Second Semster MCA Degree Examination July/August 2022

WEB Technologies

20MCA23

Module 1

Q1) a) Write a short notes on the following (10 Marks)

- **i) Web browser**
- **ii) Web server**
- **iii) MIME**

WEB BROWSERS

- Documents provided by servers on the Web are requested by browsers, which are programs running on client machines.
- They are called browsers because they allow the user to browse the resources available on servers.
- Mosaic was the first browser with a graphical user interface.
- A browser is a client on the Web because it initiates the communication with a server, which waits for a request from the client before doing anything.
- In the simplest case, a browser requests a static document from a server.
- The server locates the document among its servable documents and sends it to the browser, which displays it for the user.
- Sometimes a browser directly requests the execution of a program stored on the server.
- The output of the program is then returned to the browser.
- Examples: Internet Explorer, Mozilla Firefox, Netscape Navigator, Google Chrome, Opera etc.,

WEB SERVERS

Web servers are programs that provide documents to requesting browsers. Example: Apache

Web server operations:

- All the communications between a web client and a web server use the HTTP
- When a web server begins execution, it informs the OS under which it is running & it runs as a background process
- A web client or browser, opens a network connection to a web server, sends information requests and possibly data to the server, receives information from the server and closes the connection.

- The primary task of web server is to monitor a communication port on host machine, accept HTTP commands through that port and perform the operations specified by the commands.
- When the URL is received, it is translated into either a filename or a program name

General characteristics of web server:

- The file structure of a web server has two separate directories
- The root of one of these is called document root which stores web documents
- The root of the other directory is called the server root which stores server and its support software's
- The files stored directly in the document root are those available to clients through top level URLs
- The secondary areas from which documents can be served are called virtual document trees.
- Many servers can support more than one site on a computer, potentially reducing the cost of each site and making their maintenance more convenient. Such secondary hosts are called virtual hosts.
- Some servers can serve documents that are in the document root of other machines on the web; in this case they are called as proxy servers

MULTIPURPOSE INTERNET MAIL EXTENSIONS

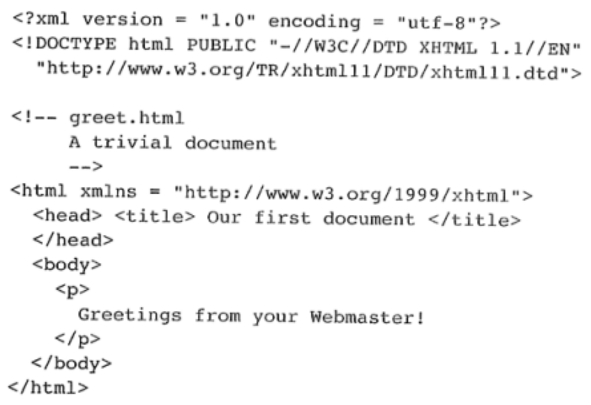
- MIME stands for Multipurpose Internet Mail Extension.
- The server system apart from sending the requested document, it will also send MIME information.
- The MIME information is used by web browser for rendering the document properly.
- The format of MIME is: type/subtype
- Example: text/html , text/doc , image/jpeg , video/mpeg
- When the type is either text or image, the browser renders the document without any
- problem
- However, if the type is video or audio, it cannot render the document
- It has to take the help of other software like media player, win amp etc.,
- These softwares are called as helper applications or plugins
- These non-textual information are known as HYPER MEDIA
- Experimental document types are used when user wants to create a customized information & make it available in the internet
- The format of experimental document type is: type/x-subtype
- Example: database/x-xbase , video/x-msvideo
- Along with creating customized information, the user should also create helper applications.
- This helper application will be used for rendering the document by browser.

- The list of MIME specifications is stored in configuration file of web server.

b) Discuss the basic structure of XHTML document (05 Marks)

Standard XHTML Document Structure

- Every XHTML document must begin with an xml declaration element that simply identifies the document as being one based on XML. This element includes an attribute that specifies the version number 1.0.
- The xml declaration usually includes a second attribute, encoding, which specifies the encoding used for the document, utf-8
- Following is the xml declaration element, which should be the first line of every XHTML document: <?xml version = "1.0" encoding = "utf-8"?>
- Note that this declaration must begin in the first character position of the document file
- The xml declaration element is followed immediately by an SGML DOCTYPE command which specifies the particular SGML document-type definition (DTD) with which the document complies, among other things.
- The following command states that the document in which it is included complies with XHTML 1.0
- !DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
- An XHTML document must include the four tags <html>, <head>, <title>, and <body>.
- The <html> tag identifies the root element of the document So, XHTML documents always have an <html> tag immediately following the DOCTYPE command, and they always end with the closing html tag, </html>.
- The html element includes an attribute, xmlns, that specifies the XHTML namespace, as shown in the following element:
  <html xmlns = "http://www.w3.org/1999/xhtml">
- Although the xmlns attribute’s value looks like a URL, it does not specify a document. It is just a name that happens to have the form of a URL.
- An XHTML document consists of two parts, named the head and the body.
- The <head> element contains the head part of the document, which provides information about the document and does not provide the content of the document.
- The body of a document provides the content of the document.
- The content of the title element is displayed by the browser at the top of its display window, usually in the browser window ’s title bar

- 

```
<?xml version = "1.0" encoding = "utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
  "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<!-- greet.html
     A trivial document
     -->
<html xmlns = "http://www.w3.org/1999/xhtml">
  <head> <title> Our first document </title>
  </head>
  <body>
    <p>
      Greetings from your Webmaster!
    </p>
  </body>
</html>
```

c) what are the rules to be followed to make use of HTML elements in Xhtml document? Explain (05 Marks)

The fundamental syntactic units of HTML are called tags.

- In general, tags are used to specify categories of content.
- The syntax of tag is the tag's name surrounded by angle brackets (<and>).
- Tag names must be written in all lower case letters.
- Most tags appear in pairs: an opening tag and a closing tag.
- The name of a closing tag is the name of its corresponding opening tag with a slash attached to the beginning. For example, if the tag's name is <p>, the corresponding closing tag is named </p>.
- Whatever appears between opening tag and its closing tag is the content of the tag. Not all tags can have content.
- The opening tag and its closing tag together specify a container for the content they enclose.
- The container and its content together are called an element.
- Example: <p>This is extremely simple. </p>
- The paragraph tag, <p>, marks the beginning of the content; the </p> tag marks the end of the content of the paragraph element.
- Attributes, which are used to specify alternative meanings of a tag, can appear between an opening tag's name and its right angle bracket.
- They are specified in keyword form, which means that the attribute's name is followed by an equal's sign and the attribute's value. Attribute names, like tag names, are written in lowercase letters. Attribute values must be delimited by double quotes.
- Comments in programs increase the readability of those programs. Comments in XHTML have the same purpose. They can appear in XHTML in the following form: <!- - anything except two adjacent dashes - ->
- Browsers ignore XHTML comments—they are for people only. Comments can be spread over as many lines as are needed. For example, you could have the following comment:

<!--PetesHome.html

This document describes the home page of pete’s pickles

-->

Q2 a) Demonstrate the use of image maps with a suitable example (10 Marks)

What is image mapping :

In image mapping an image is specified with certain set of coordinates inside the image which act as hyperlink areas to different destinations. It is different from an image link since in image linking, an image can be used to serve a single link or destination whereas in a mapped image, different coordinates of the image can serve different links or destinations.

