class="float-end">Float end on all viewport sizes</div>

<div class="float-none">Don't float on all viewport sizes</div>
```
- Responsive variations also exist for each float value.
- Example:
 

```
<div class="float-sm-start">Float start on viewports sized SM (small) or wider</div>

<div class="float-md-start">Float start on viewports sized MD (medium) or wider</div>

<div class="float-lg-start">Float start on viewports sized LG (large) or wider</div>

<div class="float-xl-start">Float start on viewports sized XL (extra-large) or wider</div>

```
- Here are all the support classes:
  - .float-start
  - .float-end
  - .float-none
  - .float-sm-start
  - .float-sm-end
  - .float-sm-none
  - .float-md-start
  - .float-md-end
  - .float-md-none
  - .float-lg-start
  - .float-lg-end
  - .float-lg-none
  - .float-xl-start
  - .float-xl-end
  - .float-xl-none
  - .float-xxl-start
  - .float-xxl-end
  - .float-xxl-none

#### 5.5.7. Interaction

- Utility classes that change how users interact with contents of a website.
- Text selection:-
- Change the way in which the content is selected when the user interacts with it.

```
<p class="user-select-all">This paragraph will be entirely selected when clicked by the user.</p>
<p class="user-select-auto">This paragraph has default select behavior.</p>
<p class="user-select-none">This paragraph will not be selectable when clicked by the user.</p>
```

- **Pointer events**

- Bootstrap provides .pe-none and .pe-auto classes to prevent or add element interactions.

```
<p>This link can not be
clicked.</p>
```

```
<p>This link can be clicked (this is default behavior).</p>
```

```
<p class="pe-none">This link can not be clicked
because the <code>pointer-events</code> property is inherited from its parent. However, this link has a <code>pe-auto</code> class and can be clicked.</p>
```

### 5.5.8. Opacity

- The opacity property sets the opacity level for an element. The opacity level describes the transparency level, where 1 is not transparent at all, .5 is 50% visible, and 0 is completely transparent.
- Set the opacity of an element using .opacity-{value} utilities.

- Example:

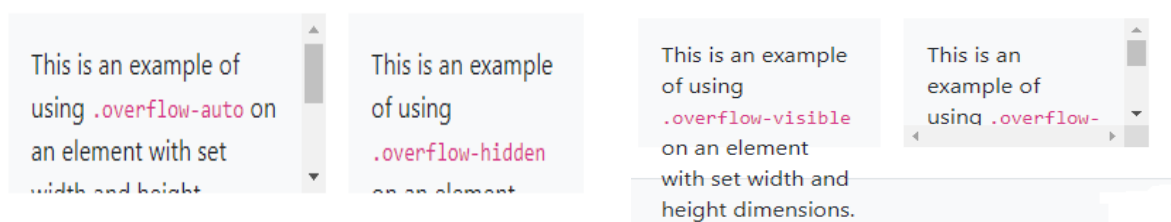
```
<div class="opacity-100">...</div>
<div class="opacity-75">...</div>
<div class="opacity-50">...</div>
<div class="opacity-25">...</div>
```

### 5.5.9. Overflow

- Use these shorthand utilities for quickly configuring how content overflows an element

- Example:

```
<div class="overflow-auto">...</div>
<div class="overflow-hidden">...</div>
<div class="overflow-visible">...</div>
<div class="overflow-scroll">...</div>
```



### 5.5.10. Position

- Use these shorthand utilities for quickly configuring the position of an element.

- Example:

```
<div class="position-static">...</div>
<div class="position-relative">...</div>
<div class="position-absolute">...</div>
<div class="position-fixed">...</div>
<div class="position-sticky">...</div>
```

### 5.5.11. Shadow

- Add or remove shadows to elements with box-shadow utilities.

- Example:

