Introduction to HTML & CSS

HTML: **H**yper**T**ext **M**arkup **L**anguage

- HTML documents are simply text documents with a specific form
  - Documents comprised of **content** and **markup tags**
  - Content: actual information being conveyed
  - The markup tags tell the Web browser **how to display** the page
  - An HTML file must have an **htm** or **html** file extension
  - An HTML file can be created using a **simple text editor**

Our First Example

- If you are running Windows, start Notepad
- If you are on a Mac, start SimpleText
- If you telnet to csupomona.edu, use "pico"
- Type in the following:

```
<html>
<head>
<title>Title of page</title>
</head>
<body>
This is my first homepage. <b>This text is bold</b>
</body>
</html>
```

- Open this file using a browser, and you will see…

HTML Tags

- HTML tags are used to mark-up HTML elements
  - Surrounded by angle brackets **<** and **>**
  - HTML tags normally come in pairs, like <tagname> (start tag) and </tagname> (end tag)
  - The text between the start and end tags is the element content
  - Not case-sensitive
  - Follow the latest web standards:
    - Use lowercase tags

Tag Attributes

- Tags can have attributes that provide additional information to an HTML element
  - Attributes always come in name/value pairs like: name="value"
  - Attributes are always specified in the start tag
  - Attribute values should always be enclosed in quotes. Double quotes are most common.
  - Also case-insensitive: however, lowercase is recommended
  - <tagname a1="v1" a2="v2"></tagname>
  - For example, <table border="0"> is a start tag that defines a table that has no borders

Nested Tags

- Whenever you have HTML tags within other HTML tags, you must close the nearest tag first
- Example:

  <H1> <I> The Nation </I> </H1>

HTML Document Structure

- Entire document enclosed within <html> and </html> tags
- Two subparts:
  - Head
    - Enclosed within <head> and </head>
    - Within the head, more tags can be used to specify title of the page, meta-information, etc.
  - Body
    - Enclosed within <body> and </body>
    - Within the body, content is to be displayed
    - Other tags can be embedded in the body

Structure of a Web Page

- All Web pages share a common structure
- All Web pages should contain a pair of <HTML>, <HEAD>, <TITLE>, and <BODY> tags

```
<HTML>
<HEAD>
<TITLE> Example </TITLE>
</HEAD>
<BODY>
    This is where you would
    include the text and images
    on your Web page.
</BODY>
</HTML>
```

The <TITLE> Tag

- Choose the title of your Web page carefully; The title of a Web page determines its ranking in certain search engines
- The title will also appear on Favorite lists, History lists, and Bookmark lists to identify your page

Comment Statements

- Comment statements are notes in the HTML code that explain the important features of the code
- The comments do not appear on the Web page itself but are a useful reference to the author of the page and other programmers
- To create a comment statement use the <!- - …. --> tags

Page Formatting

- To define the background color, use the BGCOLOR attribute in the <BODY> tag
- To define the text color, use the TEXT attribute in the <BODY> tag

Example

```
<HTML>
<HEAD>
<TITLE> Example </TITLE>
</HEAD>
<body BGCOLOR="red"  TEXT="white"
                              BASEFONT SIZE="10px">
    This is where you would include the text and images
  on your Web page.
</BODY>
</HTML>
```

Text Formatting

- Manipulating text in HTML can be tricky; Oftentimes, what you see is NOT what you get
- For instance, special HTML tags are needed to create paragraphs, move to the next line, and create headings

Text Formatting Tags

<b> - Bold text
 - Important text
<i> - Italic text
 - Emphasized text
<mark> - Marked text
<small> - Small text
<del> - Deleted text
<ins> - Inserted text
<sub> - Subscript text
<sup> - Superscript text

Changing the Font

- The expression

  <FONT FACE = “fontname”> … </FONT>

  can be used to change the font of the enclosed text
- To change the size of text use the expression

  <FONT SIZE=n> …. </FONT>

  where n is a number between 1 and 7

Changing the Font

- To change the color, use

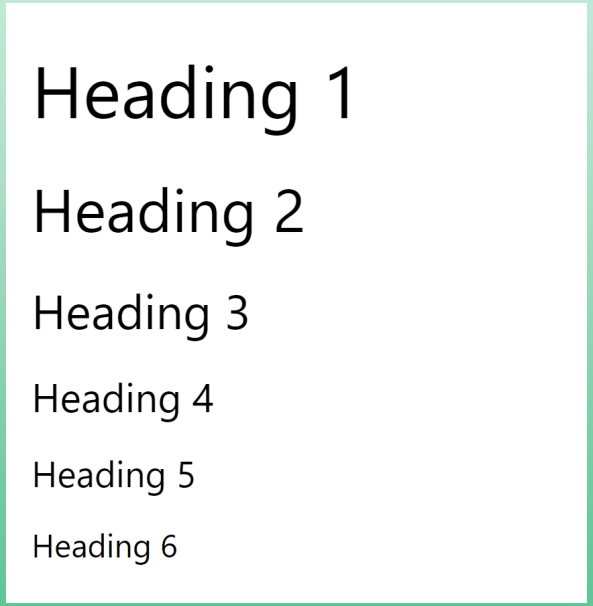
  <FONT COLOR=“red”>….
  </FONT>
- The color can also be defined using hexadecimal representation (Example: #ffffff)
- These attributes can be combined to change the font, size, and color of the text all at once; For example

<FONT SIZE=4 FACE=“Courier” COLOR=“red”> ….
</FONT>

Headings

- Web pages are typically organized into sections with headings; To create a heading use the expression <Hn>….</Hn> where n is a number between 1 and 7
- In this case, the 1 corresponds to the largest size heading while the 7 corresponds to the smallest size
- Search engines use the headings to index the structure and content of your web pages.

Headings



Headings - Bigger Headings

You can specify the size for any heading with the style attribute, using the CSS font-size property

<h1 style="font-size:60px;">Heading 1</h1>

Aligning Text

- The ALIGN attribute can be inserted in the <P> and <Hn> tags to right justify, center, or left justify the text
- For example, <H1 ALIGN=”CENTER”> The New York Times </H1> would create a centered heading of the largest size

HTML Styles (style attribute)

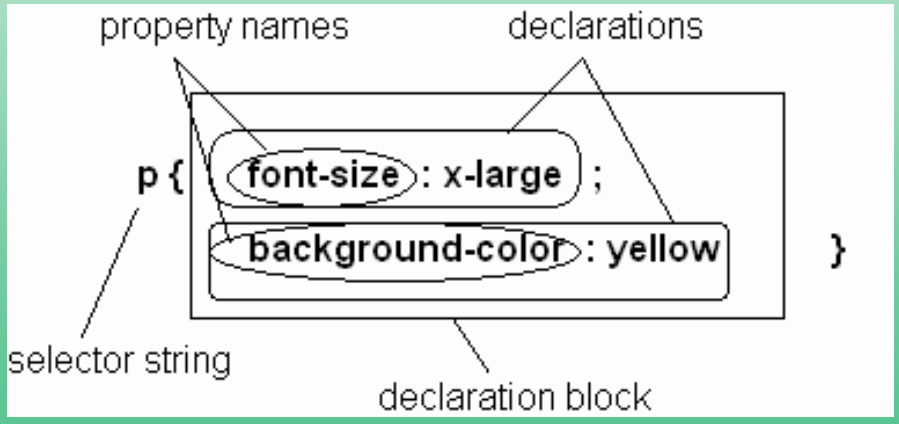
- Setting the style of an HTML element, can be done with the style attribute.
- The HTML style attribute has the following syntax:

<tagname style="property:value;">

The property is a CSS property. The value is a CSS value.

HTML Styles (CSS syntax)

Parts of a style rule (or statement)



CSS - Selector

Single element type:

```
p { font-size:smaller; letter-spacing:1em}
```

Multiple element types:

```
h1,h2,h3,h4,h5 { background-color : purple}
```

All element types:

```
* {font-weight : bold}
```

Specific elements by id :

```
#p1,#p3 { background-color : aqua}
```

Class selector:

```
.c1 { background-color:red}
```

CSS box model….

- The “CSS box model“ is a set of rules that define how every web page on the Internet is rendered.
- CSS treats each element in your HTML document as a “box” with a bunch of different properties that determine where it appears on the page.

CSS box Block and Inline elements

- All the HTML elements that we've been working will have a default type of box. For instance,
  - <h1> and <p> are block-level elements,
  - and are inline elements.

CSS box Block and Inline elements

- Block boxes always appear below the previous block element.
- The width of block boxes is set automatically based on the width of its parent container. In this case, our blocks are always the width of the browser window.
- The default height of block boxes is based on the content it contains. When you narrow the browser window, the <h1> gets split over two lines, and its height adjusts accordingly.

CSS box Block and Inline elements

- Inline boxes don’t affect vertical spacing. They’re not for determining layout—they’re for styling stuff inside of a block.
- The width of inline boxes is based on the content it contains, not the width of the parent element.

Changing box behavior

- Default box type of HTML elements can be changed with the CSS display property.

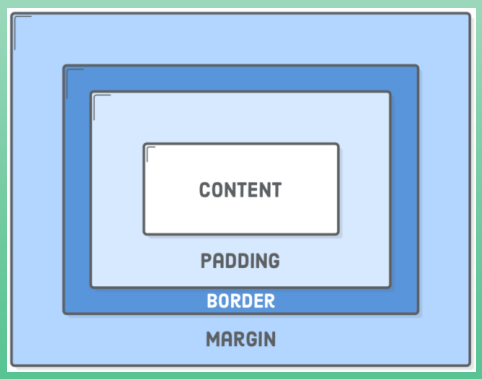
```
display : block;
display : inline;
```

- For example, if you want to make and elements blocks instead of inline elements type

```
em, strong {
    background-color: #B2D6FF;
    display: block;
}
```

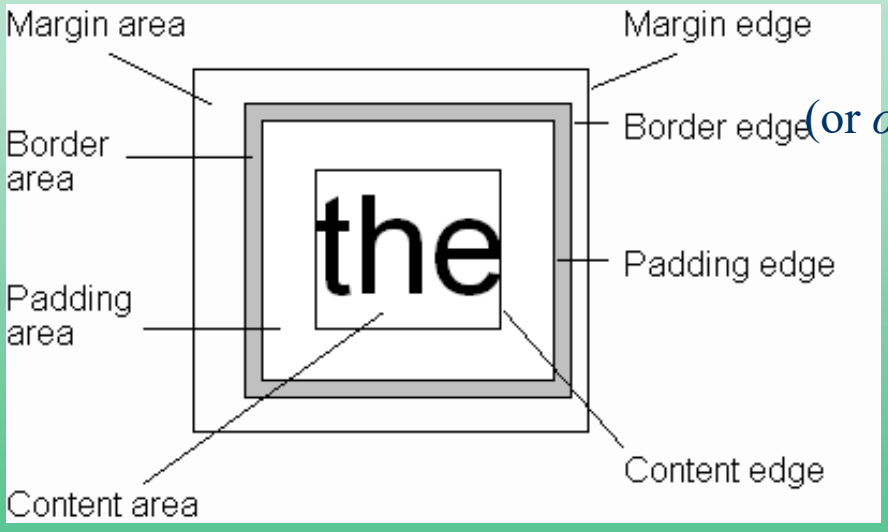
Content, padding, border, and margin

- Content – The text, image, or other media content in the element.
- Padding – The space between the box's content and its border.
- Border – The line between the box's padding and margin.
- Margin – The space between the box and surrounding boxes.

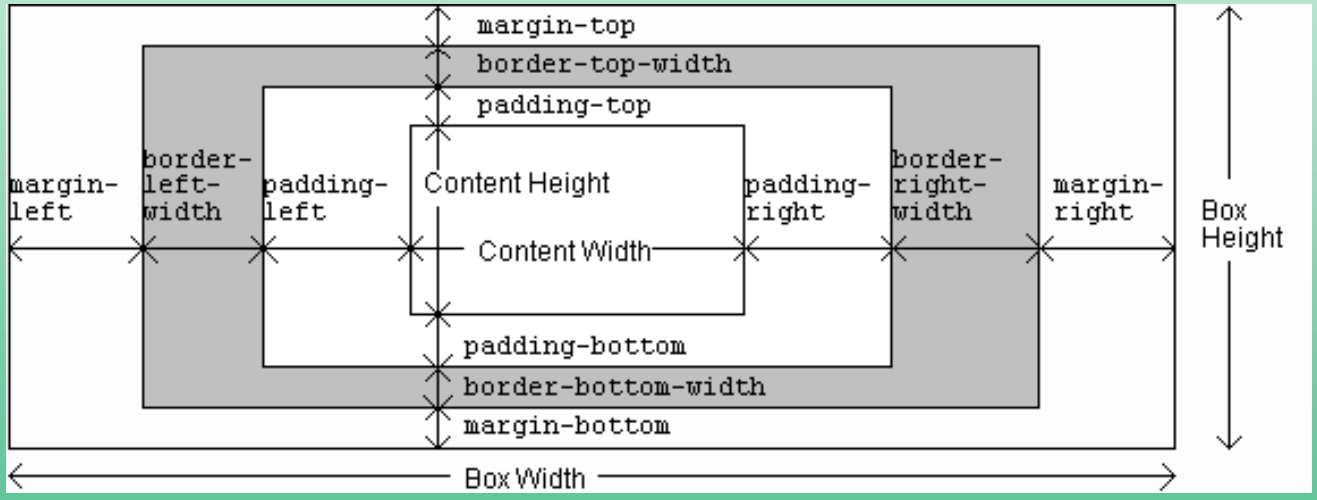


HTML Styles – CSS box model

Every rendered element occupies a box:

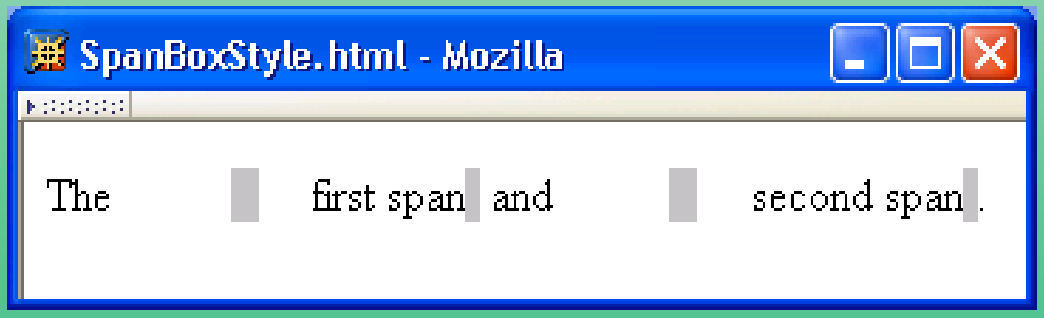


HTML Styles – CSS box model



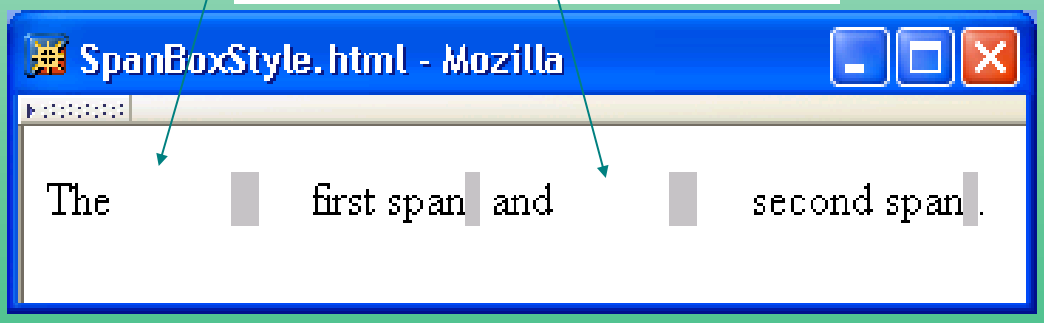
CSS Box Model

```
span { margin-left: 1cm;
       border-left-width: 10px;
       border-left-color: silver;
       border-left-style: solid;
       padding-left: 0.5cm;
       border-right-width: 5px;
       border-right-color: silver;
       border-right-style: solid }
```



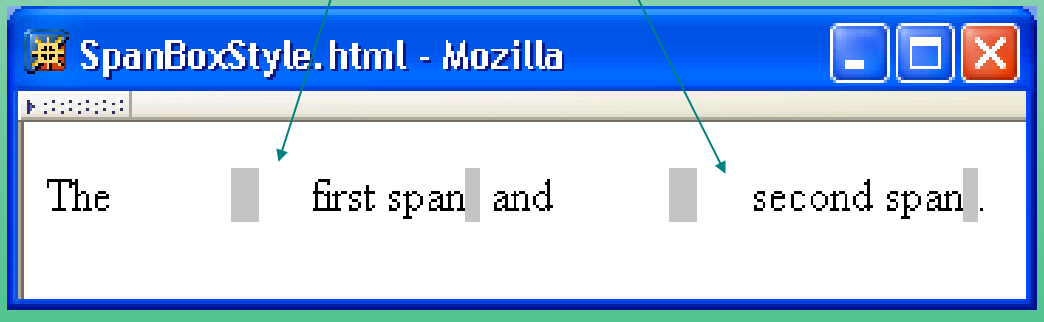
CSS Box Model

```
span { margin-left: 1cm;
       border-left-width: 10px;
       border-left-color: silver;
       border-left-style: solid;
       padding-left: 0.5cm;
       border-right-width: 5px;
       border-right-color: silver;
       border-right-style: solid }
```



CSS Box Model

```
span { margin-left: 1cm;
       border-left-width: 10px;
       border-left-color: silver;
       border-left-style: solid;
       padding-left: 0.5cm;
       border-right-width: 5px;
       border-right-color: silver;
       border-right-style: solid }
```



CSS Box Model

TABLE 3.9: Basic CSS style properties associated with the box model.