Elements required in Mapping an Image :

There are three basic html elements which are required for creating a mapped image.

Map : It is used to create a map of the image with clickable areas.

Image : It is used for the image source on which mapping is done.

Area : It is used within the map for defining clickable areas.

Image Maps

The HTML <map> tag defines an image map. An image map is an image with clickable areas. The areas are defined with one or more <area> tags.

```
<!DOCTYPE html>

<html>

<body>

<h2>Image Maps</h2>

<p>Click on the computer, the phone, or the cup of coffee to go to a new page and read more about the topic:</p>

<img src="workplace.jpg" alt="Workplace" usemap="#workmap" width="400" height="379">

<map name="workmap">

  <area shape="rect" coords="34,44,270,350" alt="Computer" href="computer.htm">

  <area shape="rect" coords="290,172,333,250" alt="Phone" href="phone.htm">
```

```
 <area shape="circle" coords="337,300,44" alt="Cup of coffee" href="coffee.htm">

</map>

</body>

</html>
```

he idea behind an image map is that you should be able to perform different actions depending on where in the image you click.

To create an image map you need an image, and some HTML code that describes the clickable areas.

The image is inserted using the <img> tag. The only difference from other images is that you must add a usemap attribute:

```
<img src="workplace.jpg" alt="Workplace" usemap="#workmap">
```

The usemap value starts with a hash tag # followed by the name of the image map, and is used to create a relationship between the image and the image map.

Then, add a <map> element.

The <map> element is used to create an image map, and is linked to the image by using the required name attribute:

```
<map name="workmap">
```

The name attribute must have the same value as the <img>'s usemap attribute.

Then, add the clickable areas.

A clickable area is defined using an <area> element.

You must define the shape of the clickable area, and you can choose one of these values:

rect - defines a rectangular region

circle - defines a circular region

poly - defines a polygonal region

default - defines the entire region

You must also define some coordinates to be able to place the clickable area onto the image.

Q2 b) What is a hyperlink? Explain the various ways of creating hyper links with example (10 Marks)

Links are specified in an attribute of an anchor tag (`<a>`), which is an inline tag. The anchor tag that specifies a link is called the *source* of that link. The document whose address is specified in a link is called the *target* of that link.

As is the case with many tags, the anchor tag can include many different attributes. However, for creating links only one is required, `href` (an acronym for hypertext `reference`). The value assigned to `href` specifies the target of the link. If the target is in another document in the same directory, the target is just the document's filename. If the target document is in some other directory, the UNIX pathname conventions are used. So, an XHTML file named `c210data.html` in a subdirectory of the directory in which the source XHTML file—say, named `airplanes`—is specified in the `href` attribute as `airplanes/c210data.html`. This is the relative method of document addressing. Absolute file addresses could be used in which the entire pathname for the file is given. However, relative links are easier to maintain, especially if a hierarchy of XHTML files must be moved. If the document is on some other machine (not on the server providing the document that includes the link), the complete URL obviously must be used.

```
<?xml version = "1.0" encoding = "utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
  "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<!-- link.html
     An example to illustrate a link
     -->
<html xmlns = "http://www.w3.org/1999/xhtml">
  <head> <title> A link </title>
  </head>
  <body>
    <h1> Aidan's Airplanes </h1>
    <h2> The best in used airplanes </h2>
    <h3> "We've got them by the hangarful" </h3>
    <h2> Special of the month </h2>
    <p>
     1960 Cessna 210 <br />
     <a href = "C210data.html"> Information on the Cessna 210 </a>
    </p>
  </body>
</html>
```

Links can include images in their content, in which case the browser displays the image with the link:

```
<a href = "c210data.html" >
  <img src = "small-airplane.jpg"
       alt = "An image of a small airplane" />
    Information on the Cessna 210
</a>
```

Targets within Documents

If the target of a link is not at the beginning of a document, it must be some element within the document, in which case there must be some means of specifying it. The target element can include an id attribute, which can then be used to identify it in an href attribute. Consider following example

<h2 id=”avionics”> Avionics</h2>

The target is specified in the href attribute value by preceding the id value with a pound sing (#), as in the following example

<a href=”#avionics”>What about avionics? </a>

When target is a part or fragment of another document, the name of the part is specified at the end of URL, separated by pound sign(#)

<a href=”AIDAN1.html#avionics”>Avionics</a>

Module 2

Q3 a) Mention any five CSS selectors and explain their uses with a suitable example (10 Marks)

1) Simple Selector Forms:

In case of simple selector, a tag is used. If the properties of the tag are changed, then it reflects at all the places when used in the program. The selector can be any tag. If the new properties for a tag are not mentioned within the rule list, then the browser uses default behaviour of a tag.

Eg:

```
h1 { font-size : 24pt; }

h2, h3{ font-size : 20pt; }

body b em { font-size : 14pt; }
```

Only applies to the content of ‘em’ elements that are descendent of bold element in the body of the document. This is a contextual selector

2) Class Selectors:

Class selectors are used to allow different occurrences of the same tag to use different style specifications.

Eg

```
<head>

<style type = "text/css">

p.one { font-family: 'Lucida Handwriting'; font-size: 25pt; color: Red; }
```

```
p.two{ font-family: 'Monotype Corsiva'; font-size: 50pt; color: green; }

</style>

</head>

<body>

<p class = "one">Web Technology</p>

<p class = "two">Web Technology</p>

</body>
```

3) Generic Selectors:

Sometimes it is convenient to have a class of Style specification that applies to the content of more than one kind of tag. This is done by using a generic class, which is defined without a tag name in its name. In place of the tag name, you use the name of the generic class, which must begin with a period.