```
<div class="shadow-none p-3 mb-5 bg-light rounded">No shadow</div>
<div class="shadow-sm p-3 mb-5 bg-body rounded">Small shadow</div>
<div class="shadow p-3 mb-5 bg-body rounded">Regular shadow</div>
<div class="shadow-lg p-3 mb-5 bg-body rounded">Larger shadow</div>
```

#### 5.5.12. Display

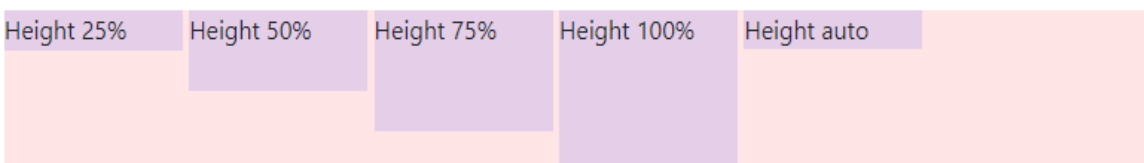
- Easily make an element as wide or as tall with our width and height utilities.
- Example:

```
<div class="w-25 p-3" style="background-color: #eee;">Width 25%</div>
<div class="w-50 p-3" style="background-color: #eee;">Width 50%</div>
<div class="w-75 p-3" style="background-color: #eee;">Width 75%</div>
<div class="w-100 p-3" style="background-color: #eee;">Width 100%</div>
<div class="w-auto p-3" style="background-color: #eee;">Width auto</div>
```



- Example:

```
<div style="height: 100px; background-color: rgba(255,0,0,0.1);">
<div class="h-25 d-inline-block" style="width: 120px; background-color: rgba(0,0,255,.1)">Height
25%</div>
<div class="h-50 d-inline-block" style="width: 120px; background-color: rgba(0,0,255,.1)">Height
50%</div>
<div class="h-75 d-inline-block" style="width: 120px; background-color: rgba(0,0,255,.1)">Height
75%</div>
<div class="h-100 d-inline-block" style="width: 120px; background-color: rgba(0,0,255,.1)">Height
100%</div>
<div class="h-auto d-inline-block" style="width: 120px; background-color:
rgba(0,0,255,.1)">Height auto</div>
</div>
```



#### 5.5.13. Spacing

- Bootstrap includes a wide range of shorthand responsive margin and padding utility classes to modify an element's appearance.



- The classes are named using the format {property}{sides}-{size} for xs and {property}{sides}-{breakpoint}-{size} for sm, md, lg, and xl.
- Where property is one of:
  - m - for classes that set margin
  - p - for classes that set padding
- Where sides is one of:
  - t - for classes that set margin-top or padding-top
  - b - for classes that set margin-bottom or padding-bottom
  - l - for classes that set margin-left or padding-left
  - r - for classes that set margin-right or padding-right
  - x - for classes that set both \*-left and \*-right
  - y - for classes that set both \*-top and \*-bottom
  - blank - for classes that set a margin or padding on all 4 sides of the element
- Where size is one of:
  - 0 - for classes that eliminate the margin or padding by setting it to 0
  - 1 - (by default) for classes that set the margin or padding to \$spacer \* .25
  - 2 - (by default) for classes that set the margin or padding to \$spacer \* .5
  - 3 - (by default) for classes that set the margin or padding to \$spacer
  - 4 - (by default) for classes that set the margin or padding to \$spacer \* 1.5
  - 5 - (by default) for classes that set the margin or padding to \$spacer \* 3
  - auto - for classes that set the margin to auto

Table 5.5.13 Spacing

Class	Description
.m-#	Set the margin to all the sides.
.mt-#	Set the top margin of an element.
.mb-#	Set the bottom margin of an element.
.ms-#	Set the left margin of an element.
.me-#	Set the right margin of an element.
.mx-#	Set the left and right margin of an element.
.my-#	Set the top and bottom margin of an element.
.p-#	Set the padding to all the sides.
.pt-#	Set the top padding of an element.
.pb-#	Set the bottom padding of an element.
.ps-#	Set the left padding of an element.
.pe-#	Set the right padding of an element.
.px-#	Set the left and right padding of an element.
.py-#	Set the top and bottom padding of an element.

- Example:
 

```
<div class="row mx-md-n5">
 <div class="col px-md-5"><div class="p-3 border bg-light">Custom column padding</div> </div>
 <div class="col px-md-5"><div class="p-3 border bg-light">Custom column padding</div> </div>
</div>
```

#### 5.5.14. Text

- Documentation and examples for common text utilities to control alignment, wrapping, weight, and more.

- **Text alignment:-**  