Property	Values
`padding-{top,right,bottom,left}`	CSS length (Sec. 3.6.2).
`padding`	One to four length values (see text).

CSS Box Model

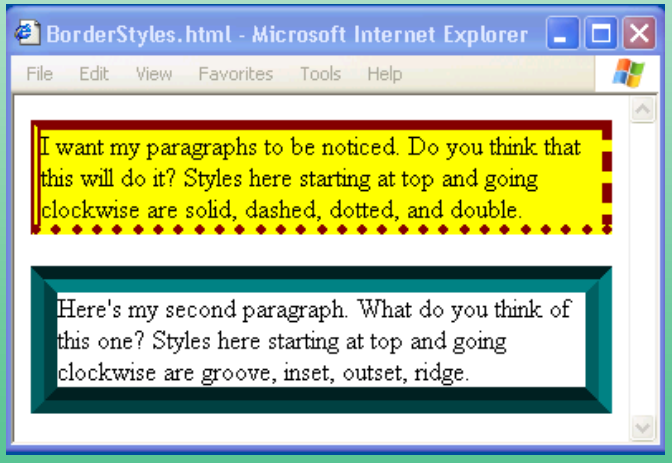
TABLE 3.9: Basic CSS style properties associated with the box model.

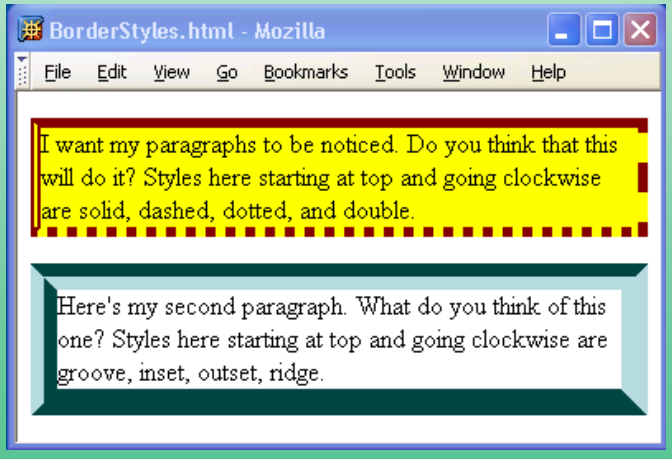
`border-{top,right,bottom,left}-width`	`thin`, `medium` (initial value), `thick`, or a length.
`border-width`	One to four `border-*-width` values.
`border-{top,right,bottom,left}-color`	Color value. Initial value is value of element's `color` property.
`border-color`	`transparent` or one to four `border-*-color` values.

CSS Box Model

TABLE 3.9: Basic CSS style properties associated with the box model.

`border-{top,right,bottom,left}-style`	`none` (initial value), `hidden`, `dotted`, `dashed`, `solid`, `double`, `groove`, `ridge`, `inset`, `outset`.
`border-style`	One to four `border-*-style` values.





CSS Box Model

TABLE 3.9: Basic CSS style properties associated with the box model.

`border-{top,right,bottom,left}`	One to three values (in any order) for `border-*-width`, `border-*-color`, and `border-*-style`. Initial values are used for any unspecified values.
`border`	One to three values; equivalent to specifying given values for each of `border-top`, `border-right`, `border-bottom`, and `border-left`.
`margin-{top,right,bottom,left}`	`auto` (see text) or length.
`margin`	One to four `margin-*` values.

CSS Box Model

If multiple declarations apply to a property, the last declaration overrides earlier specifications

```
{ border: 15px solid;
  border-left: 30px inset red;
  color: blue }
```

Left border is 30px wide, inset style, and red

Position

The position CSS property sets how an element is positioned in a document.

static

The element is positioned according to the normal flow of the document. The top, right, bottom, left, and z-index properties have no effect. This is the default value.

Position

relative

The element is positioned according to the normal flow of the document, and then offset relative to itself based on the values of top, right, bottom, and left. The offset does not affect the position of any other elements; thus, the space given for the element in the page layout is the same as if position were static.

Position

absolute

The element is removed from the normal document flow, and no space is created for the element in the page layout. It is positioned relative to its closest positioned ancestor, if any; otherwise, it is placed relative to the initial containing block. Its final position is determined by the values of top, right, bottom, and left.

Position

fixed

The element is removed from the normal document flow, and no space is created for the element in the page layout. It is positioned relative to the initial containing block established by the viewport,

Position

sticky

The element is positioned according to the normal flow of the document, and then offset relative to its nearest scrolling ancestor and containing block (nearest block-level ancestor), including table-related elements, based on the values of top, right, bottom, and left. The offset does not affect the position of any other elements.

Handling different text directions

Writing modes

A writing mode in CSS refers to whether the text is running horizontally or vertically. The writing-mode property lets us switch from one writing mode to another.

```
h1 {
  writing-mode: vertical-rl;
}
```

Writing modes

The three possible values for the writing-mode property are:

horizontal-tb: Top-to-bottom block flow direction. Sentences run horizontally.

vertical-rl: Right-to-left block flow direction. Sentences run vertically.

vertical-lr: Left-to-right block flow direction. Sentences run vertically.

HTML Styles - Background Color

- The CSS background-color property defines the background color for an HTML element.
- This example sets the background color for a page to powderblue:

```
<body style="background-color:powderblue;">
```

HTML Styles - Text Color

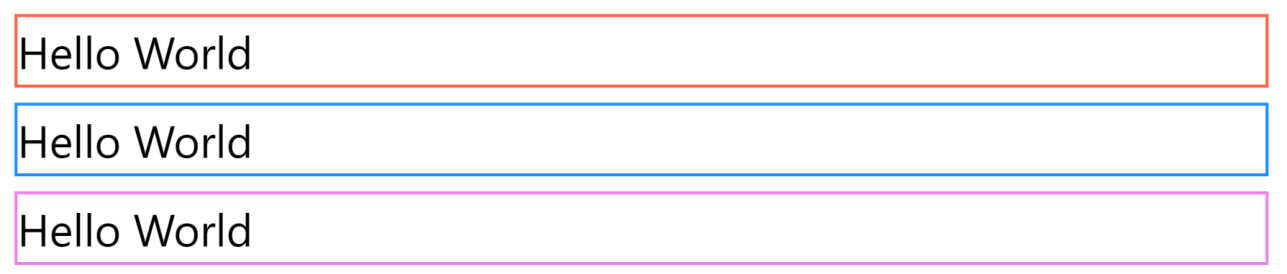
The CSS color property defines the text color for an HTML element:

- Example

```
<h1 style="color:blue;">This is a heading</h1>
<p style="color:red;">This is a paragraph.</p>
```

HTML Styles - Border Color

- You can set the color of borders:



- Example

```
<h1 style="border:2px solid Tomato;">Hello World</h1>
<h1 style="border:2px solid DodgerBlue;">Hello World</h1>
<h1 style="border:2px solid Violet;">Hello World</h1>>
```

HTML Styles - Fonts

- The CSS font-family property defines the font to be used for an HTML element:

Example

```
<h1 style="font-family:verdana;">This is a heading</h1>
<p style="font-family:courier;">This is a paragraph.</p>
```

HTML Styles - Text Size

- The CSS font-size property defines the text size for an HTML element:

- Example

```
<h1 style="font-size:300%;">This is a heading</h1>
<p style="font-size:160%;">This is a paragraph.</p>
```

HTML Styles - Text Alignment

- The CSS text-align property defines the horizontal text alignment for an HTML element:

- Example

```
<h1 style="text-align:center;">Centered Heading</h1>
<p style="text-align:center;">Centered paragraph.</p>
```

HTML Styles - Padding

The CSS padding property defines a padding (space) between the text and the border:

Example

```
p {
  border: 1px solid powderblue;
  padding: 30px;
}
```

HTML Styles - Margin

CSS

The CSS margin property defines a margin (space) outside the border:

Example

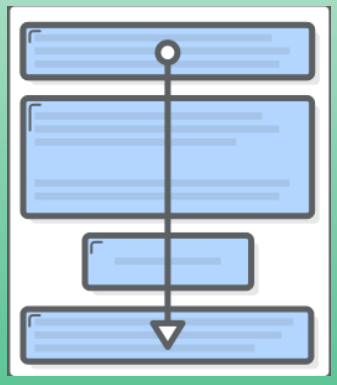
```
p {
  border: 1px solid powderblue;
  margin: 50px;
}
```

Generic boxes

- Generic box are inserted for the sake of styling a web page.
- and tags are used to insert generic box.
- Both and are “container” elements don’t have any affect on the semantic structure of an HTML document.
- They do, however, provide a hook for adding CSS styles to arbitrary sections of a web page.

CSS floats

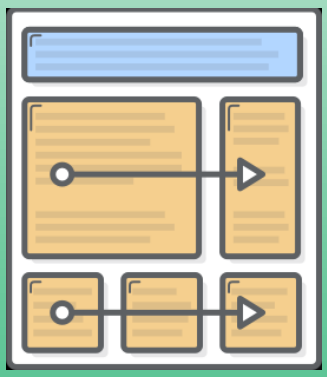
- Block elements always appeared vertically one after another, effectively limiting us to a single-column layout.



Vertical flow

CSS floats

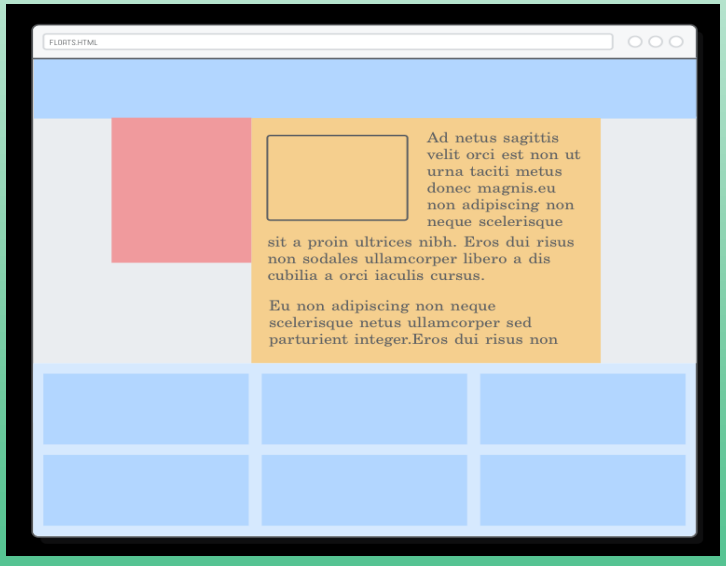
- "Floats" let you put block-level elements side-by-side instead of on top of each other.
- It lets us build all sorts of layouts, including sidebars, multi-column pages, grids, and magazine-style articles with text flowing around an image.



Horizontal flow

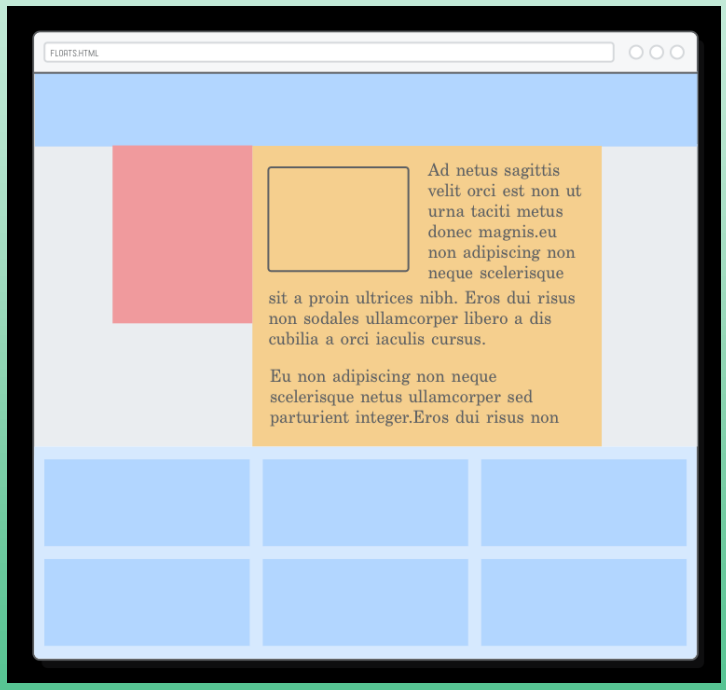
CSS floats

- Lets create the following page



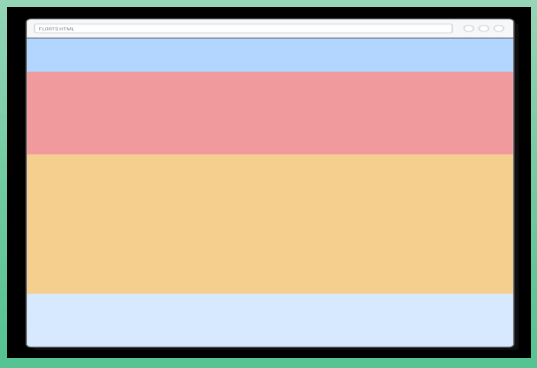
CSS floats

```
<div class='page'>
  <div class='menu'>Menu</div>
  <div class='sidebar'>Sidebar</div>
  <div class='content'>Content</div>
  <div class='footer'>Footer</div>
</div>
```



CSS floats

```
<div class='page'>
  <div class='menu'>Menu</div>
  <div class='sidebar'>Sidebar</div>
  <div class='content'>Content</div>
  <div class='footer'>Footer</div>
</div>
```



```
.menu {
  height: 100px;
  background-color: #B2D6FF;    /* Medium blue */
}
.sidebar {
  height: 300px;
  background-color: #F09A9D;    /* Red */
}
.content {
  height: 500px;
  background-color: #F5CF8E;    /* Yellow */
}
.footer {
  height: 200px;
  background-color: #D6E9FE;    /* Light blue */
}
```

- Each block-level element fills 100% of its parent elements's width
- They appear vertically one after another.

CSS floats – Shrinking with width

```
.menu {
  height: 100px;
  background-color: #B2D6FF;    /* Medium blue */
}
.sidebar {
  width: 200px;              /* Add this */
  height: 300px;
  background-color: #F09A9D;
}
.content {
  height: 500px;
  background-color: #F5CF8E;    /* Yellow */
}
.footer {
  height: 200px;
  background-color: #D6E9FE;    /* Light blue */
}
```

- The sidebar element gets narrower, but the rest of the boxes stay in the exact same position.
- All the blocks are still rendered vertically one after another.
- This is the behavior we'll be changing with floats.

CSS floats – Floating an element

- The CSS float property gives us control over the horizontal position of an element.
- By "floating" the sidebar to the left, we're telling the browser to align it to the left side of the page.