Eg

```
<head>

<style type = "text/css">

.sale{ font-family: 'Monotype Corsiva'; color: green; }

</style>

</head>

<body>

<p class = "sale">Weekend Sale</p>

<h1 class = "sale">Weekend Sale</h1>

<h6 class = "sale"> Weekend Sale</h6>

</body>
```

4) id Selectors:

An id selector allows the application of a style to one specific element.

Eg:

```
<head>

<style type = "text/css">

#one { font-family: 'Lucida Handwriting'; font-size: 25pt; color: Red; }

#two { font-family: 'Monotype Corsiva'; font-size: 50pt; color: green; }

</style>

</head>

<body>

<p id = "one">Web Technology</p>

<p id = "two">Web Technology</p>

</body>
```

5) Universal Selectors:

The universal selector, denoted by an asterisk (*), applies its style to all elements in a document.

```
<head>

<style type = "text/css">

*{ font-family: 'Lucida Handwriting'; font-size: 25pt; color: Red; }

</style>

</head>

<body>

<p>Web Technology</p>

<p>Web Technology</p>

</body>
```

Q3 b) Discuss on the different ways of including CSS sytle information to a HTML document (10 Marks)

The format of style specification depends on the level of stylesheet.

Inline Style Specification: appears as values of the style attribute of a tag, the general form is as follows:

Style = “Property1 : Value1; Property2 : Value2; Property3 : Value3; .... Property_n:Value_n;”

It is recommended that last property/value pair be followed by a semicolon.

Eg:

```
<h1 style ="font-family: 'Lucida Handwriting'; font-size: 50pt; color: Red;">Web Technology</h1>
```

Document Style Specification: appears as the content of a style element within the

header of a document, general form of the content of a style element is as follows:

```
<style type = “text/css”>
Rule list
</style>
```

Each style rule in a rule list has two parts: a selector, which indicates the tag or tags affected

by the rule, and a list of property–value pairs. The list has the same form as the quoted list

for inline style sheets, except that it is delimited by braces rather than double quotes. So,

the form of a style rule is as follows:

Selector { Property1 : Value1; Property2 : Value2; Property3 : Value3; ....

Property_n:Value_n; }

Eg:

```
<style type = "text/css">
h1 {
font-family: 'Lucida Handwriting';
font-size: 50pt;
color: Red;
}
```

```
</style>
```

External Style Sheet: have a form similar to that of document style sheets. The external

file consists of a list of style rules.

Eg

```
<head>

<link rel=”stylesheet” type=”text/css” href=”cssfile.css”>

</head>
```

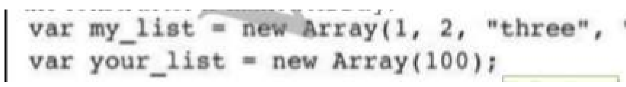
Cssfile.css

```
P{

Background-color:blue;

Color:red;

}
```

Q4 a) what is an Array in Javascript? Explain the various ways creating arrays Mention any 5 array methods and explain their use. (05 Marks)

Array OBJECT CREATION

The usual way to create any object is with the `new` operator and a call to a constructor. In the case of arrays, the constructor is named `Array`:



```
var my_list = new Array(1, 2, "three", "four");
var your_list = new Array(100);
```

The second way to create an `Array` object is with a literal array value, which is a list of values enclosed in brackets: **`var my_list_2 = [1, 2, "three", "four"];`**

CHARACTERISTICS OF Array OBJECTS

The lowest index of every JavaScript array is zero. Access to the elements of an array is specified with numeric subscript expressions placed in brackets. The length of an array is the highest subscript to which a value has been assigned, plus 1. For example, if `my_list` is an array with four elements and the following statement is executed, the new length of `my_list` will be 48.

`my_list[47] = 2222;`

The length of an array is both read and write accessible through the `length` property, which is created for every array object by the `Array` constructor.

For example, **`my_list.length = 1002;`**

An array is lengthened by setting its `length` property to a larger value, shortened by setting its `length` property to a smaller value. The next example, `insert_names.js`, illustrates JavaScript arrays. This script has an array of names, which are in alphabetical order. It uses `prompt` to get new names, one at a time, and inserts them into the existing array. Notice that each new name causes the array to grow by one element.

Array METHODS

Array objects have a collection of useful methods, most of which are described in this section.

- The **join** method converts all of the elements of an array to strings and catenates them into a single string. If no parameter is provided to join, the values in the new string are separated by commas. If a string parameter is provided, it is used as the element separator. Consider the following example:

```
var names = new Array["Mary", "Murray",
                      "Murphy", "Max"];
...
var name_string = names.join(" : ");
```

The value of `name_string` is now `"Mary : Murray : Murphy : Max"`.

- The **reverse** method reverses the order of the elements of the Array object through which it is called.

- The **sort** method coerces the elements of the array to become strings if they are not already strings and sorts them alphabetically
- The **concat** method catenates its actual parameters to the end of the Array object on which it is called.

```
var names = new Array["Mary", "Murray",
                      "Murphy", "Max"];
...
var new_names = names.concat("Moo", "Meow");
```

The `new_names` array now has length 6, with the elements of `names`, along with `"Moo"` and `"Meow"` as its fifth and sixth elements.

- The **`slice`** method does for arrays what the `substring` method does for strings, returning the part of the `Array` object specified by its parameters, which are used as subscripts. The array returned has the elements of the `Array` object through which it is called, from the first parameter up to, but not including, the second parameter.

```
var list = [2, 4, 6, 8, 10];
...
var list2 = list.slice(1, 3);
```

- The value of `list2` is now `[4, 6]`. If `slice` is given just one parameter, the array that is returned has all of the elements of the object, starting with the specified index.

Q4 b) Write javascript program that accepts a value ‘n’ as input from the user and display the first ‘n’ finanocci numbers as output (05 Marks)

```
<html lang="en">

       <head>

              <meta charset="UTF-8">

              <title>Lab3a</title>

       </head>

  <body>

    <script type="text/javascript">

      //initialize variables

      var fib1=0,fib2=1,fib=0;

      var n=prompt("Enter a number");

      if(n!=null && n>0)

        {

          document.write("<h1>First " + n + " Fibonacci numbers are: </h1><br>");

          //if input is one number

          if(n==1)

            document.write("<p>" + fib1 + "</p><br/>");

          //if input is two numbers

          else

            document.write("<p>" + fib1 + "</p><br/><p>" + fib2 + "</p><br/>");

            //if input is more than two numbers, find the next Fibonacci number

            for(i=3;i<=n;i++)

              {

                fib=fib1+fib2;

                document.write("<p>" + fib + "</p><br/>");
```

```
                        fib1=fib2;
                        fib2=fib;
                      }
                }
            else
                alert("Please enter proper input value");
        </script>
    </body>
</html>
```