```
<p class="text-justify">Ambitioni dedisse scripsisse iudicaretur. Cras mattis iudicium purus sit amet fermentum. Donec sed odio operae, eu vulputate felis rhoncus. Praeterea iter est quasdam res quas ex communi. At nos hinc posthac, sitientis puros Afros. Petierunt uti sibi concilium totius Galliae in diem certam indicere. Cras mattis iudicium purus sit amet fermentum.</p>
```

```
<p class="text-left">Left aligned text on all viewport sizes.</p>
```

```
<p class="text-center">Center aligned text on all viewport sizes.</p>
```

```
<p class="text-right">Right aligned text on all viewport sizes.</p>
```

```
<p class="text-sm-left">Left aligned text on viewports sized SM (small) or wider.</p>
```

```
<p class="text-md-left">Left aligned text on viewports sized MD (medium) or wider.</p>
```

```
<p class="text-lg-left">Left aligned text on viewports sized LG (large) or wider.</p>
```

```
<p class="text-xl-left">Left aligned text on viewports sized XL (extra-large) or wider.</p>
```
- **Text wrapping and overflow:**
  - **Wrap text with a .text-wrap class.**  

```
<div class="badge badge-primary text-wrap" style="width: 6rem;">
This text should wrap.
</div>
```
  - **Prevent text from wrapping with a .text-nowrap class.**  

```
<div class="text-nowrap bd-highlight" style="width: 8rem;">
This text should overflow the parent.
</div>
```
- **Word break:-**  

```
<p class="text-break"> mmmmmmmmmmmmmmmmmmmmmmmmmmm</p>
```
- **Text transform:-**  

```
<p class="text-lowercase">Lowercased text.</p>
```

```
<p class="text-uppercase">Uppercased text.</p>
```

```
<p class="text-capitalize">CapiTaliZed text.</p>
```
- **Font weight and italics: -**  

```
<p class="font-weight-bold">Bold text.</p>
```

```
<p class="font-weight-bolder">Bolder weight text (relative to the parent element).</p>
```

```
<p class="font-weight-normal">Normal weight text.</p>
```

```
<p class="font-weight-light">Light weight text.</p>
```

```
<p class="font-weight-lighter">Lighter weight text (relative to the parent element).</p>
```

```
<p class="font-italic">Italic text.</p>
```
- **Monospace: -**  

```
<p class="text-monospace">This is in monospace</p>
```
- **Reset color: -**  

```
<p class="text-muted">
Muted text with a reset link. </p>
```
- **Text decoration: -**  

```
Non-underlined link
```

### 5.5.15. Vertical Alignment

- Easily change the vertical alignment of inline, inline-block, inline-table, and table cell elements.

- Change the alignment of elements with the vertical-alignment utilities. Please note that vertical-align only affects inline, inline-block, inline-table, and table cell elements.
- Choose from .align-baseline, .align-top, .align-middle, .align-bottom, .align-text-bottom, and .align-text-top as needed.

- Example:

```
baseline
top
middle
bottom
text-top
text-bottom
```

#### 5.5.16. Visibility Display

- Control the visibility, without modifying the display, of elements with visibility utilities.
- Example:

```
<div class="visible">...</div>
<div class="invisible">...</div>
```

## 5.6. Typography

- Documentation and examples for Bootstrap typography, including global settings, headings, body text, lists, and more.

#### 5.6.1. Headings

- .h1 through .h6 classes are also available, for when you want to match the font styling of a heading but cannot use the associated HTML element.

- Example:

```
<p class="h1">h1. Bootstrap heading</p>
<p class="h2">h2. Bootstrap heading</p>
<p class="h3">h3. Bootstrap heading</p>
<p class="h4">h4. Bootstrap heading</p>
<p class="h5">h5. Bootstrap heading</p>
<p class="h6">h6. Bootstrap heading</p>
```

#### 5.6.2. Display Headings

- Traditional heading elements are designed to work best in the meat of your page content. When you need a heading to stand out, consider using a display heading—a larger, slightly more opinionated heading style.

- Example:

```
<h1 class="display-1">Display 1</h1>
<h1 class="display-2">Display 2</h1>
<h1 class="display-3">Display 3</h1>
<h1 class="display-4">Display 4</h1>
<h1 class="display-5">Display 5</h1>
<h1 class="display-6">Display 6</h1>
```

### 5.6.3. Lead

- Make a paragraph stand out by adding .lead.
- Example:  

```
<p class="lead">
```

This is a lead paragraph. It stands out from regular paragraphs.