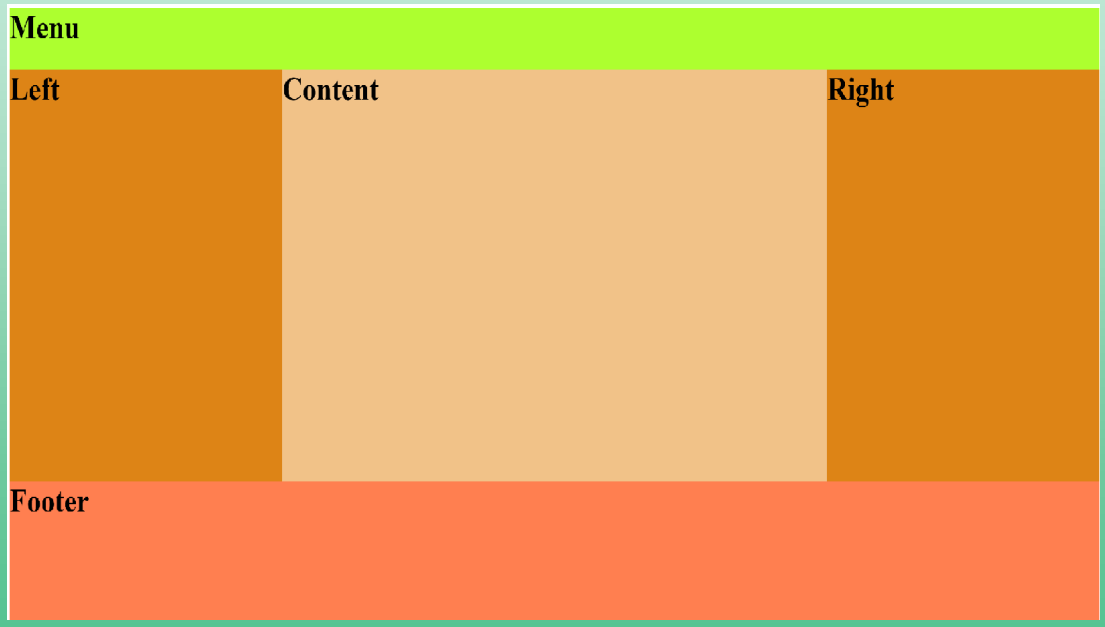
```
.sidebar {
        float: left;                /* Add this */
    width: 200px;
    height: 300px;
    background-color: #F09A9D;
}
```

CSS floats – Floating an element

```
.sidebar {
        float: left;
        width: 200px;
        height: 300px;
        background-color: #F09A9D;
}
```

- However, this doesn't just align the sidebar—it also tells surrounding elements that they can flow around the sidebar instead of beginning underneath it.
- It's as if the sidebar is inside the .content block.
- Any HTML markup in .content would wrap around the sidebar's box.
- This gives us a magazine-style layout.

CSS floats – Float example



CSS floats – Float example

```
<div class="container">
   <div class="menu"><h1>Menu</h1></div>
   <div class="section">
      <div class="leftsidebar"><h1>Left</h1></div>
      <div class="content"><h1>Content</h1></div>
      <div class="rightsidebar"><h1>Right</h1></div>
   </div>
   <div class="footer"><h1>Footer</h1></div>
</div>
```

CSS floats – Float example

```
.menu{
    height : 60px;
    background-color: greenyellow;
}
.section{
    height: 400px;
}
.leftsidebar{
    float: left;
    height: 400px;
    background-color: rgb(221, 132, 22);
    width: 25%;
}
.content{
    float :left;
    height: 400px;
    width: 50%;
    background-color: rgb(241, 194, 136);
}
```

```
.rightsidebar {
    float: left;
    height: 400px;
    width: 25%;
    background-
color: rgb(221, 132, 22);
}

.footer {
    height: 200px;
    background-color: coral;
}
```

CSS floats – Floating an element

- You can align block-level elements to left/right alignment and auto-margins for center alignment.
- This only applies to block boxes.
- Inline boxes are aligned with the text-align property

CSS floats – Floating an element

- You can align block-level elements to left/right alignment and auto-margins for center alignment.
- This only applies to block boxes.
- Inline boxes are aligned with the text-align property

CSS Flexbox Layout Module

- Flexbox (flexible box) is a layout mode of CSS3.
- Using this mode, you can easily create layouts for complex applications and web pages.
- Flexbox layout gives complete control over the direction, alignment, order, size of the boxes.

Features of Flexbox

- **Direction** − You can arrange the items on a web page in any direction such as left to right, right to left, top to bottom, and bottom to top.
- **Order** − Using Flexbox, you can rearrange the order of the contents of a web page.
- **Wrap** − In case of inconsistent space for the contents of a web page (in single line), you can wrap them to multiple lines (both horizontally) and vertically.
- **Alignment** − Using Flexbox, you can align the contents of the webpage with respect to their container.
- **Resize** − Using Flexbox, you can increase or decrease the size of the items in the page to fit in available space.

Flexbox Container

- To start using the Flexbox model, you need to first define a flex container.

```
.container
{
  display: flex; /* or inline-flex */
}
```

- This property accepts two values

  •**flex** − Generates a block level flex container.

  •**inline-flex** − Generates an inline flex container box.

Block level flex container

```
.container
{
display: flex;          // It occupies the full width of the parent container
}
```

CSS

```
.box1{background:green;}
.box2{background:blue;}
.box3{background:red;}
.box4{background:magenta;}
.box5{background:yellow;}
.box6{background:pink;}

.container{
  display:flex;
}
.box{
  font-size:35px;
  padding:15px;
}
```

HTML

```
<div class = "container">
<div class = "box box1">One</div>
<div class = "box box2">two</div>
<div class = "box box3">three</div>
<div class = "box box4">four</div>
<div class = "box box5">five</div>
<div class = "box box6">six</div>
</div>
```

Inline-flex container

```
.container
{
display: inline-flex;      // It occupies the place required for the content.
}
```

CSS

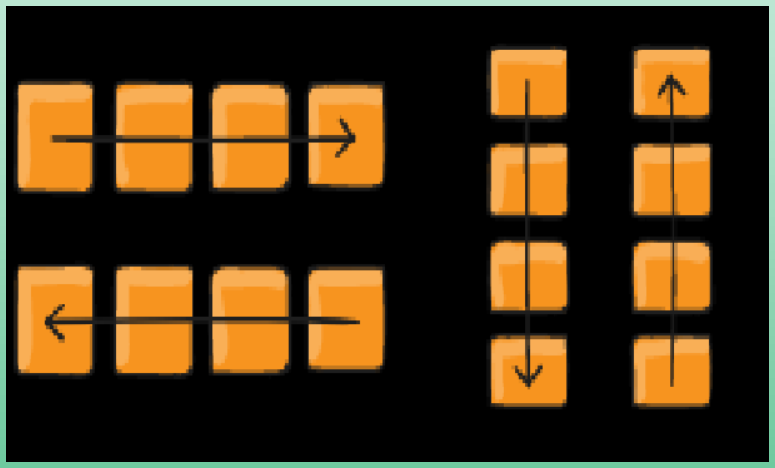
```
.box1{background:green;}
.box2{background:blue;}
.box3{background:red;}
.box4{background:magenta;}
.box5{background:yellow;}
.box6{background:pink;}

.container{
  display:inline-flex;
  border : 3px solid black;
}
.box{
  font-size:35px;
  padding:15px;
}
```

HTML

```
<div class = "container">
<div class = "box box1">One</div>
<div class = "box box2">two</div>
<div class = "box box3">three</div>
<div class = "box box4">four</div>
<div class = "box box5">five</div>
<div class = "box box6">six</div>
</div>
```

Flex-direction

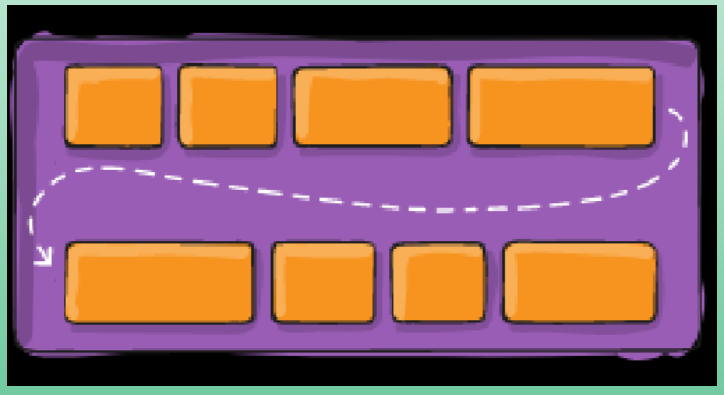
- The **flex-direction** property is used to specify the direction in which the elements of flex container (flex-items) are needed to be placed.



```
.container {
flex-direction: row | row-reverse | column | column-reverse;
}
```

Flex-wrap

- By default, flex items will all try to fit onto one line.
- You can change that and allow the items to wrap as needed with this property.

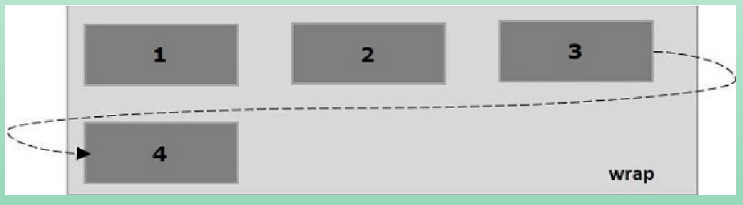


```
.container {
flex-wrap: nowrap | wrap | wrap-reverse;
}
```

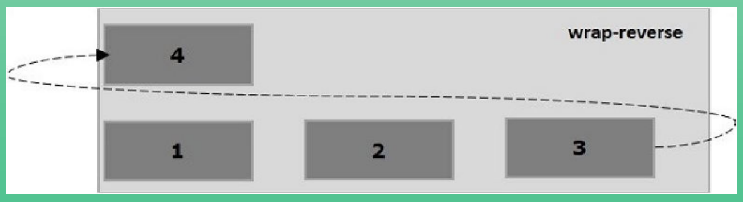
Flex-wrap

1. nowrap - all flex items will be on one line

2. wrap - flex items will wrap onto multiple lines, from top to bottom.

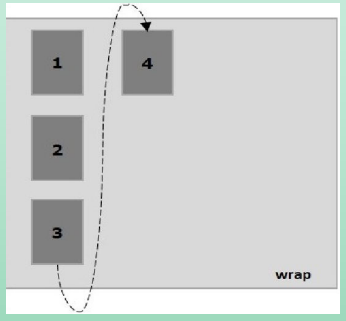


3. wrap-reverse - flex items will wrap onto multiple lines from bottom to top.

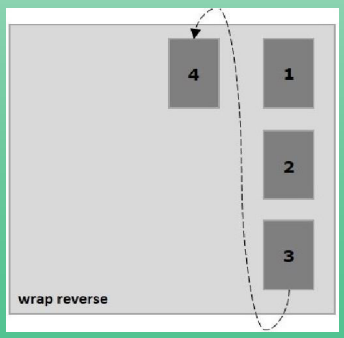


Flex-wrap

4. flex-direction : column and flex-wrap : wrap



5. flex-direction : column and flex-wrap : wrap-reverse



Flex-flow

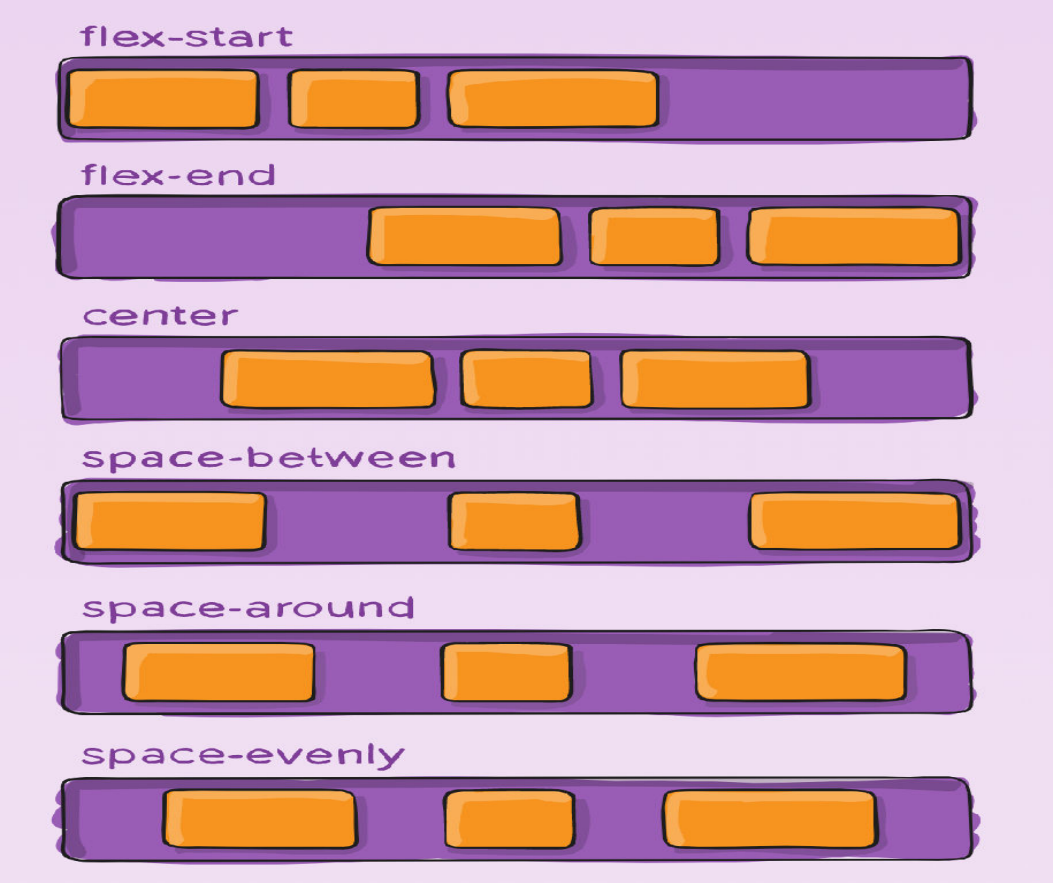
- This is a shorthand for the flex-direction and flex-wrap properties, which together define the flex container's main and cross axes. The default value is row nowrap.

```
.container {
        flex-flow: column wrap;
}
```

Flexbox – justifying contents

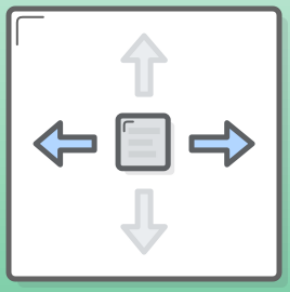
- This defines the alignment along the **main axis**.
- It helps to distribute extra free space leftover when either all the flex items on a line are inflexible, or are flexible but have reached their maximum size.
- It also exerts some control over the alignment of items when they overflow the line.
- **.container {**
  justify-content: flex-start | flex-end | center | space-between | space-around | space-evenly;
  }

Flexbox – justifying contents

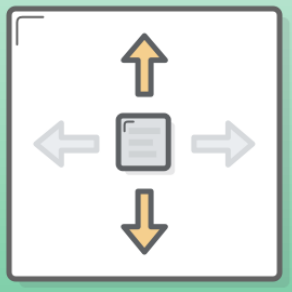


Flexbox – align-items

- The **align-items** property is same as **justify content**. But here, the items were aligned across the **cross access** (vertically).