Q4 C) Write a Javascript program that accepts valid USN as input and displays an error message otherwise. USN format 'XX00 XXX00' where 'X' is an upper case alphabet, '0' is a digit (05 Marks)

```
<!DOCTYPE html>
<html>
        <head>
                <script type="text/javascript">
                        function funValidate(){
                                var usn=document.getElementById('usn').value;
                                var pattern=/^[A-Z]{2}[0-9]{2}[A-Z]{3}[0-9]{2}$/;

                                if(usn.match(pattern))
                                {
                                        alert("Valid Format");
                                }
                                else
                                {
                                    alert("Invalid Format");
                              }
                        }

                  </script>
            </head>
            <body>
                  <label>Enter your USN: <input type="text" name="usn" id="usn" /></label>
                  <input type="submit" name="validate" onClick="funValidate()" />
            </body>
</html>
```

Module 3

Q5 a) what is Bootstrap? What are the features of Bootstrap? Crate a simple web page using Bootstrap (10 Marks)

Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains CSS- and (optionally) JavaScript-based design templates for typography, forms, buttons, navigation and other interface components.

Features 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-Friendly

Mobile-first approach: In Bootstrap 3, mobile-first styles are part of the core framework

Simple Integration

Bootstrap can be simply integrated along with distinct other platforms and frameworks, on existing sites and new ones too and one more things you can also utilize particular elements of Bootstrap along with your current CSS.

Pre-styled Components

Bootstrap approaches with pre-styled components for alerts, dropdowns, nav bars, etc.

Customizable Bootstrap

The Bootstrap can be customized as per the designs of your project.

Browser compatibility

Bootstrap is compatible with all modern browsers (Chrome, Firefox, Internet Explorer, Safari, and Opera)

Great grid system

Bootstrap is built on responsive 12-column grids, layouts and components. Whether you need a fixed grid or a responsive, it's only a matter of a few changes.

Bundled JavaScript plugins

The components such as drop down menu are made interactive with the numerous JavaScript plugins bundled in the bootstrap package.

Extensive list of components

Whether you need drop down menus, pagination or alert boxes, Bootstrap has got your covered. Some of the components pre styled are; Dropdowns, Button Groups, Navigation Bar, Breadcrumbs, Labels & Badges, Alerts, Progress Bar, And many others.

Base styling for most HTML elements

A website has many different elements such as headings, lists, tables, buttons, forms, etc. The HTML elements for which styles are provided are; Typography Code, Tables, Forms, Buttons, Images, Icons

Good documentation

Not only does Bootstrap offer styling for almost every element a typical website or web application requires, it also provides a great documentation with examples and demo that only make it more easier for even someone new.

```
<!DOCTYPE html>
```

```
<html>

            <head>

                        <title>Bootstrap 101 Template</title>

 <link href="css/bootstrap.min.css" rel="stylesheet">

            </head>

 <body>

 <h1>Hello, world! </h1>

 <script src="js/bootstrap.min.js"></script>

 </body>

</html>
```

Q5 b) Discuss on the following: (10 Marks)

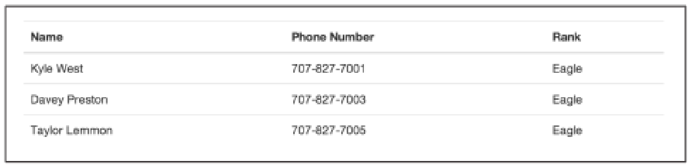
- **i) Table**
- **ii) Image**
- **iii) Button**
- **iv) Progress Bar**

Use Bootstrap Code snippet

Table

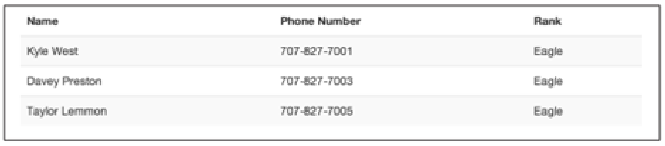
If you want a nice, basic table style with just some light padding and horizontal dividers, add the base class of .table to any table (see Figure 2-13). The basic layout has a top border on all of the <td> elements:

```
<table class="table">
  <caption>...</caption>
  <thead>
    <tr>
      <th>...</th>
      <th>...</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>...</td>
      <td>...</td>
    </tr>
  </tbody>
</table>
```



Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

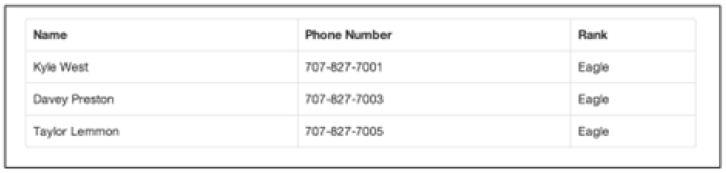
Optional Table Classes
Along with the base table markup and the .table class, there are a few additional classes that you can use to style the markup. These four classes are: .table-striped, .tablebordered, .table-hover, and .table-condensed. Striped table By adding the .table-striped class, you will get stripes on rows within the <tbody> (see Figure 2-14). This is done via the CSS :nth-child selector, which is not available on Internet Explorer 7–8



Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 2-14. Striped table class

Bordered table
If you add the .table-bordered class, you will get borders surrounding every element and rounded corners around the entire table, as shown in Figure 2-15

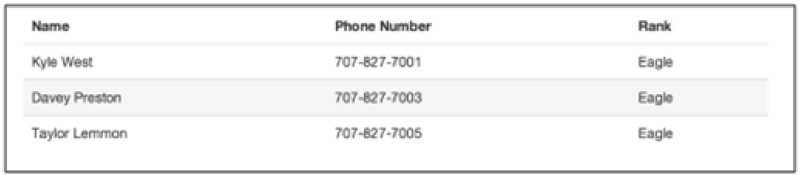


Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 2-15. Bordered table class

Hover table

Figure 2-16 shows the `.table-hover` class. A light gray background will be added to rows while the cursor hovers over them.

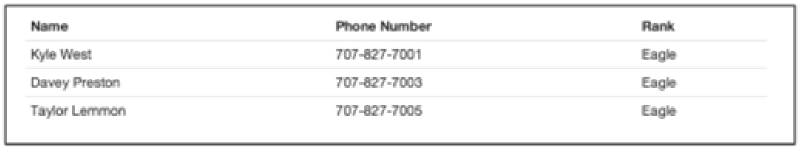


Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 2-16. Hover table class

Condensed table

If you add the `.table-condensed` class, as shown in Figure 2-17, row padding is cut in half to condense the table. This is useful if you want denser information.