```
</p>
```

### 5.6.4. Inline Text Element

- Styling for common inline HTML5 elements.
- Example:  

```
<p>You can use the mark tag to <mark>highlight</mark> text.</p>
```

```
<p>This line of text is meant to be treated as deleted text.</p>
```

```
<p><s>This line of text is meant to be treated as no longer accurate.</s></p>
```

```
<p><ins>This line of text is meant to be treated as an addition to the document.</ins></p>
```

```
<p><u>This line of text will render as underlined.</u></p>
```

```
<p><small>This line of text is meant to be treated as fine print.</small></p>
```

```
<p>This line rendered as bold text.</p>
```

```
<p>This line rendered as italicized text.</p>
```
- Output:

You can use the mark tag to highlight text.

~~This line of text is meant to be treated as deleted text.~~

~~This line of text is meant to be treated as no longer accurate.~~

This line of text is meant to be treated as an addition to the document.

This line of text will render as underlined.

This line of text is meant to be treated as fine print.

**This line rendered as bold text.**

*This line rendered as italicized text.*

### 5.6.5. Black quotes

- For quoting blocks of content from another source within your document.
- Wrap `<blockquote class="blockquote">` around any HTML as the quote.
- Example:  

```
<blockquote class="blockquote">
```

```
<p>A well-known quote, contained in a blockquote element.</p>
```

```
</blockquote>
```
- **Blackquote Alignment:-**  
 Use text utilities as needed to change the alignment of your blockquote.
- Example:  

```
<figure class="text-center">
```

```
<blockquote class="blockquote">
```

```
<p>A well-known quote, contained in a blockquote element.</p>
</blockquote>
<figcaption class="blockquote-footer">
 Someone famous in <cite title="Source Title">Source Title</cite>
</figcaption>
</figure>
```

- Output:

A well-known quote, contained in a blockquote element.  
— Someone famous in Source Title

#### 5.6.6. List

- **Unstyled :-**
  - Remove the default list-style and left margin on list items (immediate children only). This only applies to immediate children list items, meaning you will need to add the class for any nested lists as well.
  - Example:
 

```
<ul class="list-unstyled">
 This is a list.
 It appears completely unstyled.
 Structurally, it's still a list.
 However, this style only applies to immediate child elements.
 Nested lists:

 are unaffected by this style
 will still show a bullet
 and have appropriate left margin

 This may still come in handy in some situations.

```
  - Output:

This is a list.  
It appears completely unstyled.  
Structurally, it's still a list.  
However, this style only applies to immediate child elements.  
Nested lists:

- are unaffected by this style
- will still show a bullet
- and have appropriate left margin

This may still come in handy in some situations.

- **Inline: -**

- Remove a list's bullets and apply some light margin with a combination of two classes, .list-inline and .list-inline-item.
- Example:
 

```
<ul class="list-inline">
 <li class="list-inline-item">This is a list item.
 <li class="list-inline-item">And another one.
 <li class="list-inline-item">But they're displayed inline.

```

#### 5.6.7. Images

- Documentation and examples for opting images into responsive behaviour (so they never become wider than their parent) and add lightweight styles to them—all via classes.
- *Responsive Image:* -
  - Example:
 

```



```
- *Image Shapes:* -
  - Images in Bootstrap are made responsive with .img-fluid. This applies max-width: 100%; and height: auto; to the image so that it scales with the parent width.
  - Example:
 

```

```
- *Image Alignment:* -
  - Example:
 

```



```

#### 5.6.8. Figure

- Documentation and examples for displaying related images and text with the figure component in Bootstrap.
- Anytime you need to display a piece of content—like an image with an optional caption, consider using a <figure>.
- Use the included .figure, .figure-img and .figure-caption classes to provide some baseline styles for the HTML5 <figure> and <figcaption> elements. Images in figures have no explicit size, so be sure to add the .img-fluid class to your <img> to make it responsive.
- Example:
 