JUSTIFY-CONTENT

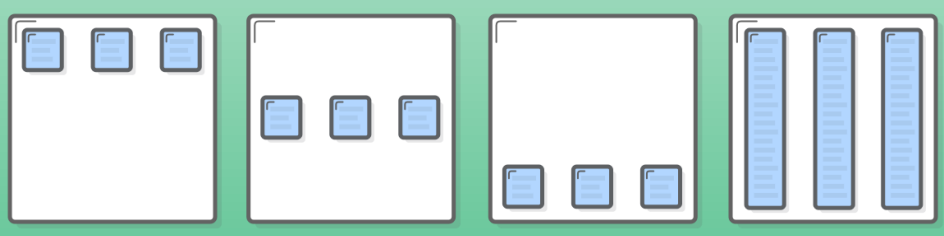


ALIGN-ITEMS

Flexbox – align-items

- The **align-items** property is same as **justify content**. But here, the items were aligned across the cross access (vertically).

```
.container {
  align-items: stretch | flex-start | flex-end | center | baseline;
}
```



FLEX-START CENTER FLEX-END STRETCH

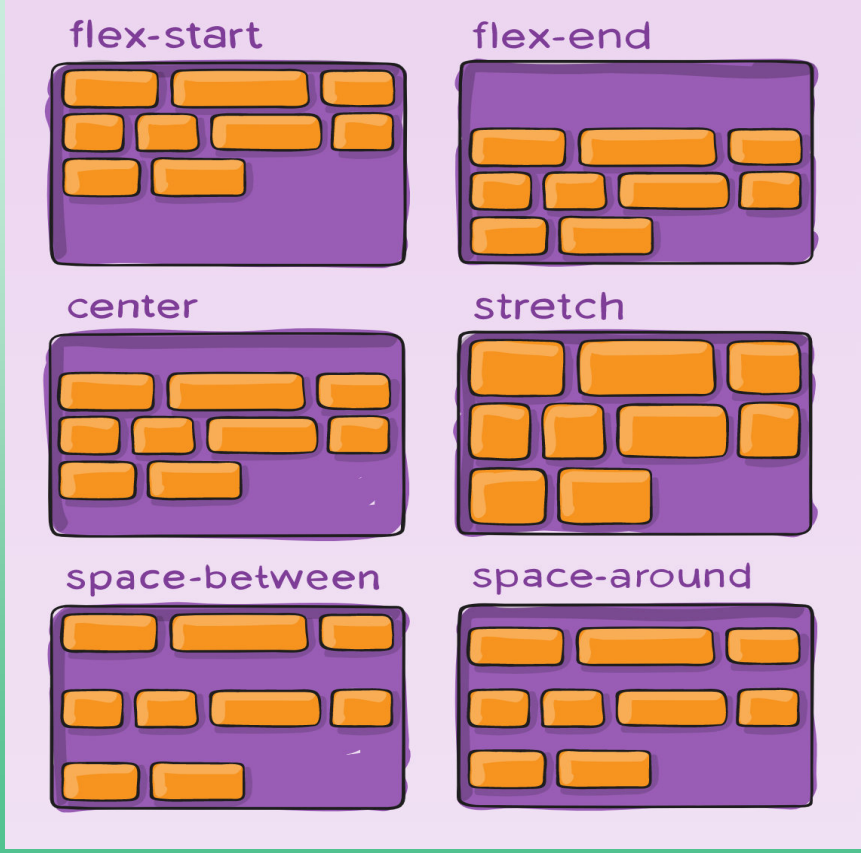
Baseline

Flexbox – align-contents

- In case the flex-container has multiple lines (when, flex-wrap: wrap), the align-content property defines the alignment of each line within the container.
- This property has no effect when there is only one line of flex items.

```
.container {
        align-content: flex-start | flex-end | center | space-between |
                       space-around | space-evenly | stretch;
}
```

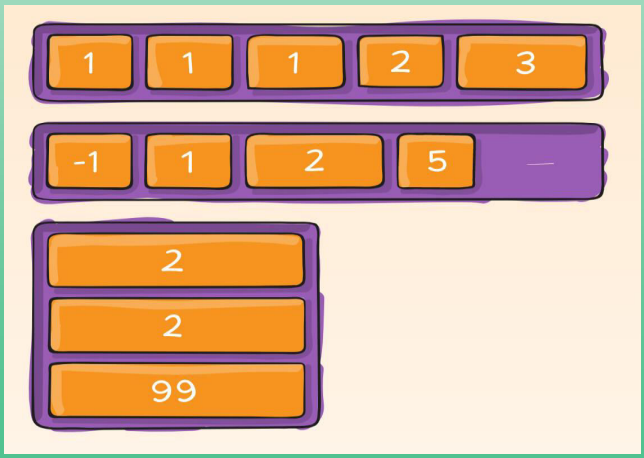
Flexbox – align-contents



Flexbox – flex-order

- The **flex-order** property is used to define the order of the flexbox item.

```
.item {
    order: 5; /* default is 0 */
}
```



Flexbox – flex-grow

- This defines the ability for a flex item to grow if necessary.
- It dictates what amount of the available space inside the flex container the item should take up.
- If all items have flex-grow set to 1, the remaining space in the container will be distributed equally to all children.
- If one of the children has a value of 2, the remaining space would take up twice as much space as the others (or it will try to, at least).

```
.item {
  flex-grow: 2; /* default 0 */
}
```

Flexbox – flex-shrink

- This defines the ability for a flex item to shrink if necessary.
- In case there is not enough space in the container, it specifies how much a flex-item should shrink.

```
.item {
  flex-shrink: 3; /* default 1 */
}
```

Flexbox – align-self

- This property is similar to **align-items**, but here, it is applied to individual flex items.

```
.item {
    align-self: auto | flex-start | flex-end | center | baseline | stretch;
}
```

Inserting Images

- Type <IMG SRC = “image.ext”>, where image.ext indicates the location of the image file
- The WIDTH=n and HEIGHT=n attributes can be used to adjust the size of an image
- The attribute BORDER=n can be used to add a border n pixels thick around the image

Alternate Text

- Some browsers don’t support images. In this case, the ALT attribute can be used to create text that appears instead of the image.
- Example:

  <IMG SRC=“satellite.jpg” ALT = “Picture of satellite”>

Links

- A link lets you move from one page to another, play movies and sound, send email, download files, and more….
- A link has three parts: a **destination**, a **label**, and a **target**
- To create a link type

  <A HREF=“page.html”> label </A>

Anatomy of a Link

<A HREF="page.html"> label </A>

- In the above link, "page.html" is the destination. The destination specifies the address of the Web page or file the user will access when he/she clicks on the link.
- The label is the text that will appear underlined or highlighted on the page

Example: Links

- To create a link to CNN, I would type:
  <A HREF="http://www.cnn.com">CNN</A>
- To create a link to MIT, I would type:
  <A HREF="http://www.mit.edu">MIT</A>

Changing the Color of Links

- The LINK, VLINK, and ALINK attributes can be inserted in the <BODY> tag to define the color of a link
  - LINK defines the color of links that have not been visited
  - VLINK defines the color of links that have already been visited
  - ALINK defines the color of a link when a user clicks on it

Changing the Color of Links

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

- An unvisited link is underlined and blue
- A visited link is underlined and purple
- An active link is underlined and red

Changing the Color of Links

```
a:link {
  color: green;
  background-color: transparent;
  text-decoration: none;
}

a:visited {
  color: pink;
  background-color: transparent;
  text-decoration: none;
}
```

Changing the Color of Links

```
a:hover {
  color: red;
  background-color: transparent;
  text-decoration: underline;
}

a:active {
  color: yellow;
  background-color: transparent;
  text-decoration: underline;
}
```

Changing the target of Links

- The target attribute specifies where to open the linked document.
- The target attribute can have one of the following values:

_blank - Opens the linked document in a new window or tab
_self - Opens the linked document in the same window/tab as it was clicked (this is default)
_parent - Opens the linked document in the parent frame
_top - Opens the linked document in the full body of the window

Using Links to Send Email

- To create a link to an email address, type

```
<a href="mailto:email_address"> Label</a>
```

- For example, to create a link to send email to myself, I would type:

```
<a href="mailto: savitasoft@gmail.com">email SSCE</a>
```

Anchors

- Anchors enable a user to jump to a specific place on a Web site
- Two steps are necessary to create an anchor.
  - First you must create the anchor itself.
  - Then you must create a link to the anchor from another point in the document.

Anchors

- To create the anchor itself, type

  label

at the point in the Web page where you want the user to jump to.

- To create the link, type

  label

at the point in the text where you want the link to appear

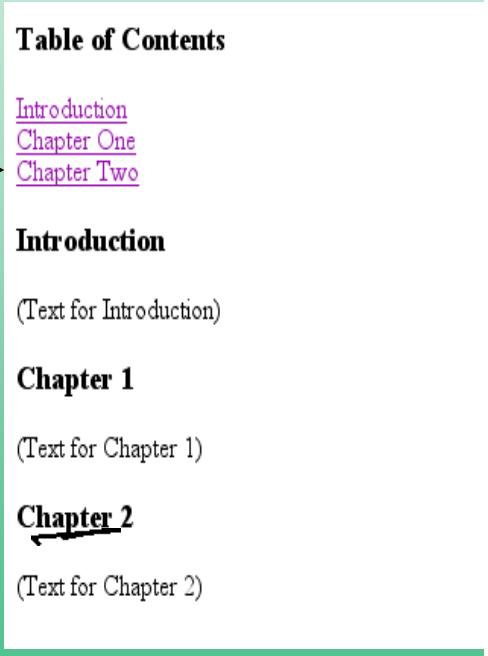
Example: Anchor

Chapter Two

Link

Chapter 2

Anchor



Ordered Lists

- Ordered lists are a list of numbered items.
- To create an ordered list, type:

```
<OL>
    <LI> This is step one. </LI
    <LI> This is step two. </LI>
    <LI> This is step three </LI
</OL>
```

Here's how it would look on the Web:

1. This is step one.
2. This is step two.
3. This is step three.

More Ordered Lists….

- The TYPE=x attribute allows you to change the the kind of symbol that appears in the list.
  - A is for capital letters
  - a is for lowercase letters
  - I is for capital roman numerals
  - i is for lowercase roman numerals

Unordered Lists

- An unordered list is a list of bulleted items
- To create an unordered list, type:

```
<UL>
    <LI> First item in list </LI>
    <LI> Second item in list </LI>
    <LI> Third item in list</LI>
</UL>
```

Here's how it would look on the Web:

- First item in list
- Second item in list
- Third item in list

More Unordered Lists...

- The TYPE=shape attribute allows you to change the type of bullet that appears
  - *circle* corresponds to an empty round bullet
  - *square* corresponds to a square bullet
  - *disc* corresponds to a solid round bullet; this is the default value

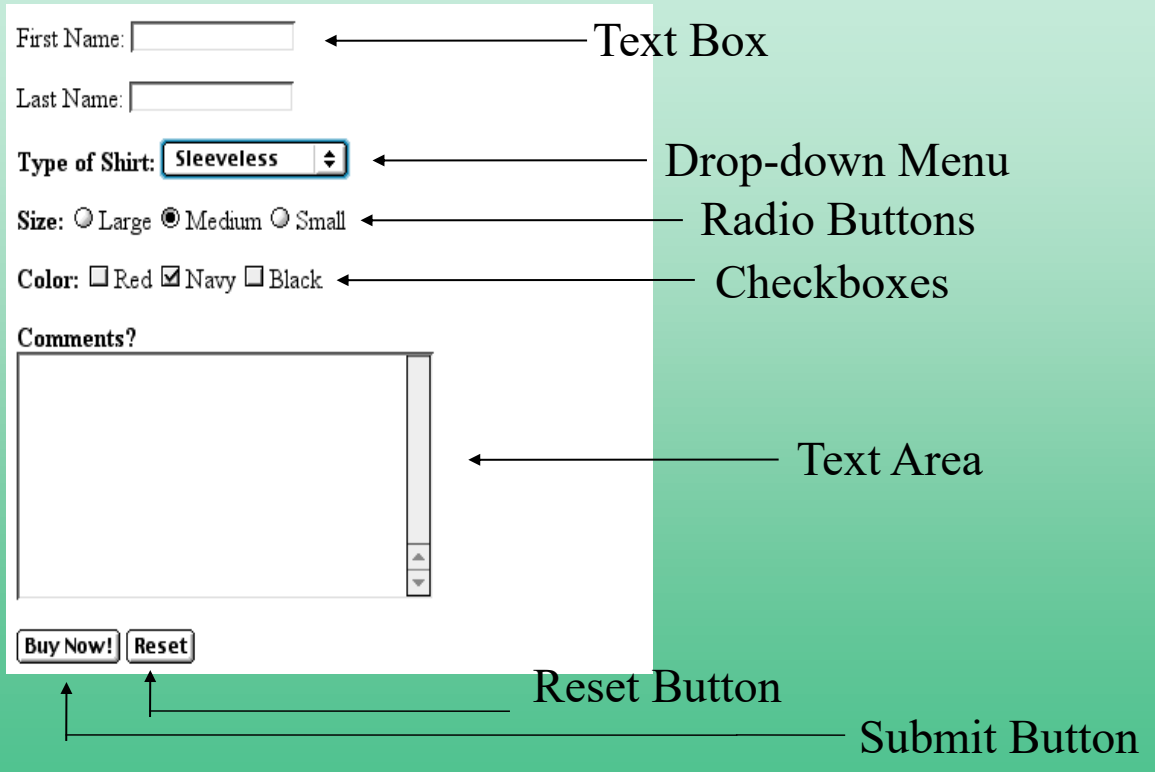
Forms

- What are forms?
  - An HTML form is an area of the document that allows users to enter information into fields.
  - A form may be used to collect personal information, opinions in polls, user preferences and other kinds of information.

Forms

- There are two basic components of a Web form: the shell, the part that the user fills out, and the script which processes the information
- HTML tags are used to create the form shell.
- Using HTML you can create text boxes, radio buttons, checkboxes, drop-down menus, and more...

Example: Form



The Form Shell

- A form shell has three important parts:
  - the <FORM> tag, which includes the address of the script which will process the form
  - the form elements, like text boxes and radio buttons
  - the submit button which triggers the script to send the entered information to the server

Creating the Shell

- To create a form shell, type

  <form method="POST" action="script_url">

  where "script_url" is the address of the script
- Create the form elements
- End with a closing </FORM> tag

Creating Text Boxes

- To create a text box, type

<INPUT TYPE="text" NAME="name" VALUE="value" SIZE="n" MAXLENGTH="n">

- The NAME, VALUE, SIZE, and MAXLENGTH attributes are optional

Text Box Attributes

- The NAME attribute is used to identify the text box to the processing script
- The VALUE attribute is used to specify the text that will initially appear in the text box
- The SIZE attribute is used to define the size of the box in characters
- The MAXLENGTH attribute is used to define the maximum number of characters that can be typed in the box

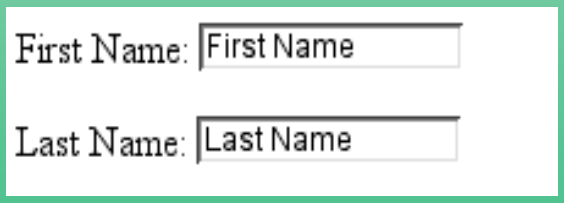
Example: Text Box

First Name: <INPUT TYPE="text" NAME="FirstName" VALUE="First Name" SIZE="20">

Last Name: <INPUT TYPE="text" NAME="LastName" VALUE="Last Name" SIZE="20">

<label for="textbox id">Label</label
<input type="text" name="textboxname" id="textbox id" value="init value"
- placeholder="placeholder" size="width" maxlength="max chars">

Here's how it would look on the Web:



Creating Larger Text Areas

- To create larger text areas, type

```
<TEXTAREA NAME="name" ROWS="n1" COLS="n2" WRAP>
    Default Text
</TEXTAREA>
```

where n1 is the height of the text box in rows and n2 is the width of the text box in characters

- The WRAP attribute causes the cursor to move automatically to the next line as the user types

Example: Text Area

```
<B>Comments?</B>
<BR>
<TEXTAREA NAME="Comments" ROWS=”10” COLS=”50” WRAP>
</TEXTAREA>
```

Creating Radio Buttons

- To create a radio button, type

  <INPUT TYPE=“radio” NAME=“name” VALUE=“data”>Label

  where

  “data” is the text that will be sent to the server if the button is checked and

  “Label” is the text that identifies the button to the user

Example: Radio Buttons

```
<B> Size: </B>
<INPUT TYPE="radio" NAME="Size" VALUE="Large">Large
<INPUT TYPE="radio" NAME="Size" VALUE="Medium">Medium
<INPUT TYPE="radio" NAME="Size" VALUE="Small">Small
```

Creating Checkboxes

- To create a checkbox, type

  <INPUT TYPE="checkbox" NAME="name" VALUE="value">Label

Example: Checkboxes

```
<B> Color: </B>
<INPUT TYPE="checkbox" NAME="Color" VALUE="Red">Red
<INPUT TYPE="checkbox" NAME="Color" VALUE="Navy">Navy
<INPUT TYPE="checkbox" NAME="Color" VALUE="Black">Black
```

Creating Drop-down Menus

- To create a drop-down menu, type

  <SELECT NAME="name" SIZE="n" MULTIPLE>

  Then type

  <OPTION VALUE= "value">Label </OPTION>

- In this case the SIZE attribute specifies the height of the menu in lines and MULTIPLE allows users to select more than one menu option

Example: Drop-down Menu

```
<B>WHICH IS FAVOURITE FRUIT:</B>
<SELECT>
   <OPTION VALUE="MANGOES">MANGOES
   <OPTION VALUE="PAPAYA">PAPAYA
   <OPTION VALUE="GUAVA">GUAVA
   <OPTION VALUE="BANANA"> BANANA
   <OPTION VALUE="PINEAPPLE">PINEAPPLE
</SELECT>
```

Creating a Submit Button

- To create a submit button, type

  <INPUT TYPE="submit">

- If you would like the button to say something other than submit, use the VALUE attribute

- For example,

  <INPUT TYPE="submit" VALUE="Buy Now!">

would create a button that says "Buy Now!"