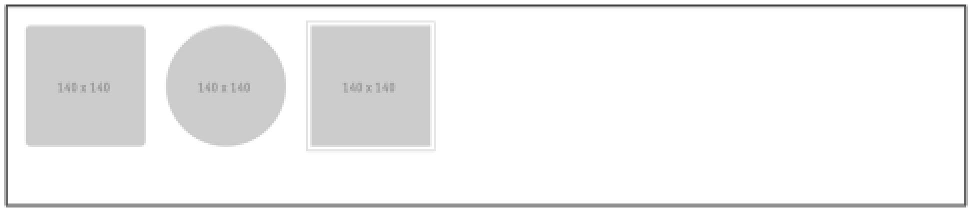
Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 2-17. Condensed table class

Images

Images have three classes that can be used to apply some simple styles: .img-rounded adds border-radius:6px to give the image rounded corners, .img-circle makes the entire image round by adding border-radius: 500px, and .img-polaroid adds a bit of padding and a gray border:

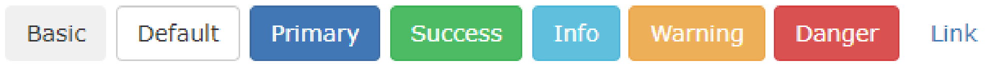
```
<img src="..." class="img-rounded">
<img src="..." class="img-circle">
<img src="..." class="img-polaroid">
```



Button

Button Styles

Bootstrap provides different styles of buttons:



To achieve the button styles above, Bootstrap has the following classes:

- `.btn`
- `.btn-default`
- `.btn-primary`
- `.btn-success`
- `.btn-info`
- `.btn-warning`
- `.btn-danger`
- `.btn-link`

The following example shows the code for the different button styles:

```
<button type="button" class="btn">Basic</button>
<button type="button" class="btn btn-default">Default</button>
<button type="button" class="btn btn-primary">Primary</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-link">Link</button>
```

Progress Bar

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

Bootstrap provides several types of progress bars.

A default progress bar in Bootstrap looks like this:

To create a default progress bar, add a `.progress` class to a `<div>` element:

```
<div class="progress">
  <div class="progress-bar" role="progressbar" aria-valuenow="70"
  aria-valuemin="0" aria-valuemax="100" style="width:70%">
    <span class="sr-only">70% Complete</span>
  </div>
</div>
```

Q6 a) Discuss briefly on the Grid system of Bootstrap (10Marks)

Default Grid System

The default Bootstrap grid (see Figure 1-1) system utilizes 12 columns, making for a 940px-wide container without responsive features enabled. With the responsive CSS file added, the grid adapts to be 724px or 1170px wide, depending on your viewport. Below 767px viewports, such as the ones on tablets and smaller devices, the columns become fluid and stack vertically. At the default width, each column is 60 pixels wide and offset 20 pixels to the left. An example of the 12 possible columns is in Figure 1-1

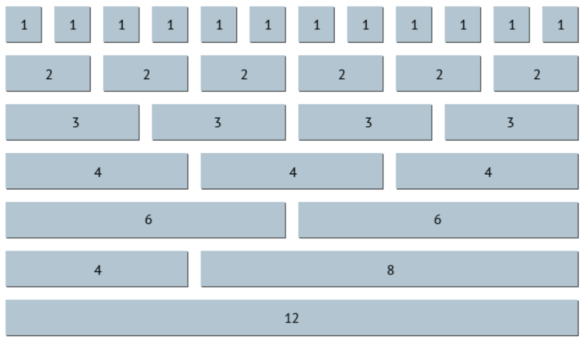


Figure 1-1. Default grid

Basic Grid HTML

To create a simple layout, create a container with a <div> that has a class of .row and add the appropriate amount of .span* columns. Since we have a 12-column grid, we just need the amount of .span* columns to equal 12. We could use a 3-6-3 layout, 4-8, 3-5-4, 2-8-2… we could go on and on, but I think you get the gist.
The following code shows .span8 and .span4, which adds up to 12:

```
<div class="row">
<div class="span8">...</div>
<div class="span4">...</div>
</div>
```

Offsetting Columns

You can move columns to the right using the .offset* class. Each class moves the span over that width. So an .offset2 would move a .span7 over two columns (see Figure 1-2):

```
<div class="row">
<div class="span2">...</div>
<div class="span7 offset2">...</div>
</div>
```

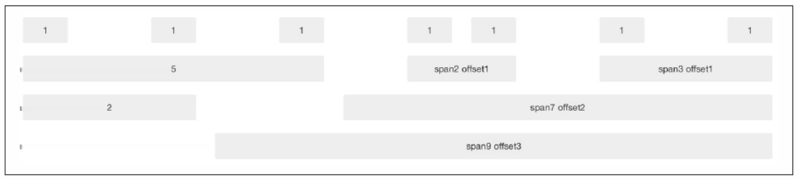


Figure 1-2. Offset grid

Nesting Columns

To nest your content with the default grid, inside of a .span*, simply add a new .row with enough .span* that it equals the number of spans of the parent container (see Figure 1-3):

```
<div class="row">
  <div class="span9">
    Level 1 of column
    <div class="row">
      <div class="span6">Level 2</div>
      <div class="span3">Level 2</div>
    </div>
  </div>
</div>
```

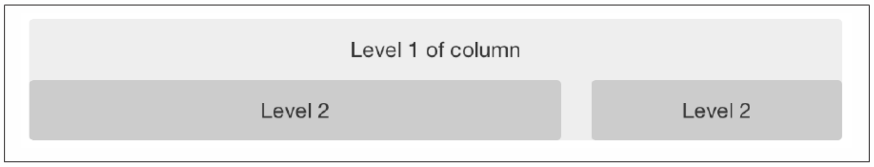


Figure 1-3. Nesting grid

Fluid Grid System

The fluid grid system uses percentages instead of pixels for column widths. It has the same responsive capabilities as our fixed grid system, ensuring proper proportions for key screen resolutions and devices. You can make any row "fluid" by changing .row to .row-fluid. The column classes stay exactly the same, making it easy to flip between fixed and fluid grids. To offset, you operate in the same way as the fixed grid system— add .offset* to any column to shift by your desired number of columns:

```
<div class="row-fluid">
  <div class="span4">...</div>
  <div class="span8">...</div>
</div>

<div class="row-fluid">
  <div class="span4">...</div>
  <div class="span4 offset2">...</div>
</div>
```