```
<figure class="figure">

 <figcaption class="figure-caption">A caption for the above image.</figcaption>
</figure>
```
-

### 5.6.9. Tables

- Bootstrap provides a series of classes that can be used to apply various styling to the tables such as changing the heading appearance, making the rows stripped, adding or removing borders, making rows hoverable, etc. Bootstrap also provides classes for making tables responsive.
- *Simple Table:* -
  - The .table class is used to create simple Bootstrap table. This class name is used with <table> tag to create table.
  - Example:  
`<table class="table"> Table Contents... </table>`
- *Stripped rows:* -
  - The .table-striped class is used to create alternate dark and light rows. Use the combination of table and table-striped classes within the <table> tag to create stripped table.
  - Example:  
`<table class="table table-striped"> Table Contents... </table>`
- *Bordered Table:* -
  - The .table-bordered class is used to add border on all side of table and cell. Use the combination of table and table-bordered classes within the <table> tag to create bordered table.
  - Example:  
`<table class="table table-bordered"> Table Contents... </table>`
- *Hover rows:* -
  - : The .table-hover class is used to add hover effect (change background color to gray when the mouse moves over) on table rows. Use the combination of table and table-hover classes within the <table> tag to create hover rows table.
  - Example:  
`<table class="table table-hover"> Table Contents... </table>`
- *Black/Dark Table:* -
  - The .table-dark class is used to add the black background color of a table. Use the combination of table and table-dark classes within the <table> tag to create dark table.
  - Example:  
`<table class="table table-dark"> Table Contents... </table>`
- *Dark Stripped Table:* -
  - The .table-dark and .table-striped classes are used to create dark stripped table. Use the combination of table, table-dark and table-striped classes within the <table> tag to create the dark stripped table.
  - Example  
`<table class="table table-dark table-striped"> Table Contents... </table>`
- *Dark Hoverable Table:* -
  - The .table-dark and .table-hover classes are used to add hover effect (change background color to dark gray when the mouse moves over) on table rows. Use the combination of table, table-dark and table-hover classes within the <table> tag to create the dark hover effect table.
  - Example  
`<table class="table table-dark table-hover"> Table Contents... </table>`

- **Borderless Table: -**

- The .table-borderless is used to remove the border from table. Use the combination of table and table-borderless classes within the <table> tag to create borderless table.
- Example  

```
<table class="table table-borderless"> Table Contents... </table>
```

- **Colored Table: -**

- Bootstrap provides a number of contextual classes that can be used to color the entire row or a single cell of a table. These classes should be used with the light table and not with the dark table for better appearance. The list of contextual classes is given below.
- table-primary: Blue- Indicates an important action
- table-secondary: Grey- Indicates a slightly less important action
- table-success: Green- Indicates a successful or positive action
- table-danger: Red- Indicates a dangerous or potentially negative action
- table-warning: Orange- Indicates a warning that might need attention
- table-info: Light blue- Indicates a neutral informative change or action
- table-light: Light grey table or table row background
- table-dark: Dark grey table or table row background
- table-active: Grey- Applies the hover color to the table row or table cell

- Example

```
<table class="table">
<thead>
 <tr>
 <th>Firstname</th>
 <th>Lastname</th>
 <th>Email</th>
 </tr>
</thead>
<tbody>
<tr>
 <td>Default</td>
 <td>Defaultson</td>
 <td>def@somemail.com</td>
</tr>
<tr class="table-primary">
 <td>Primary</td>
 <td>Joe</td>
 <td>joe@example.com</td>
</tr>
<tr class="table-success">
 <td>Success</td>
 <td>Doe</td>
 <td>john@example.com</td>
</tr>
</tbody>
```



</table>

## 5.7. Bootstrap Components

- Bootstrap comes with lots of ready components, in this section we will explore those components.
- Here is the list of components from bootstrap 5,

### 5.7.1. Alert

- Provide contextual feedback messages for typical user actions with the handful of available and flexible alert messages.
- Alerts are available for any length of text, as well as an optional dismiss button. For proper styling, use one of the eight required contextual classes (e.g., .alert-success). For inline dismissal, use the alerts jQuery plugin.

- Example:

```
<div class="alert alert-primary" role="alert">
 This is a primary alert—check it out!
</div>
<div class="alert alert-secondary" role="alert">
 This is a secondary alert—check it out!
</div>
<div class="alert alert-success" role="alert">
 This is a success alert—check it out!
</div>
<div class="alert alert-danger" role="alert">
 This is a danger alert—check it out!
</div>
<div class="alert alert-warning" role="alert">
 This is a warning alert—check it out!
</div>
<div class="alert alert-info" role="alert">
 This is a info alert—check it out!
</div>
<div class="alert alert-light" role="alert">
 This is a light alert—check it out!
</div>
<div class="alert alert-dark" role="alert">
 This is a dark alert—check it out!
</div>
```

### 5.7.2. BreadCrumb

- Indicate the current page's location within a navigational hierarchy that automatically adds separators via CSS.

- Example:

```
<nav aria-label="breadcrumb">
 <ol class="breadcrumb">
 <li class="breadcrumb-item active" aria-current="page">Home

</nav>
```

```
<nav aria-label="breadcrumb">
 <ol class="breadcrumb">
 <li class="breadcrumb-item">Home
 <li class="breadcrumb-item active" aria-current="page">Library

</nav>

<nav aria-label="breadcrumb">
 <ol class="breadcrumb">
 <li class="breadcrumb-item">Home
 <li class="breadcrumb-item">Library
 <li class="breadcrumb-item active" aria-current="page">Data

</nav>
```

- Output:

Home

Home / Library

Home / Library / Data

### 5.7.3. Buttons

- Use Bootstrap's custom button styles for actions in forms, dialogs, and more with support for multiple sizes, states, and more.

- *Button Styles: -*

Example:

```
<button type="button" class="btn btn-primary">Primary</button>
<button type="button" class="btn btn-secondary">Secondary</button>
<button type="button" class="btn btn-success">Success</button>
<button type="button" class="btn btn-danger">Danger</button>
<button type="button" class="btn btn-warning">Warning</button>
<button type="button" class="btn btn-info">Info</button>
<button type="button" class="btn btn-light">Light</button>
<button type="button" class="btn btn-dark">Dark</button>
<button type="button" class="btn btn-link">Link</button>
```

- *Outline Buttons:-*

Example:

```
<button type="button" class="btn btn-outline-primary">Primary</button>
<button type="button" class="btn btn-outline-secondary">Secondary</button>
<button type="button" class="btn btn-outline-success">Success</button>
<button type="button" class="btn btn-outline-danger">Danger</button>
<button type="button" class="btn btn-outline-warning">Warning</button>
<button type="button" class="btn btn-outline-info">Info</button>
<button type="button" class="btn btn-outline-light">Light</button>
<button type="button" class="btn btn-outline-dark">Dark</button>
```

- **Button Sizes: -**

Example:

```
<button type="button" class="btn btn-primary btn-lg">Large</button>
<button type="button" class="btn btn-primary">Normal</button>
<button type="button" class="btn btn-primary btn-sm">Small</button>
<button type="button" class="btn btn-primary btn-xs">XSmall</button>
```

- **Block Level Buttons: -**

Example:

```
<button type="button" class="btn btn-primary btn-block">Button 1</button>
```

- **Active/Disabled Buttons: -**

Example:

```
<button type="button" class="btn btn-primary btn-block">Button 1</button>
```

- **Block Level Buttons: -**

- A button can be set to an active (appear pressed) or a disabled (unclickable) state.
- The class .active makes a button appear pressed, and the class .disabled makes a button unclickable.

Example:

```
<button type="button" class="btn btn-primary active">Active Primary</button>
<button type="button" class="btn btn-primary disabled">Disabled Primary</button>
<button type="button" class="btn btn-primary btn-block">Button 1</button>
```

#### 5.7.4. Forms

- Bootstrap provides three types of form layouts:
  - Vertical form (default)
  - Horizontal form
  - Inline form
- Standard rules for all three form layouts
  - Wrap labels and form controls in <div class="form-group"> (needed for optimum spacing)
  - Add class below class to all textual elements

Table 5.7.4 Bootstrap Forms

Form Element	Bootstrap Class
Text, Password, File, Color, etc...	form-control
Select	form-select
Checkbox, Radio	form-check-input
Range	form-range


- **Vertical Form: -**

- Example