Creating a Reset Button

- To create a reset button, type

```
<INPUT TYPE="reset"  VALUE="caption">
```

- The VALUE attribute can be used in the same way to change the text that appears on the button

Tables

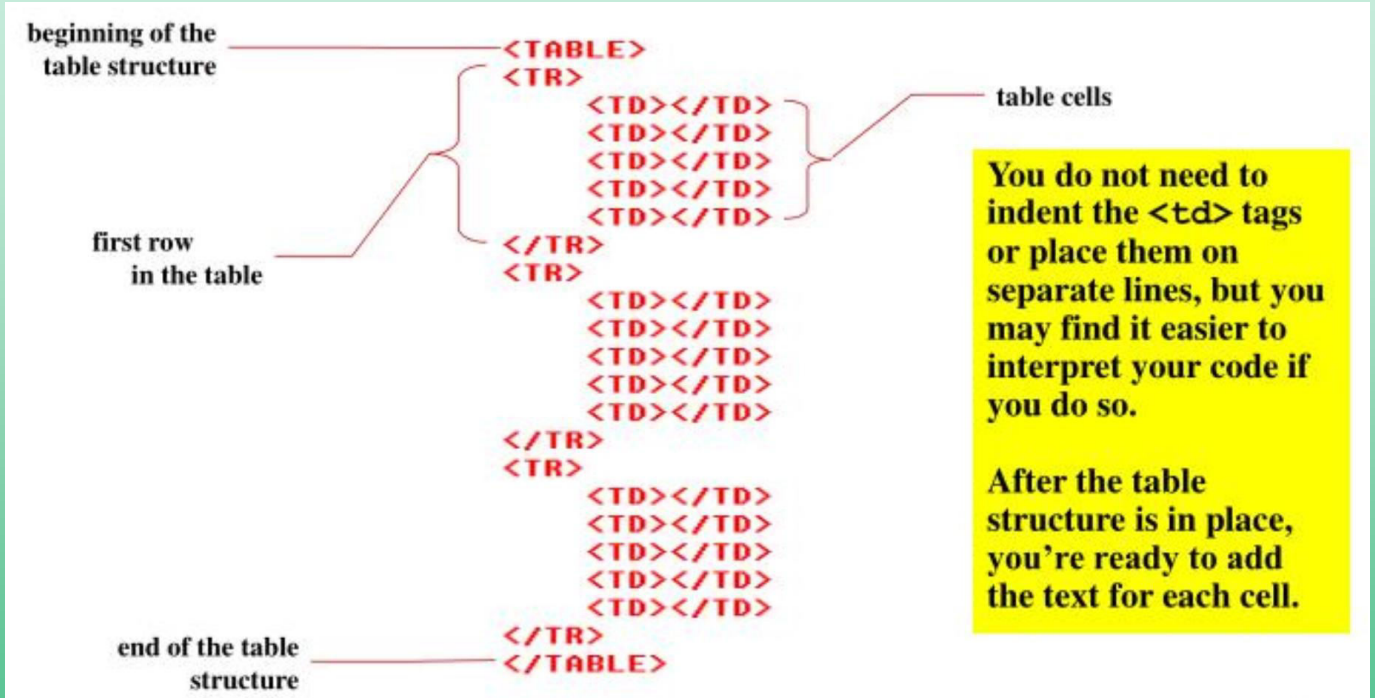
- The HTML tables allow web authors to arrange data like text, images, links, other tables, etc. into rows and columns of cells.
- The **<table>** tag is used to create a table.
- The **<tr>** tag is used to create table rows.
- The **<td>** tag is used to create data cells.

Tables

```
<table border = "1">
        <tr>
          <td>Row 1, Column 1</td>
          <td>Row 1, Column 2</td>
        </tr>

        <tr>
          <td>Row 2, Column 1</td>
          <td>Row 2, Column 2</td>
        </tr>
</table>
```

Row 1, Column 1	Row 1, Column 2
Row 2, Column 1	Row 2, Column 2

Tables



Creating Headings

- HTML provides the <th> tag for table headings.
- Text formatted with the <th> tag is centered within the cell and displayed in a boldface font.
- The <th> tag is most often used for column headings, but you can use it for any cell that you want to contain centered boldfaced text.

Creating Headings

Creating a Table Caption

- HTML allows you to specify a caption for a table.
- The syntax for creating a caption is:

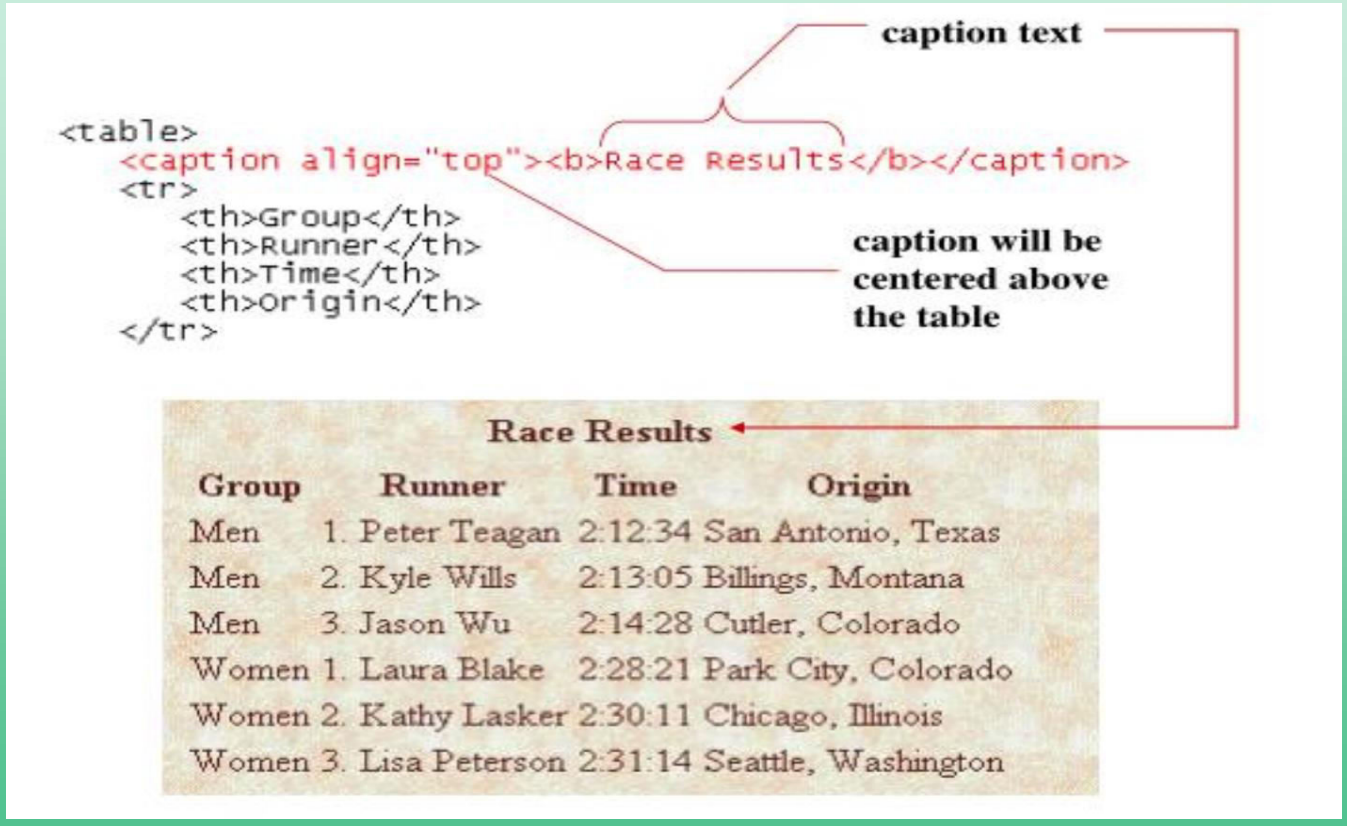
```
<caption align="alignment"
        style="text-align: center; caption-side : top"
        >caption text</caption>
```

- **alignment** indicates the caption placement
  - a value of "*bottom*" centers the caption below the table
  - a value of "top" or "center" centers the caption above the table
  - a value of "left" or "right" place the caption above the table to the left or right

Creating a Table Caption

- Only Internet Explorer supports all caption values.
- Netscape supports only the "top" and "bottom" values.
- The <caption> tag works only with tables, the tag must be placed within the table structure.
- Captions are shown as normal text without special formatting.
- Captions can be formatted by embedding the caption text within other HTML tags. For example, place the caption text within a pair of <b> and <i> tags causes the caption to display as bold and italic provides the <th> tag for table headings.
- Text formatted with the <th> tag is centered within the cell and displayed in a boldface font.

Creating table caption



Modifying the Appearance of a Table

- You can modify the appearance of a table by adding:
  - gridlines
  - borders
  - background color
- HTML also provides tags and attributes to control the placement and size of a table.

Adding a Table Border

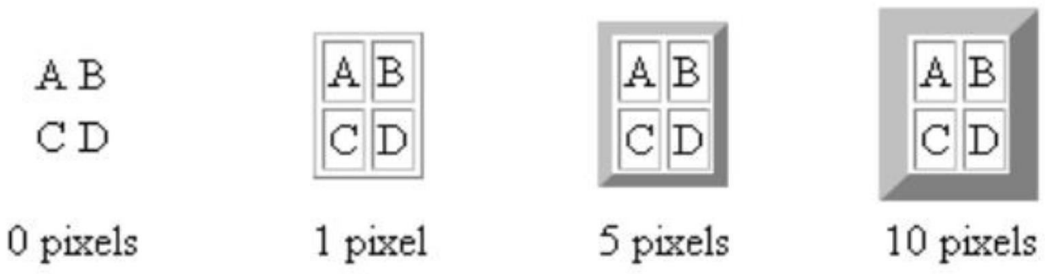
- By default, browsers display tables without table borders.
- A table border can be added using the border attribute to the <table> tag.
- The syntax for creating a table border is:

  <table border="value">

  value is the width of the border in pixels
- The size attribute is optional; if you don't specify a size, the browser creates a table border 1 pixel wide.

Adding a Table Border

This figure shows the effect on a table's border when the border size is varied.



Adding a Table Border

```
<table border="5">
   <caption align="top"><b>Race Results</b></caption>
   <tr>
      <th>Group</th>
      <th>Runner</th>
      <th>Time</th>
      <th>Origin</th>
   </tr>
```

Only the outside border is affected by the border attribute; the internal gridlines are not affected.

Race Results

Group	Runner	Time	Origin
Men	1. Peter Teagan	2:12:34	San Antonio, Texas
Men	2. Kyle Wills	2:13:05	Billings, Montana
Men	3. Jason Wu	2:14:28	Cutler, Colorado
Women	1. Laura Blake	2:28:21	Park City, Colorado
Women	2. Kathy Lasker	2:30:11	Chicago, Illinois
Women	3. Lisa Peterson	2:31:14	Seattle, Washington

Defining Cell Padding

- To control the space between the table text and the cell borders, add the cellpadding attribute to the table tag.
- The syntax for this attribute is:

  <table cellpadding="value">

  value is the distance from the table text to the cell border, as measured in pixels
- the default cell padding value is 1 pixel

Defining Cell Padding

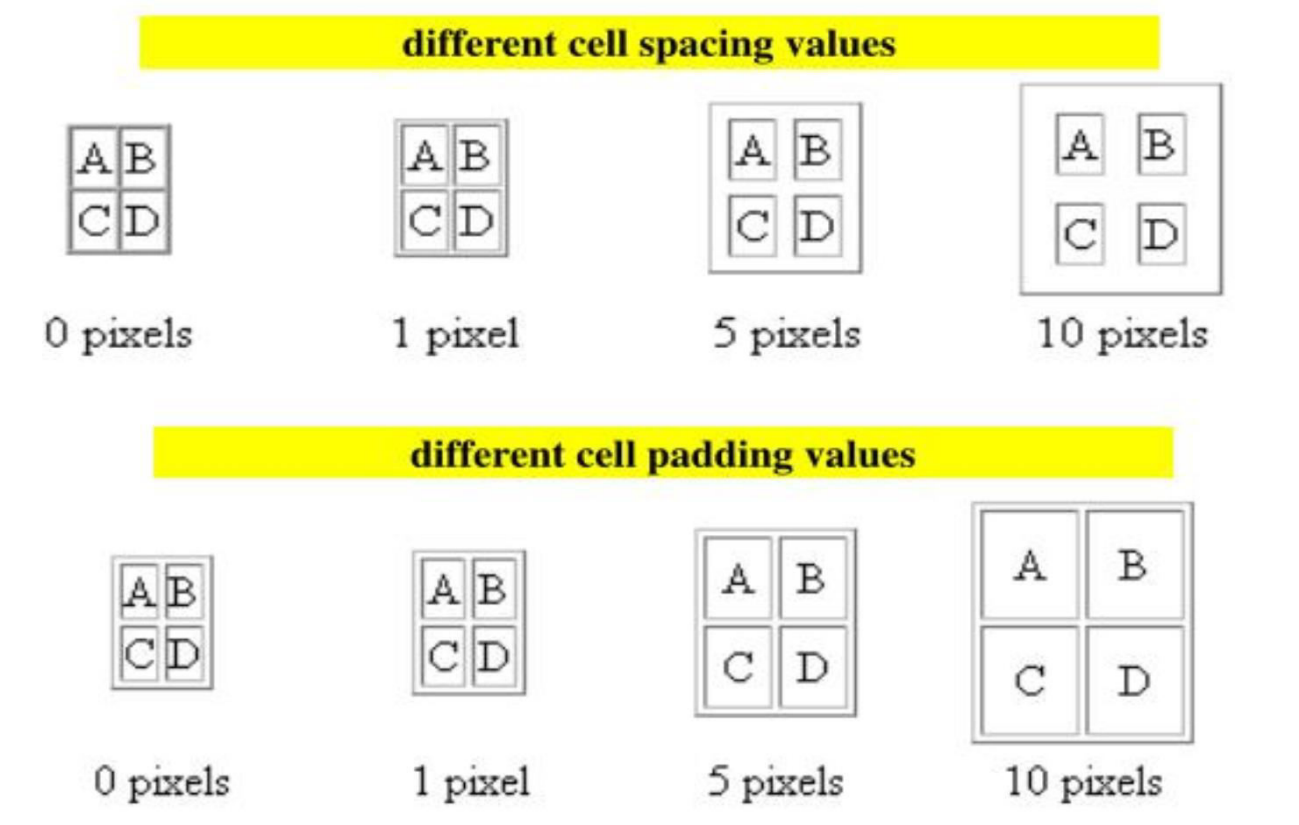


Table Frames and Rules

- Two additional table attributes introduced in HTML 4.0 are the frames and rules attributes.

- With the frame and rule attributes you can control how borders and gridlines are applied to the table.

- The frames attribute allows you to determine which sides of the table will have borders.

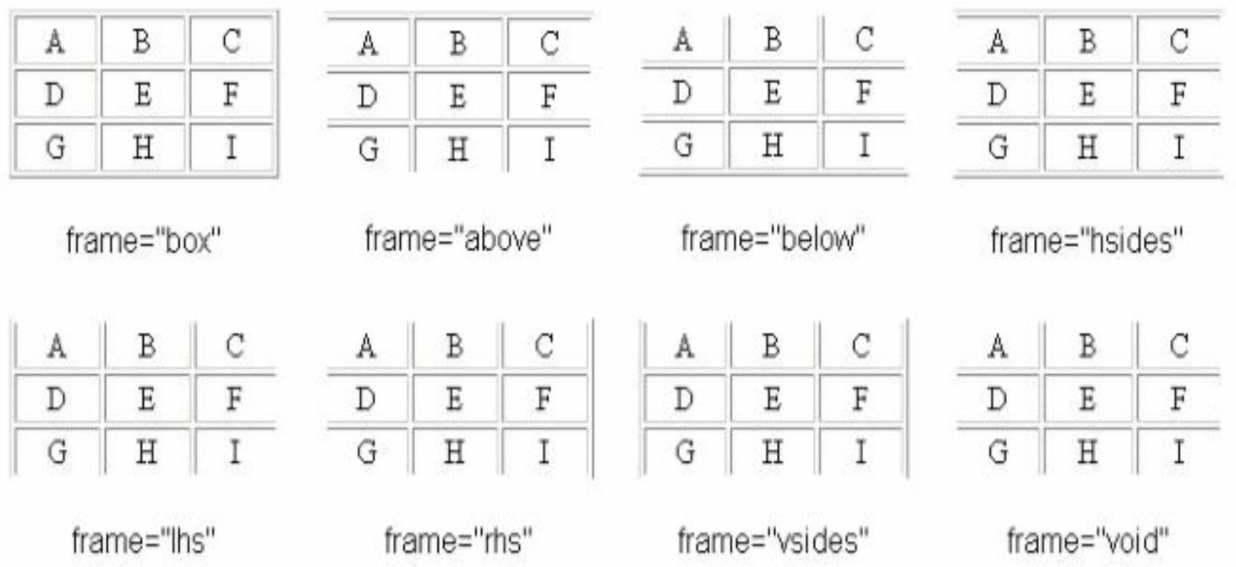
- The frame attribute syntax is:

  <table frame="type">

  type is either "box" (the default), "above", "below", "hsides", "vsides", "lhs", "rhs", or "void"

Table Frames and Rules

This figure shows the effect of each of the frame values on the table grid.