Nesting a fluid grid is a little different. Since we are using percentages, each .row resets the column count to 12. For example, if you were inside a .span8, instead of two .span4 elements to divide the content in half, you would use two .span6 divs (see Figure 1-4). This is the case for responsive content, as we want the content to fill 100% of the container:

```html
<div class="row-fluid">
  <div class="span8">
              <div class="row">
                      <div class="span6">...</div>
                      <div class="span6">...</div>
              </div>
  </div>
</div>
```

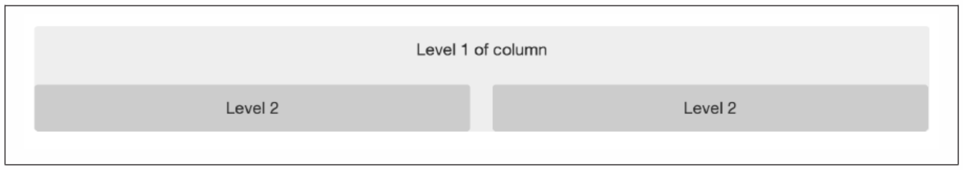


Figure 1-4. Nesting fluid grid

Q6 b) List out the various types of forms in Bootstrap. Explain their use with code snippet.

Bootstrap Form Layouts

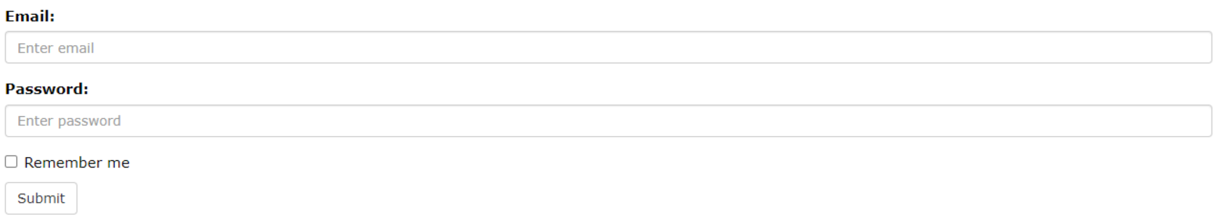
Bootstrap provides three types of form layouts:

- Vertical form (this is 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 `.form-control` to all textual `<input>`, `<textarea>`, and `<select>` elements

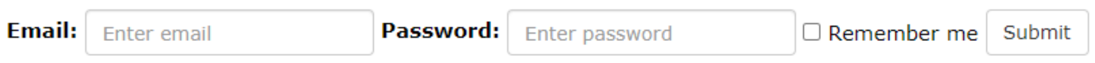
Bootstrap Vertical Form (default)



The following example creates a vertical form with two input fields, one checkbox, and a submit button:

```
<form action="/action_page.php">
  <div class="form-group">
    <label for="email">Email address:</label>
    <input type="email" class="form-control" id="email">
  </div>
  <div class="form-group">
    <label for="pwd">Password:</label>
    <input type="password" class="form-control" id="pwd">
  </div>
  <div class="checkbox">
    <label><input type="checkbox"> Remember me</label>
  </div>
  <button type="submit" class="btn btn-default">Submit</button>
</form>
```

Bootstrap Inline Form



In an inline form, all of the elements are inline, left-aligned, and the labels are alongside.

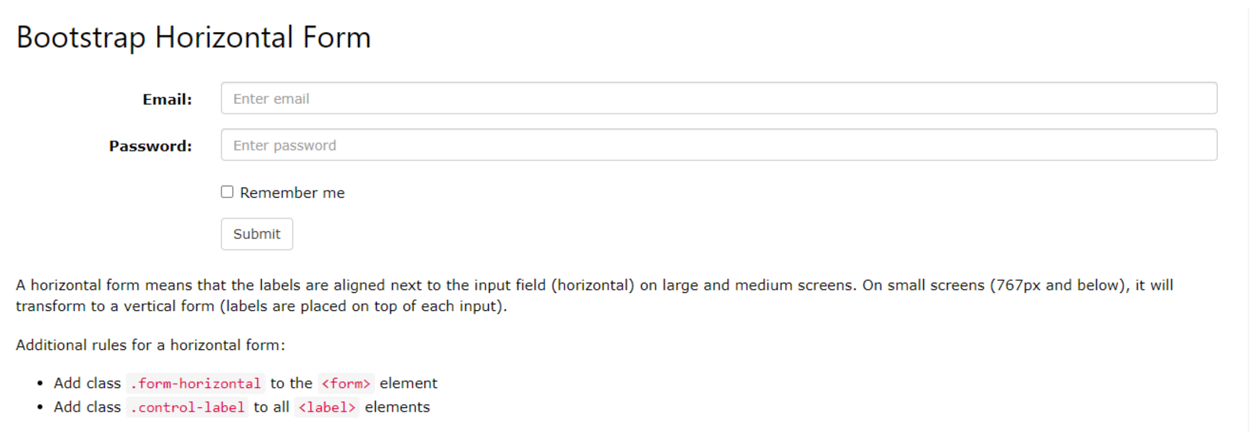
Note: This only applies to forms within viewports that are at least 768px wide!

Additional rule for an inline form:

- Add class `.form-inline` to the `<form>` element

The following example creates an inline form with two input fields, one checkbox, and one submit button:

```
<form class="form-inline" action="/action_page.php">
  <div class="form-group">
    <label for="email">Email address:</label>
    <input type="email" class="form-control" id="email">
  </div>
  <div class="form-group">
    <label for="pwd">Password:</label>
    <input type="password" class="form-control" id="pwd">
  </div>
  <div class="checkbox">
    <label><input type="checkbox"> Remember me</label>
  </div>
  <button type="submit" class="btn btn-default">Submit</button>
</form>
```


Bootstrap Horizontal Form

Email: Enter email

Password: Enter password

☐ Remember me

Submit

A horizontal form means that the labels are aligned next to the input field (horizontal) on large and medium screens. On small screens (767px and below), it will transform to a vertical form (labels are placed on top of each input).

Additional rules for a horizontal form:

- Add class `.form-horizontal` to the `<form>` element
- Add class `.control-label` to all `<label>` elements

```
<form class="form-horizontal" action="/action_page.php">
  <div class="form-group">
    <label class="control-label col-sm-2" for="email">Email:</label>
    <div class="col-sm-10">
      <input type="email" class="form-control" id="email" placeholder="Enter email">
    </div>
  </div>
  <div class="form-group">
    <label class="control-label col-sm-2" for="pwd">Password:</label>
    <div class="col-sm-10">
      <input type="password" class="form-control" id="pwd" placeholder="Enter password">
    </div>
  </div>
  <div class="form-group">
    <div class="col-sm-offset-2 col-sm-10">
      <div class="checkbox">
        <label><input type="checkbox"> Remember me</label>
      </div>
    </div>
  </div>
  <div class="form-group">
    <div class="col-sm-offset-2 col-sm-10">
      <button type="submit" class="btn btn-default">Submit</button>
    </div>
  </div>
</form>
```

Module 4

Q7 a) what is Jquery? What are the advantages of JQuery?