```
<form action="#"> <!-- vertical is the default layout so no need to specify class -->
<div class="form-group">
 <label for="txtEmailAddress">Email address:</label>
 <input type="email" class="form-control" id="txtEmailAddress">
</div>
<div class="form-group">
```

```
<label for="txtPassword">Password:</label>
<input type="password" class="form-control" id="txtPassword">
</div>
<div class="checkbox">
 <label><input type="checkbox" class="form-check-input"> Remember me</label>
</div>
<button type="submit" class="btn btn-primary">Submit</button>
</form>
```

▪ Output



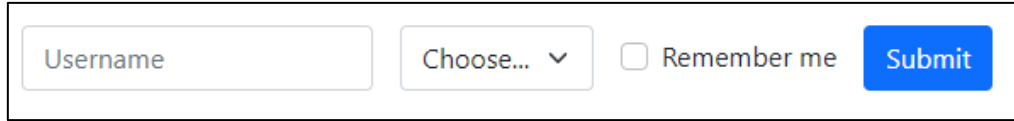
• *Inline Form:* -

- Use the .row-cols-\* classes to create responsive horizontal layouts.
- The .col-12 helps stack the form controls.
- The .align-items-center aligns the form elements to the middle, making the .form-checkbox align properly.
- Example

```
<form class="row row-cols-lg-auto g-3 align-items-center">
 <div class="col-12">
 <input type="text" class="form-control" placeholder="Username">
 </div>
 <div class="col-12">
 <select class="form-select">
 <option selected>Choose...</option>
 <option value="1">One</option>
 <option value="2">Two</option>
 <option value="3">Three</option>
 </select>
 </div>
 <div class="col-12">
 <div class="form-check">
 <input class="form-check-input" type="checkbox" id="cbRem">
 <label class="form-check-label" for="cbRem">
 Remember me
 </label>
 </div>
 </div>
 <div class="col-12">
 <button type="submit" class="btn btn-primary">Submit</button>
 </div>
```

</form>

▪ Output

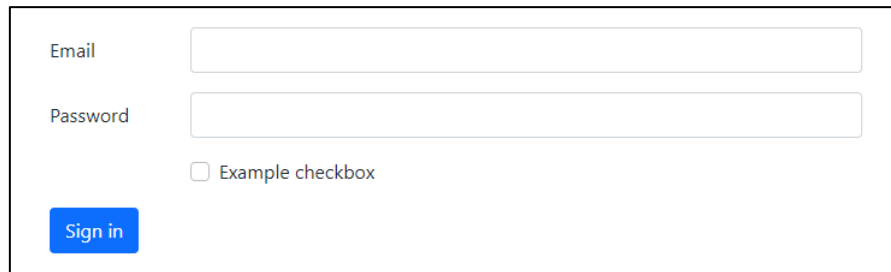


• **Horizontal Form: -**

▪ Example:

```
<form>
<div class="row mb-3">
 <label for="txtMail" class="col-sm-2 col-form-label">Email</label>
 <div class="col-sm-10">
 <input type="email" class="form-control" id="txtMail">
 </div>
</div>
<div class="row mb-3">
 <label for="txtPass" class="col-sm-2 col-form-label">Password</label>
 <div class="col-sm-10">
 <input type="password" class="form-control" id="txtPass">
 </div>
</div>
<div class="row mb-3">
 <div class="col-sm-10 offset-sm-2">
 <div class="form-check">
 <input class="form-check-input" type="checkbox" id="gridCheck1">
 <label class="form-check-label" for="gridCheck1">
 Example checkbox
 </label>
 </div>
 </div>
</div>
<button type="submit" class="btn btn-primary">Sign in</button>
</form>
```

▪ Output:



NOTE : FOR THE MORE BOOTSTRAP COMPONENT PLEASE REFER THE FOLLOWING OFFICIAL BOOTSTRAP WEBSITE LINK

Link :- <https://getbootstrap.com/docs/5.1/getting-started/introduction/>