Creating Frames and Rules

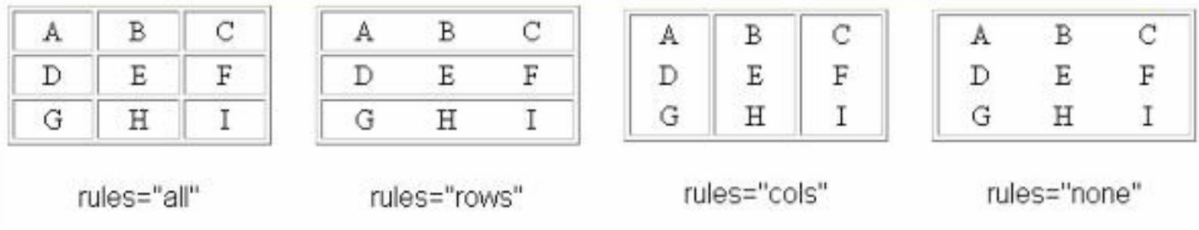
- The rules attribute lets you control how the table gridlines are drawn (not supported by Netscape)

- The syntax of the rules attribute is:

  <table rules="type">

  type is either "all", "rows", "cols", or "none" the effect of each of the rules attribute values on a table

the effect of each of the rules attribute values on a table



Working with Table and Cell Size

• The size of a table is determined by text it contains in its cells.

• By default, HTML places text on a single line.

• As you add text in a cell, the width of the column and table expands to the edge of the page.

• Once the page edge is reached, the browser reduces the size of the remaining columns to keep the text to a single line •

• When the browser can no longer increase or decrease the size of the column and table it wraps the text to a second line. •

Defining the Table Size

• The syntax for specifying the table size is:

<table width="size" height="size">

• size is the width and height of the table as measured in pixels or as a percentage of the display area

• To create a table whose height is equal to the entire height of the display area, enter the attribute height="100%".

• If you specify an absolute size for a table in pixels, its size remains constant, regardless of the browser or monitor settings used.

Defining Cell and Column Sizes

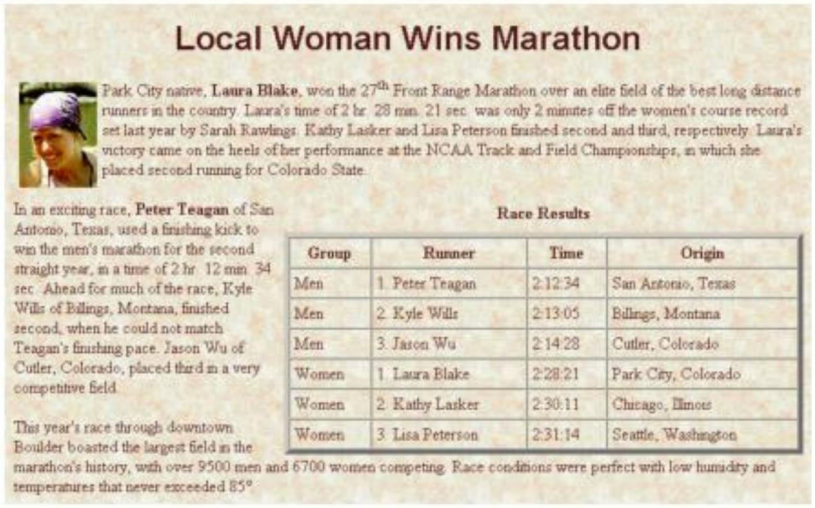
- To set the width of an individual cell, add the width attribute to either the <td> or <th> tags.
- The syntax is: width="value" - value can be expressed in pixels or as a percentage of the table width
- The height attribute can also be used in the <td> or <th> tags to set the height of individual cells.
- The height attribute is expressed either in pixels or as a percentage of the height of the table.

Aligning a Table on the Web Page

- By default, a browser places a table on the left margin of a Web page, with surrounding text placed above and below the table.
- To align a table with the surrounding text, use the align attribute as follows: align="alignment"
- alignment equals "left", "right", or "center"
- left or right alignment places the table on the margin of the Web page and wraps surrounding text to the side
- center alignment places the table in the horizontal center of the page, but does not allow text to wrap around it

Aligning a Table on the Web Page

```
<table border="5" cellspacing="0" cellpadding="4" width="500" align="right">
   <caption align="top"><b>Race Results</b></caption>
   <tr>
      <th width="50">Group</th>
      <th>Runner</th>
      <th>Time</th>
      <th>Origin</th>
   </tr>
```



Local Woman Wins Marathon

Park City native, **Laura Blake**, won the 27th Front Range Marathon over an elite field of the best long distance runners in the country. Laura's time of 2 hr. 28 min. 21 sec. was only 2 minutes off the women's course record set last year by Sarah Rawlings. Kathy Lasker and Lisa Peterson finished second and third, respectively. Laura's victory came on the heels of her performance at the NCAA Track and Field Championships, in which she placed second running for Colorado State.

In an exciting race, **Peter Teagan** of San Antonio, Texas, used a finishing kick to win the men's marathon for the second straight year, in a time of 2 hr. 12 min. 34 sec. Ahead for much of the race, Kyle Wills of Billings, Montana, finished second, when he could not match Teagan's finishing pace. Jason Wu of Cutler, Colorado, placed third in a very competitive field.

Race Results

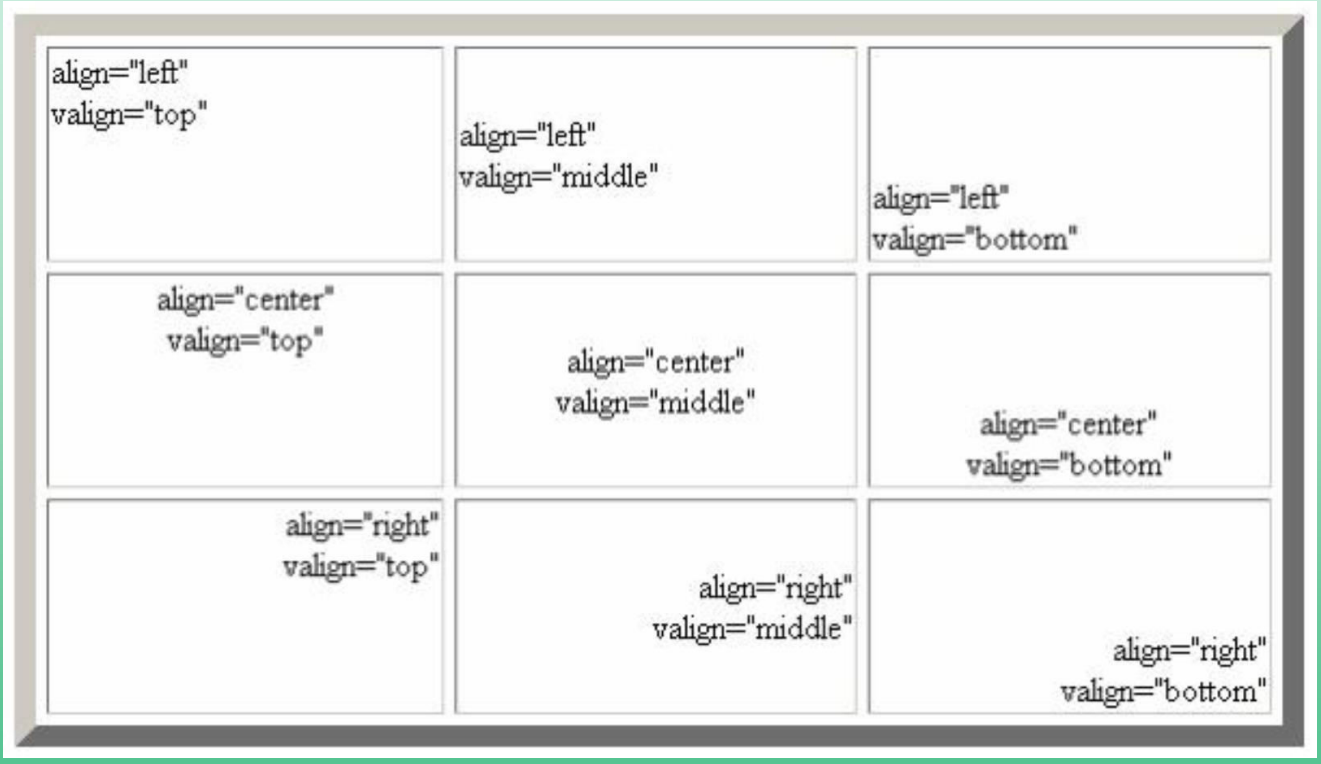
Group	Runner	Time	Origin
Men	1. Peter Teagan	2:12:34	San Antonio, Texas
Men	2. Kyle Wills	2:13:05	Billings, Montana
Men	3. Jason Wu	2:14:28	Cutler, Colorado
Women	1. Laura Blake	2:28:21	Park City, Colorado
Women	2. Kathy Lasker	2:30:11	Chicago, Illinois
Women	3. Lisa Peterson	2:31:14	Seattle, Washington

This year's race through downtown Boulder boasted the largest field in the marathon's history, with over 9500 men and 6700 women competing. Race conditions were perfect with low humidity and temperatures that never exceeded 85°.

Aligning the Contents of a Table

- • By default, cell text is placed in the middle of the cell, aligned with the cell's left edge.
- • By using the align and valign attributes, you can specify the text's horizontal and vertical placement.
- • To align the text for a single column, you must apply the align attribute to every cell in that column.

Aligning the Contents of a Table



Spanning Rows and Columns

- To merge several cells into one, you need to create a spanning cell.

- A spanning cell is a cell that occupies more than one row or column in a table.

- Spanning cells are created by inserting the rowspan and colspan attribute in a <td> or <th> tag.

- The syntax for these attributes is: rowspan="value" colspan="value" - value is the number of rows or columns that the cell spans

Spanning Rows and Columns

This cell spans two columns and two rows

this cell spans three columns

This cell spans three rows

Today's Opinion Poll Question		Political Party		
		Democrat	Republican	Independent
"Do you favor or oppose increasing the minimum wage?"	Favor	70%	35%	55%
	Oppose	25%	60%	30%
	Unsure	5%	5%	15%

Row Spanning

four table cells in the first row

only three table cells are required for the second and third rows

```
<table>
   <tr>
      <td rowspan="3">1: This cell spans three rows</td>
      <td>2</td>
      <td>3</td>
      <td>4</td>
   </tr>
   <tr>
      <td>5</td>
      <td>6</td>
      <td>7</td>
   </tr>
   <tr>
      <td>8</td>
      <td>9</td>
      <td>10</td>
   </tr>
</table>
```

HTML code

1: This cell spans three rows	2	3	4
	5	6	7
	8	9	10

resulting table

Column Spanning

```
<table border="5" cellspacing="0" cellpadding="4" width="500" align="right">
   <caption align="top"><b>Race Results</b></caption>
   <tr>
      <th colspan="2">Runner</th>
      <th>Time</th>
      <th>Origin</th>
   </tr>
   <tr>
      <td rowspan="3">Men</td>
      <td>1. Peter Teagan</td>
      <td align="right">2:12:34</td>
      <td>San Antonio, Texas</td>
   </tr>
   <tr>
      <td>2. Kyle Wills</td>
      <td align="right">2:13:05</td>
      <td>Billings, Montana</td>
   </tr>
   <tr>
      <td>3. Jason Wu</td>
      <td align="right">2:14:28</td>
      <td>Cutler, Colorado</td>
   </tr>
   <tr>
      <td rowspan="3">Women</td>
      <td>1. Laura Blake</td>
      <td align="right">2:28:21</td>
      <td>Park City, Colorado</td>
   </tr>
   <tr>
      <td>2. Kathy Lasker</td>
      <td align="right">2:30:11</td>
      <td>Chicago, Illinois</td>
   </tr>
   <tr>
      <td>3. Lisa Peterson</td>
      <td align="right">2:31:14</td>
      <td>Seattle, Washington</td>
   </tr>
</table>
```

Race Results

Runner		Time	Origin
Men	1. Peter Teagan	2:12:34	San Antonio, Texas
	2. Kyle Wills	2:13:05	Billings, Montana
	3. Jason Wu	2:14:28	Cutler, Colorado
Women	1. Laura Blake	2:28:21	Park City, Colorado
	2. Kathy Lasker	2:30:11	Chicago, Illinois
	3. Lisa Peterson	2:31:14	Seattle, Washington

Another example of Spanning

```
<TR>
    <TD BGCOLOR=YELLOW ROWSPAN=2>Gargoyle Judge</TD>
    <TD WIDTH=60>48222</TD>
    <TD>Bust</TD>
    <TD>Interior Plaster</TD>
    <TD ALIGN=RIGHT WIDTH=50>$140</TD>
</TR>
<TR>
    <TD WIDTH=60>48223</TD>
    <TD>Bust</TD>
    <TD>Gothic Stone</TD>
    <TD ALIGN=RIGHT WIDTH=50>$155</TD>
</TR>
</TABLE>
```

Here is a sample of our products

Name	Item #	Type	Finish	Price
Bacchus	48059	Wall Mount	Interior Plaster	$95
Praying Gargoyle	48159	Garden Figure	Gothic Stone	$125
Gargoyle Judge	48222	Bust	Interior Plaster	$140
	48223	Bust	Gothic Stone	$155

Changing a Cell's Color

- To change a cell's color, add the BGCOLOR="color" attribute to the <TD> tag
- Example:

  <TD BGCOLOR="blue">

Dividing Table into Horizontal Sections

- You can also create a horizontal section consisting of one or more rows. This allows you to format the rows all at once
- To create a horizontal section, type <THEAD>, <TBODY>, or <TFOOT> before the first <TR> tag of the section
- Netscape does not support these tags

Controlling Line Breaks

- Unless you specify otherwise a browser will divide the lines in a cell as it sees fit.
- The NOWRAP attribute placed within the <TD> tag forces the browser to keep all the text in a cell on one line
- Example:
  - <TD NOWRAP>Washington, D.C.

Parting Words….

- If you can imagine a way to lay out your page, chances are it is possible using HTML
- When in doubt, use an HTML reference

Bootstrap 4

What is Bootstrap?

- Bootstrap is a free front-end framework for faster and easier web development
- Bootstrap includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many other, as well as optional JavaScript plugins
- Bootstrap also gives you the ability to easily create responsive design

What is Responsive Web Design?

- Responsive web design is about creating web sites which automatically adjust themselves to look good on all devices, from small phones to large desktops.

```
<div class="jumbotron text-center">
  <h1>My First Bootstrap Page</h1>
  <p>Resize this responsive page to see the effect!</p>
</div>
<div class="container">
  <div class="row">
    <div class="col-sm-4">
      <h3>Column 1</h3>
      <p>Lorem ipsum dolor..</p>
    </div>
    <div class="col-sm-4">
      <h3>Column 2</h3>
      <p>Lorem ipsum dolor..</p>
    </div>
    <div class="col-sm-4">
      <h3>Column 3</h3>
      <p>Lorem ipsum dolor..</p>
    </div>
  </div>
</div>
```

Bootstrap 3 vs. Bootstrap 4

- Bootstrap 4 is the newest version of Bootstrap; with new components, faster stylesheet and more responsiveness.

- Bootstrap 4 supports the latest, stable releases of all major browsers and platforms. However, Internet Explorer 9 and down is not supported.

Why Use 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 Oper

How to add Bootstrap in Angular 9

- Install bootstrap with jquery and popper js by typing following commands from command prompt.

```
npm install bootstrap --save
npm install jquery --save
npm install popper.js --save
```

- Import bootstrap into project by adding following lines in angular.json file.

```
"styles": [
    "node_modules/bootstrap/dist/css/bootstrap.min.css",
    "src/styles.css"
],
"scripts": [
    "node_modules/jquery/dist/jquery.min.js",
    "node_modules/bootstrap/dist/js/bootstrap.min.js"
]
```

How to add Bootstrap in Angular 9 …cont.

```
<div class="container">
  <h1>Install Bootstrap 4 in Angular 9 - ItSolutionStuff.com</h1>

  <div class="card">
    <div class="card-header">
      Featured
    </div>
    <div class="card-body">
      <h5 class="card-title">Special title treatment</h5>
      <p class="card-text">With supporting text below as a natural lead-in to additional content.</p>
      <a href="#" class="btn btn-primary">Go somewhere</a>
    </div>
  </div>
</div>
```

Creating web page with bootstrap 4

- **Add the HTML5 doctype**
  - Bootstrap uses HTML elements and CSS properties that require the HTML5 doctype.
  - Always include the HTML5 doctype at the beginning of the page, along with the lang attribute and the correct character set:

```
<!DOCTYPE html>
<html lang="en">
      <head>
        <meta charset="utf-8">
      </head>
</html>
```

Creating web page with bootstrap 4 ….cont.