• jQuery is a JavaScript library designed to simplify HTML DOM tree traversal and manipulation, as well as event handling, CSS animation, and Ajax.

• jQuery is a lightweight, "write less, do more", JavaScript library.

• Using raw JavaScript can result in dozens of lines of code.

• The creators of jQuery specifically created the library to make common tasks trivial.

• The real power in this jQuery statement comes from the selector, an expression for identifying target elements on a page that allows us to easily identify and grab the elements we need.

Advantages of jQuery

- Using raw JavaScript can result in dozens of lines of code for each of these tasks.
- The creators of jQuery specifically created the library to make common tas ks trivial. For example, designers will use JavaScript to "zebra-stripe" tables— highlighting every other row in a table with a contrasting color—taking up to 10 lines of code or more. Here's how we accomplish it using jQuery:
  `$("table tr:nth-child(even)").addClass("striped");`
- jQuery statements to make your pages come alive.
- The real power in this jQuery statement comes from the selector, an expression for identifying target elements on a page that allows us to easily identify and grab the elements we need

Q7 b) Explain the syntax of JQuery script with a suitable example

The jQuery wrapper

To collect a group of elements, we use the simple syntax

$(selector)

or

jQuery(selector)

For example, to retrieve the group of links nested inside a <p> element, we use the following

$("p a")

The $() function (an alias for the jQuery() function) returns a special Java-Script object containing an array of the DOM elements that match the selector.

The document ready handler

jQuery provides a simple means to trigger the execution of code once the DOM tree, but not external image resources, has loaded. The formal syntax to define such code is as follows:

Syntax

```
$(document).ready(function() {
//jquery statements
});
```

Example

```
$(document).ready(function() {
$("table tr:nth-child(even)").addClass("even");
});
```

First, we wrap the document instance with the jQuery() function, and then we apply the ready() method, passing a function to be executed when the document is ready to be manipulated

Q7 c) Write a JQuery program to animate a rectangular Box

```
<html>
  <head>
    <title>lab14b</title>
    <script type="text/javascript"
            src="jquery-1.2.1.js"></script>

    <script type="text/javascript">
      $(function(){

        $('#btn1').click(function(){
          $('#box').each(function(){
            $(this).animate(
              {
                width: $(this).width() * 2,
                height: $(this).height() * 2
              },
              'slow'
            );
          });
        });
                        $( "#btn2" ).click(function() {
                                $('#box').each(function(){
                                        $(this).animate(
                                          {
                                                width: "100px",
                                          height: "100px"
                                          },
                                         'slow'
                                        );
                                });
                        });

      });
    </script>
  </head>

  <body>
      <div>
                        <button id="btn1">Animate</button>
                        <button id="btn2">Reset</button>
                        <div id="box"
style="background:#B40F04;height:100px;width:100px;margin:6px;"></div>
      </div>
  </body>
</html>
```

Q8 a) Mention any ten JQuery selectors and explain their use with code snippet. (10Marks)

Using basic CSS selectors

 a—This selector matches all link (<a>) elements.

 #specialID—This selector matches elements that have an id of specialID.

 .specialClass—This selector matches elements that have the class of specialClass.

 a#specialID.specialClass—This selector matches links with an id of specialID and a class of special Class.

 p a.specialClass—This selector matches links with a class of specialClass declared within <p> elements

Child selector

A more advanced approach is to use child selectors, in which a parent and its direct child are separated by the right angle bracket character (>), as in

p >a

Attribute selectors are also extremely powerful.

a[href^=http://]

form[method]

input[type=text]

div[title^=my]

a[href$=.pdf]

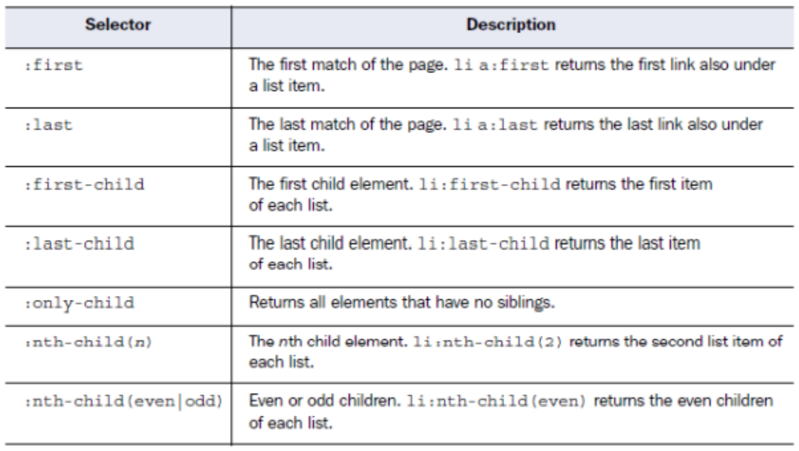
a[href*=jquery.com]

container selector

li:has(a)

This selector matches all <li> elements that contain an <a> element.

Selecting by position



Selector	Description
`:first`	The first match of the page. `li a:first` returns the first link also under a list item.
`:last`	The last match of the page. `li a:last` returns the last link also under a list item.
`:first-child`	The first child element. `li:first-child` returns the first item of each list.
`:last-child`	The last child element. `li:last-child` returns the last item of each list.
`:only-child`	Returns all elements that have no siblings.
`:nth-child(n)`	The *n*th child element. `li:nth-child(2)` returns the second list item of each list.
`:nth-child(even|odd)`	Even or odd children. `li:nth-child(even)` returns the even children of each list.