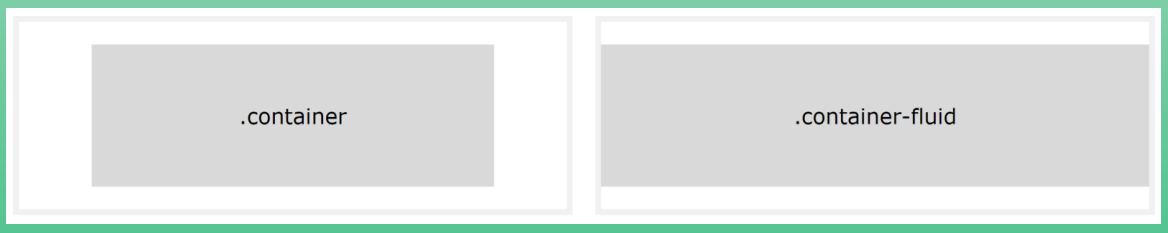
- **Bootstrap is mobile-first**
  - Bootstrap 3 & 4 is designed to be responsive to mobile devices. Mobile-first styles are part of the core framework.
  - To ensure proper rendering and touch zooming, add the following <meta> tag inside the <head> element:

```
<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.

Creating web page with bootstrap 4 ….cont.

- **Containers**
  - Bootstrap 4 requires a containing element to wrap site contents.
  - There are two container classes to choose from:
    1. The **.container** class provides a responsive fixed width container
    2. The **.container-fluid** class provides a full width container, spanning the entire width of the viewport

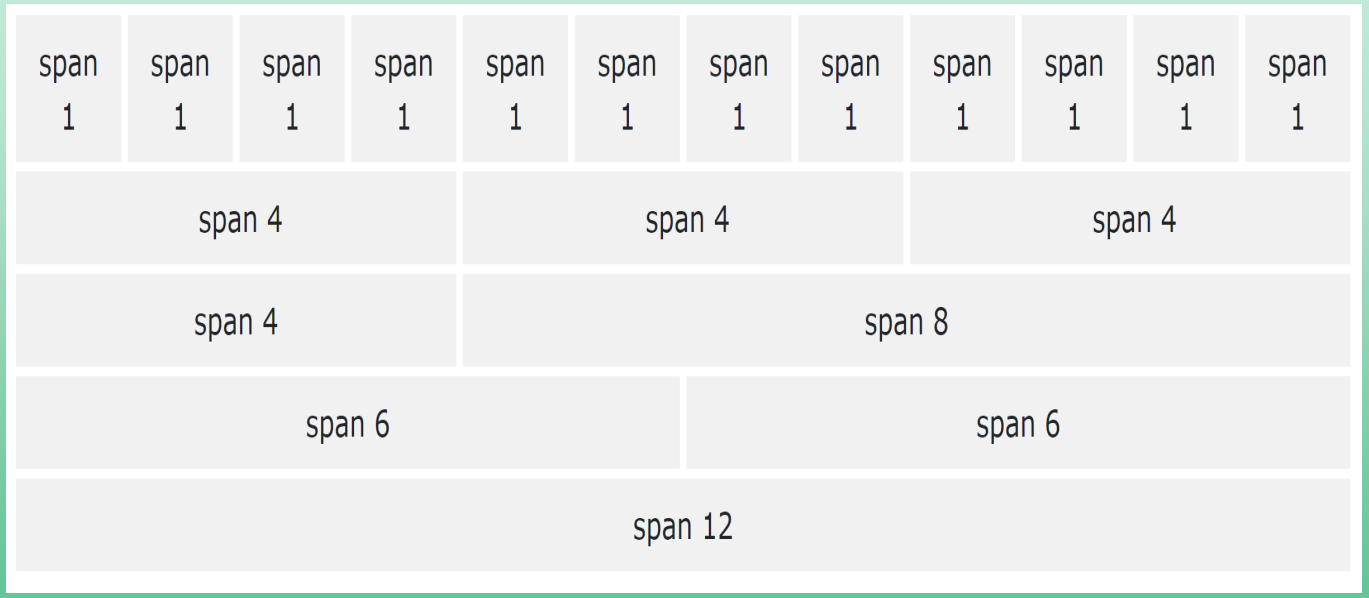


Note: Containers cannot be nested (you cannot put a container inside another container).

Bootstrap 4 Grid System

- Bootstrap's grid system is built with flexbox and allows up to 12 columns across the page.
- If you do not want to use all 12 columns individually, you can group the columns together to create wider columns:
- The grid system is responsive, and the columns will re-arrange automatically depending on the screen size.
- Make sure that the sum adds up to 12 or fewer (it is not required that you use all 12 available columns).

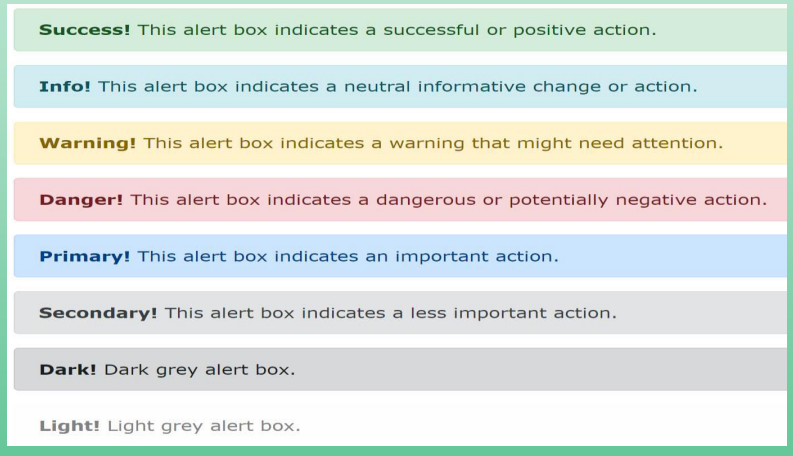
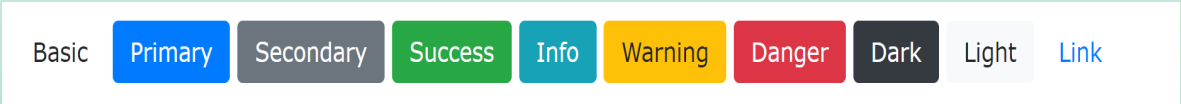
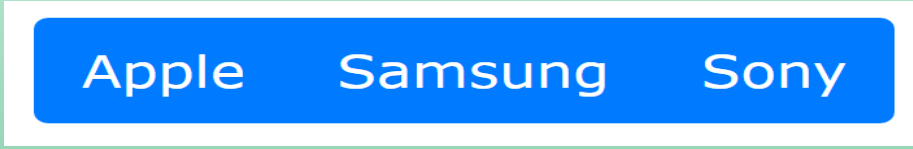
Bootstrap 4 Grid System ….Cont.



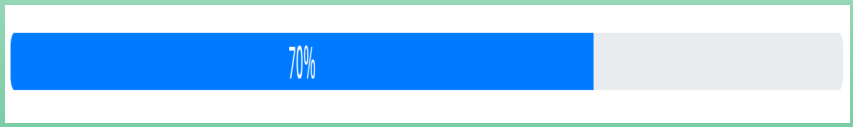
•Grid Classes

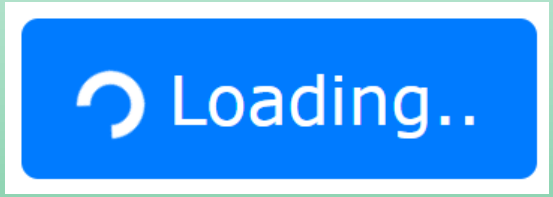
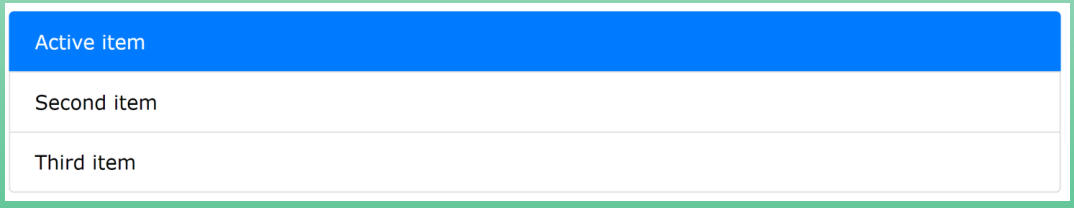
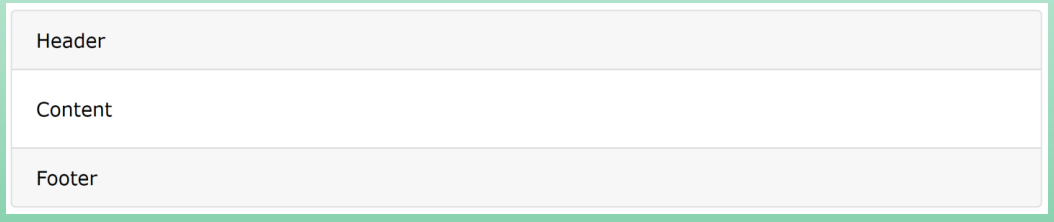
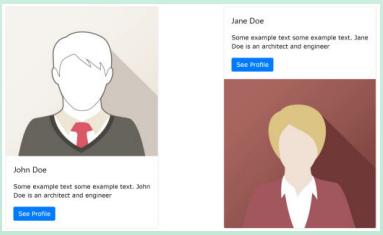
- The Bootstrap 4 grid system has five classes:
  - .col- (extra small devices - screen width less than 576px)
  - .col-sm- (small devices - screen width equal to or greater than 576px)
  - .col-md- (medium devices - screen width equal to or greater than 768px)
  - .col-lg- (large devices - screen width equal to or greater than 992px)
  - .col-xl- (xlarge devices - screen width equal to or greater than 1200px)

Basic Structure of a Bootstrap 4 Grid

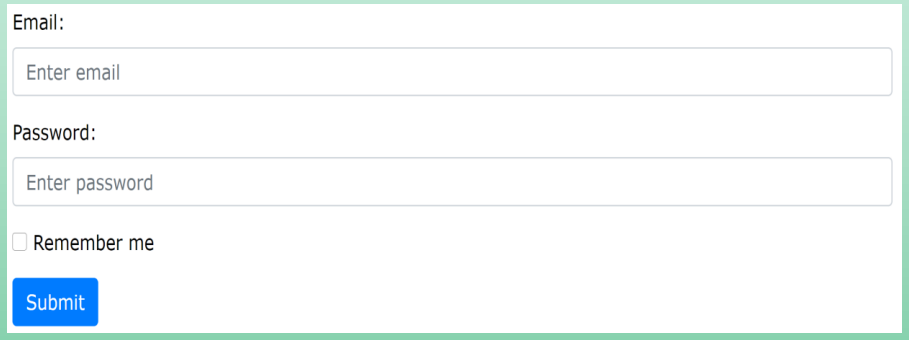
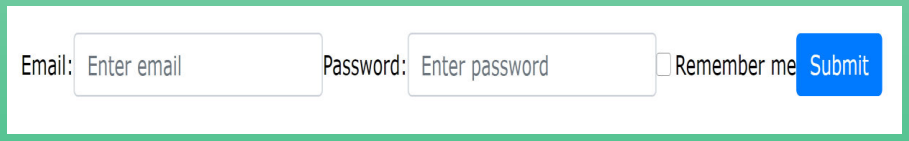
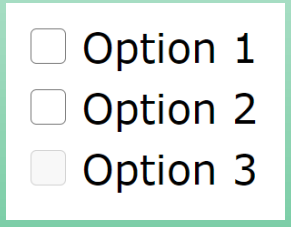
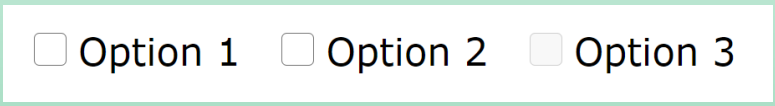
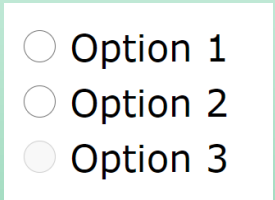
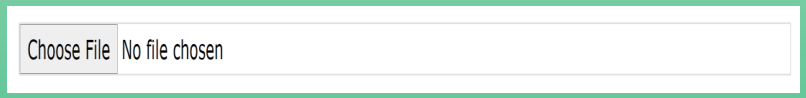
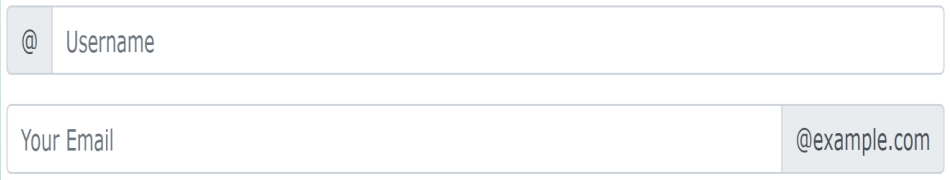
- ```
  <!-- Control the column width, and how they should appear on different devices -->

  <!-- Or let Bootstrap automatically handle the layout -->

  ```
- 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 or xl,
- The second star represents a number, which should add up to 12 for each row.
  ```

Bootstrap 4 Color classes

- Following are the classes for text colors:
  - **text-muted** This text is muted.
  - **text-primary** This text is primary.
  - **text-success** This text indicates success.
  - **text-info** This text represents some information
  - **text-warning** This text represents a warning.
  - **text-danger** This text represents danger.
  - **text-secondary** Secondary text.
  - **text-white** Dark grey text.
  - **text-dark** Dark grey text.
  - **text-body (default body color/often black) and .text-light:**

- You can also add 50% opacity for black or white text with following classes
  - **text-black-50**
  - **text-white-50**

Bootstrap 4 Color classes ….Cont…

- Following are the classes for background color:

  - **bg-primary**
  - **bg-success**
  - **bg-info**
  - **bg-warning**
  - **bg-danger**
  - **bg-secondary**
  - **bg-dark**
  - **bg-light.**

Bootstrap table class

- **table class**
  - ➢ table has a light padding and horizontal dividers.

Firstname	Lastname	Email
John	Doe	john@example.com
Mary	Moe	mary@example.com
July	Dooley	july@example.com

Bootstrap Striped rows table class

- **table-striped class**
  - ➢ The .table-striped class adds zebra-stripes to a table.

Firstname	Lastname	Email
John	Doe	john@example.com
Mary	Moe	mary@example.com
July	Dooley	july@example.com

Bootstrap Bordered table class

- **table-bordered class**
  - The table-bordered class adds borders on all sides of the table and cells

Firstname	Lastname	Email
John	Doe	john@example.com
Mary	Moe	mary@example.com
July	Dooley	july@example.com

Bootstrap Black/dark table class

- **table-dark class**
  - The .table-dark class adds a black background to the table

Firstname	Lastname	Email
John	Doe	john@example.com
Mary	Moe	mary@example.com
July	Dooley	july@example.com

Other table classes

- Hover Rows
  - The .table-hover class adds a hover effect (grey background color) on table rows.

- Borderless Table
  - The .table-borderless class removes borders from the table

Table contextual classes

- Contextual classes can be used to color the whole table (<table>), the table rows (<tr>) or table cells (<td>).

  - .table-primary
  - .table-success
  - .table-danger
  - .table-info
  - .table-warning
  - .table-active
  - .table-secondary
  - .table-light
  - .table-dark

Firstname	Lastname
Default	Defaultson
Primary	Joe
Success	Doe
Danger	Moe
Info	Dooley
Warning	Refs
Active	Activeson
Secondary	Secondson
Light	Angie
Dark	Bo

Bootstrap 4 Image shape classes

Rounded Corners:

Circle:

Thumbnail:

- Rounded Corners
  The .rounded class adds rounded corners to an image:

```
<img src="cinqueterre.jpg" class="rounded" alt="Cinque Terre">
```

- Circle
  The .rounded-circle class shapes the image to a circle

- Thumbnail
  The .img-thumbnail class shapes the image to a thumbnail (bordered)

Aligning Images

- Float an image to the right or left using **float-right** class & **float-left.**

```
<img src="paris.jpg" class="float-left">
<img src="paris.jpg" class="float-right">
```

- Image can be centered by adding the utility classes **.mx-auto** (margin:auto) and **.d-block** (display:block) to the image:

```
<img src="paris.jpg" class="mx-auto d-block">
```

Example

```
<div class="container">
    <div class="row">
        <div class="col-sm-4">
          <img src="../assets/bird1.jpg" alt="Bird1" width="200" height="200"
               class="rounded-circle">
        </div>
        <div class="col-sm-4">
          <img src="../assets/bird2.jpg" alt="Bird2"  width="200" height="200"
               class="rounded">
        </div>
        <div class="col-sm-4">
          <img src="../assets/bird3.jpg" alt="Bird3"  width="200" height="200">
        </div>
    </div>
</div>
```

Alert messages

- Bootstrap 4 provides an easy way to create predefined alert messages.

- Alerts are created with the .alert class, followed by one of the contextual classes

.alert-success,
.alert-info,
.alert-warning,
.alert-danger,
.alert-primary,
.alert-secondary,
.alert-light
.alert-dark:




```
<div class="alert alert-success">
  <strong>Success!</strong> Indicates a successful or positive action.
</div>
```

Button class

- Bootstrap 4 provides different styles of buttons:




```
<button type="button" class="btn">Basic</button>
<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-info">Info</button>
<button type="button" class="btn btn-warning">Warning</button>
<button type="button" class="btn btn-danger">Danger</button>
<button type="button" class="btn btn-dark">Dark</button>
<button type="button" class="btn btn-light">Light</button>
<button type="button" class="btn btn-link">Link</button>
```

Active/Disabled Buttons

- A button can be set to an active (appear pressed) or a disabled (unclickable) state.


```
<button type="button" class="btn btn-primary active">Active Primary</button>
<button type="button" class="btn btn-primary" disabled>Disabled Primary</button>
<a href="#" class="btn btn-primary disabled">Disabled Link</a>
```

Button Groups

- Button group allows you to group a series of buttons together (on a single line) in a button group

- Use a element with class .btn-group to create a button group




- ```

    <button type="button" class="btn btn-primary">Apple</button>
    <button type="button" class="btn btn-primary">Samsung</button>
    <button type="button" class="btn btn-primary">Sony</button>

  ```

Vertical Button Groups

- Bootstrap 4 also supports vertical button groups

- Use the class .btn-group-vertical to create a vertical button group


- ```
  <div class="btn-group-vertical">
    <button type="button" class="btn btn-primary">Apple</button>
    <button type="button" class="btn btn-primary">Samsung</button>
    <button type="button" class="btn btn-primary">Sony</button>
  </div>
  ```

Progress bar

- A progress bar can be used to show a user how far along he/she is in a process.

```
<div class="progress" style="width : 300px; height:50px">
   <div class="progress-bar bg-success" style="width:70%; height:50px">70%</div>
</div>
```



Spinners

- To create a spinner/loader, use the .spinner-border class

```
<div class="spinner-border"></div>
```

- Colored Spinners

```
<div class="spinner-border text-muted"></div>
<div class="spinner-border text-primary"></div>
<div class="spinner-border text-success"></div>
<div class="spinner-border text-info"></div>
<div class="spinner-border text-warning"></div>
<div class="spinner-border text-danger"></div>
<div class="spinner-border text-secondary"></div>
<div class="spinner-border text-dark"></div>
<div class="spinner-border text-light"></div>
```

Growing spinners

- The spinner-grow class is used if you want the spinner/loader to grow instead of "spin“

- ```

  ```

Spinner button

- ```
  <button class="btn btn-primary">
    <span class="spinner-border spinner-border-sm"></span>
    Loading..
  </button>
  ```



Pagination

- If you have a web site with lots of pages, you may wish to add some sort of pagination to each page.

-
```
<ul class="pagination">
  <li class="page-item"><a class="page-link" href="#">Previous</a></li>
  <li class="page-item"><a class="page-link" href="#">1</a></li>
  <li class="page-item"><a class="page-link" href="#">2</a></li>
  <li class="page-item"><a class="page-link" href="#">3</a></li>
  <li class="page-item"><a class="page-link" href="#">Next</a></li>
</ul>
```

List group

- To create a list group, use an <ul> element with class .list-group, and <li> elements with class .list-group-item

-
```
<ul class="list-group">
  <li class="list-group-item active">Active item</li>
  <li class="list-group-item">Second item</li>
  <li class="list-group-item">Third item</li>
</ul>
```



Cards

- A card is a bordered box with some padding around its content.
- It includes options for headers, footers, content, colors, etc.




```
<div class="card">
    <div class="card-header">Header</div>
    <div class="card-body">Content</div>
    <div class="card-footer">Footer</div>
</div>
```

Cards images



- Card box can have image within it.
- Add .card-img-top or .card-img-bottom to an <img> to place the image at the top or at the bottom inside the card.
- Image to be added outside of the .card-body to span the entire width

```
<div class="card" style="width:400px">
  <img class="card-img-top" src="img_avatar1.png" alt="Card image">
  <div class="card-body">
    <h4 class="card-title">John Doe</h4>
    <p class="card-text">Some example text.</p>
    <a href="#" class="btn btn-primary">See Profile</a>
  </div>
</div>
```

Navigation menus

- Nav are simple navigation menus.

- Add the .nav class to a <ul> element, followed by .nav-item for each <li> and add the .nav-link class to their links

```
<ul class="nav">
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link disabled" href="#">Disabled</a>
  </li>
</ul>
```

Navigation menus …cont.

- Other classes that can be used with nav are as follows :

  - justify-content-center - Places menu center
  - justify-content-end - Places menu right
  - flex-column - Places menu vertically
  - nav-tabs - Places menu horizontal like tab
  - nav-pills - Places menu horizontal like pills

Navigation bar

- A navigation bar is a navigation header that is placed at the top of the page
- With Bootstrap, a navigation bar can extend or collapse, depending on the screen size.
- A standard navigation bar is created with the .navbar class, followed by a responsive collapsing class: .navbar-expand-xl|lg|md|sm (stacks the navbar vertically on extra large, large, medium or small screens).
- To add links inside the navbar, use a <ul> element with class="navbar-nav". Then add <li> elements with a .nav-item class followed by an <a> element with a .nav-link class

Navigation bar ….cont..


```
<nav class="navbar navbar-expand-sm bg-primary navbar-dark">
  <!-- Brand/logo -->
  <a class="navbar-
    brand" href="#"><img src="../assets/image1.jpg" width="40" height=
    "40"></a>
   <!-- Links -->
  <ul class="navbar-nav">
    <li class="nav-item">  <a class="nav-link" href="#">Link 1</a> </li>
     <li class="nav-item"> <a class="nav-link" href="#">Link 2</a> </li>
     <li class="nav-item"> <a class="nav-link" href="#">Link 3</a> </li>
  </ul>
</nav>
```

Navigation bar ….cont..

- Vertical nav bar - remove .navbar-expand-xl|lg|md|sm
- justify-content-center - Centered nav bar
- navbar-brand - To add logo Logo

Collapsible navigation bar

- To create a collapsible navigation bar, use a
  - Button with class="navbar-toggler", data-toggle="collapse" and data-target="#thetarget".
  - Then wrap the navbar content (links, etc) inside a div element with class="collapse navbar-collapse", followed by an id that matches the data-target of the button: "thetarget".

Collapsible navigation bar ….cont…

```
<nav class="navbar navbar-expand-md bg-dark navbar-dark">
  <!-- Brand logo-->
  <a class="navbar-brand" href="#">SavitaSoft</a>

  <!-- Toggler/collapsibe Button -->
  <button class="navbar-toggler" type="button"
            data-toggle="collapse" data-target="#collapsibleNavbar">
    <span class="navbar-toggler-icon"></span>
  </button>

  <!-- Navbar links -->
  <div class="collapse navbar-collapse" id="collapsibleNavbar">
    <ul class="navbar-nav">
        <li class="nav-item">  <a class="nav-link" href="#">Home</a>  </li>
        <li class="nav-item">  <a class="nav-link" href="#">About</a>  </li>
        <li class="nav-item">  <a class="nav-link" href="#">Contact</a>  </li>
    </ul>
  </div>
</nav>
```

Bootstrap 4 forms

- Bootstrap provides two types of form layouts:
  - Stacked (full-width) form



  - Inline form <form class="form-inline" action="/action_page.php">



Bootstrap 4 inputs - Textbox & Textarea

- Bootstrap supports all the HTML5 input types: text, password, datetime, datetime-local, date, month, time, week, number, email, url, tel, and color.

- Use .form-control class for textbox and textarea to style inputs with full-width and proper padding

```
<input type="text" class="form-control" id="usr">
```

```
<div class="form-group">
  <label for="usr">Name:</label>
  <input type="text" class="form-control" id="usr">
</div>
<div class="form-group">
  <label for="comment">Comment:</label>
  <textarea class="form-control" rows="5" id="comment"></textarea>
</div>
```

Bootstrap 4 inputs - Checkbox

- Use a wrapper element with class="form-check" to ensure proper margins for labels and checkboxes.

- Add the .form-check-label class to label elements, and .form-check-input to style checkboxes properly inside the .form-check container



-
```
<div class="form-check">
  <label class="form-check-label">
    <input type="checkbox" class="form-check-input" value="">Option 1
  </label>
</div>
```

Bootstrap 4 inputs - Inline Checkbox

- Use .form-check-inline class if you want the checkboxes to appear on the same line.



-
```
<div class="form-check-inline">
  <label class="form-check-label">
    <input type="checkbox" class="form-check-input" value="">Option 1
  </label>
</div>
```

Bootstrap 4 inputs - Radiobutton

- Radio buttons are used if you want to limit the user to just one selection from a list of preset options.




```
<div class="form-check">
  <label class="form-check-label">
    <input type="radio" class="form-check-input" name="optradio"> Option 1
  </label>
</div>
<div class="form-check disabled">
  <label class="form-check-label">
    <input type="radio" class="form-check-input" name="optradio" disabled>Option 2
  </label>
</div>
```

Bootstrap 4 inputs - List box & dropdown box

- Listboxes are used if you want to allow the user to pick from multiple options. By applying multiple, can be converted to list box.

```
<div class="form-group">
    <label for="sel1">Select list:</label>
    <select class="form-control" id="sel1">
      <option>1</option>
      <option>2</option>
      <option>3</option>
      <option>4</option>
    </select>
 </div>
```

Bootstrap 4 inputs - Selcting Range and File

- Add the .form-control-range class to input type "range“.

```
<input type="range" class="form-control-range" name="range">
```

- Add the .form-control-file class to input type "file"




```
<input type="file" class="form-control-file border" name="file">
```

Input groups



- The **.input-group** class is a container to enhance an input by adding an icon, text or a button in front or behind the input field as a "help text".
- Use **.input-group-prepend** to add the help text in front of the input, and **.input-group-append** to add it behind the input.
- At last, add the **.input-group-text** class to style the specified help text
- ```

      @

    <input type="text" class="form-control" placeholder="Username">

  ```

Bootstrap Carousel

- The carousel is a slideshow for cycling through a series of content, built with CSS 3D transforms and a bit of JavaScript.
- It works with a series of images, text, or custom markups.
- It also includes support for previous/next controls and indicators.
  ```

Bootstrap Carousel

- To add carousel add a div with attribute class as carousel, data-ride as carousel and data-interval as 1000.

```
<div id="demo"
    class="carousel slide"
      data-ride="carousel"
      data-interval="1000">
       .....
       .....
</div>
```

Bootstrap Carousel inner div

- Add Carousel-inner div within Carousel div.

- Carousel-inner div will contain carousel items

```
<div class="carousel-inner">
  .....carousel
    items.....
  .....carousel
    items.....
</div>
```

Bootstrap Carousel items

- Add div with carousel-item class attribute to add image, caption, and text

```
<div class="carousel-item active">
    <img src="assets/images/img1.jpg"
        alt="SavitaSoft"
        width="100%"
        height="400">
</div>
```

Carousel prev and next control

- Add hyper link to provide previous and next control

```
<a href="#demo" class="carousel-control-prev" data-slide="prev">
    <span class="carousel-control-prev-icon"></span>
</a>
<a href="#demo" class="carousel-control-next" data-slide="next">
    <span class="carousel-control-next-icon"></span>
</a>
```

Carousel navigation indicator

- Carousel indicator

```
<ul class="carousel-indicators">
    <li data-target="#demo" data-slide-to="0" class="active"></li>
    <li data-target="#demo" data-slide-to="1"></li>
    <li data-target="#demo" data-slide-to="2"></li>
</ul>
```

Bootstrap Carousel

- Carousel caption

```
<div class="carousel-item active">
    <img src="assets/images/img1.jpg" alt="SavitaSoft" width="100%"
height="400">
    <div class="carousel-caption">
        <h3>SAVITASOFT</h3>
        <p>The name that spells quality education</p>
    </div>
</div>
```

Bootstrap Carousel

- Carousel crossfade

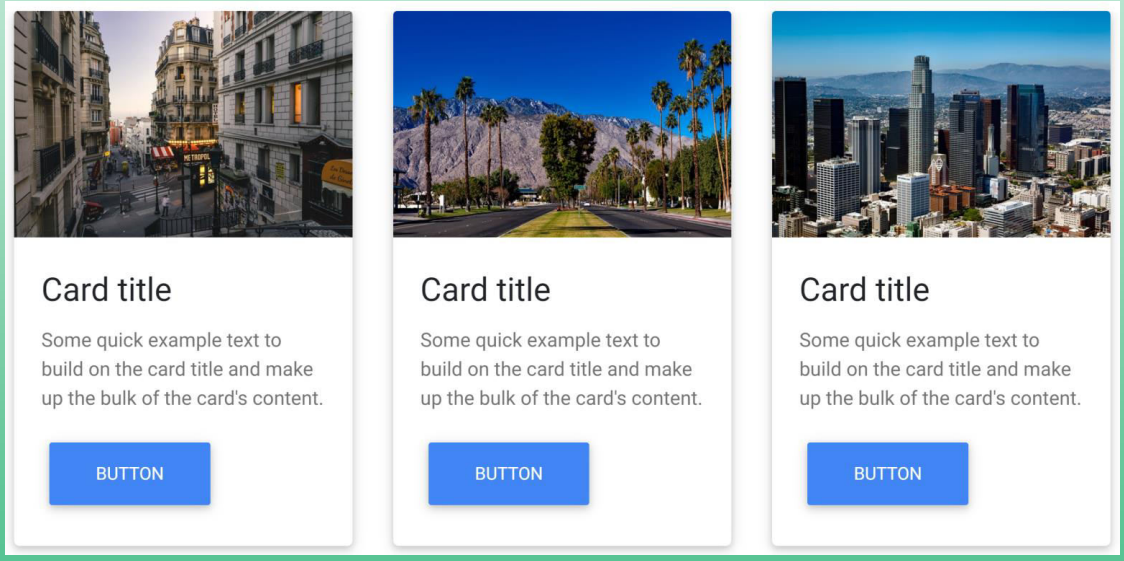
```
<div id="carouselExampleFade"
    class="carousel slide carousel-fade"
        data-ride="carousel">
```

Bootstrap Carousel example

```
<div id="demo" class="carousel slide" data-ride="carousel" data-interval="1000">

        <!-- Indicators -->
    <ul class="carousel-indicators">
      <li data-target="#demo" data-slide-to="0" class="active"></li>
      <li data-target="#demo" data-slide-to="1"></li>
      <li data-target="#demo" data-slide-to="2"></li>
    </ul>

    <!-- The slideshow -->
    <div class="carousel-inner">
      <div class="carousel-item active">
        <img src="../assets/bird1.jpg" alt="Los Angeles" width="100%" height="400">
      <div class="carousel-item">
        <img src="../assets/bird2.jpg" alt="Chicago" width="100%" height="400">
      </div>
      <div class="carousel-item">
        <img src="../assets/bird3.jpg" alt="New York" width="100%" height="400">
      </div>
    </div>

    <!-- Left and right controls -->
    <a class="carousel-control-prev" href="#demo" data-slide="prev">
      <span class="carousel-control-prev-icon"></span>
    </a>
    <a class="carousel-control-next" href="#demo" data-slide="next">
      <span class="carousel-control-next-icon"></span>
    </a>

  </div>
```

Bootstrap Carousel

- Multi item Carousel



Bootstrap Carousel

- https://mdbootstrap.com/docs/jquery/javascript/carousel/