Q8 b) Demonstrate the use of any five JQuery method to manipulate HTML elements with code snippet. (05 Marks)

Q8 c) Write a JQuery program that perfoms fading in and fading out of the HTML element. (05 Marks)

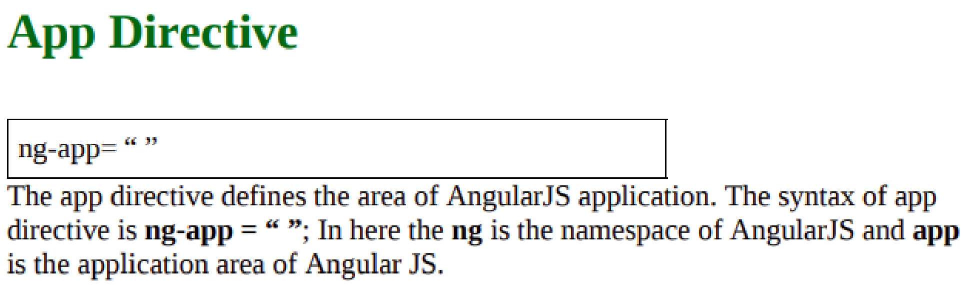
```
<!DOCTYPE html>
<html>
        <head>
                <script type="text/javascript" src="jquery-1.2.1.js"></script>
                <script type="text/javascript">
                        $(document).ready(function(){
                         $("#btn1").click(function(){
                                $("div").fadeOut(3000);
                         });
                          $("#btn2").click(function(){
                                $("div").fadeIn(3000);
                         });
                        });
                </script>
                <title>lab14a</title>
        </head>
        <body align="center">
                <button id="btn1">Fade Out</button>
                <button id="btn2">Fade In</button>
                <br/><br/>
                <div style="background:#2E9AFE;">Div One</div><br/>
                <div style="background:#2E900E;">Div Two</div>
        </body>
</html>
```

Module 5

Q9 a) what is an angular JS Directive? Explain the use of the following directives with code snippet. (08 Marks)

The directives of AngularJS

The AngularJS allows us to extend HTML in very simple way using attributes. The attributes are basically directives. There are different types of directives which can play different roles in Application. They are App Directive, Model Directive, Bind Directive, Init Directive, and Repeat Directive. Let`s discuss one by one in detail



App Directive

```
ng-app= " "
```

The app directive defines the area of AngularJS application. The syntax of app directive is **ng-app = " "**; In here the **ng** is the namespace of AngularJS and **app** is the application area of Angular JS.

Model Directive

```
ng-model = “data”
```

The model directive is used to bind the inputted value from HTML controls (input, checkbox and select etc.) to application data. The **ng-model = “data”** is the syntax of model directive. Let`s take an example for better understanding.

Example 2.2

```
<!DOCTYPE html>
<html >
<head>
   <title>AngularJSfor beginners</title> <script src="js\angular.min.js">
</script>
</head>
<body>
<div ng-app=""> <p>User Name: <br> <input type="text" ng-model =
"Username"></p> </div>
</body>
```

Bind Directive

```
<p>ng-bind = “data”</p>
```

The bind directive is used to bind the data value to an html element <p>; the syntax of bind directive is **<p>ng-bind = “data”</p>**. Let`s take an example for better understanding.

Example 2.3

```
<!DOCTYPE html>
<html >
<head>
   <title>AngularJSfor beginners</title> <script src="js\angular.min.js">
</script> </head>
<body>
<div ng-app=""> <p>User Name: <br> <input type="text" ng-model =
"Username"></p> <p ng-bind ="Username"></p> </div>
</body>
</html>
```

Open the notepad and paste the above mentioned code with .html extension, and type username “Ray Yao” in the input box.

Init Directive

```
ng-init = "data = 'value'"
```

The init directive is used to initialize the data with a value. The syntax of init directive is **ng-init = "data = 'value'"**. Let`s take an example for better understanding.

Example 2.4

```
<!DOCTYPE html>
<html >
<head>
   <title>AngularJSfor beginners</title> <script src="js\angular.min.js">
</script> </head>
<body>
<div ng-app=""    ng-init="Username= 'Andy Smith' "> <p>User Name:
<input type="text" ng-model = "Username"></p> <p ng-
bind="Username"></p> </div>
</body>
</html>
```

Q9 b) What is the use of a filter in Angular Js? Explain the use of JSON, Currency, Number, Lowercase.

(08 Marks)

Filter is used to format the value of data. The pipe sign (|) indicates that filter is

used. The proper syntax of filter looks like this:

Value | filter

Let`s try to understand the filers one by one

JSON Filter

The **json** filter in AngularJs is used to convert a JavaScript object into a JSON. string.JavaScript object that we are using can be of any kind of JavaScript Object. The json filter piped the object or any expression with JSON so that the result will be displayed in the form of a list, which is bound with the expression syntax.

Syntax:

```
{{ object | json : spacing }}
```

Parameter value:

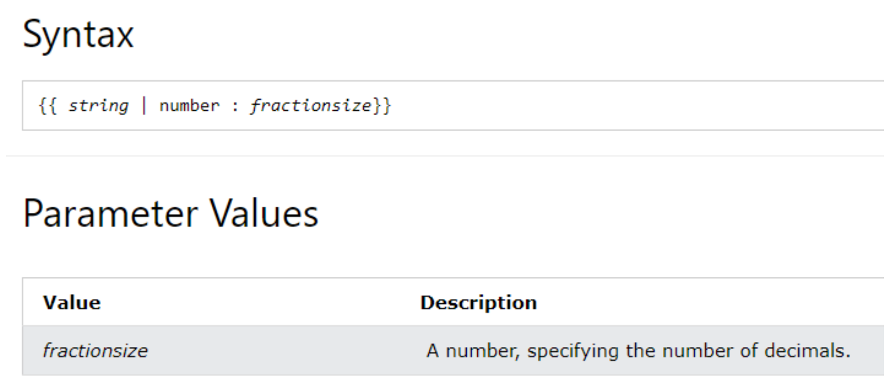
json: It is used to specify that the object should be displayed in JSON format.

spacing: It is an optional parameter with a default value of 2 that specifies the number of spaces per indentation.

```
    <body style="text-align: center;">
        <div ng-app="result" ng-controller="resultCtrl">
            <h1 style="color:green ;">GeeksforGeeks</h1>
            <h3>AngularJS json Filter</h3>
            <h4>Result:</h4>
            <pre>{{marks | json}}</pre>
        </div>

        <script>
            var app = angular.module('result', []);
            app.controller('resultCtrl', function ($scope) {
                $scope.marks = {
                    "Math": 98,
                    "Computer": 93,
                    "Physics": 95,
                    "Chemistry": 95,
                    "English": 74
                };
            });
        </script>
    </body>
    </html>
```

Number Filter

The `number` filter formats a number to a string.



Syntax

```
{{ string | number : fractionsize}}
```

Parameter Values

Value	Description
fractionsize	A number, specifying the number of decimals.

```
<div ng-app="myApp" ng-controller="nCtrl">

<h1>{{prize | number}}</h1>

</div>

<script>
var app = angular.module('myApp', []);
app.controller('nCtrl', function($scope) {
    $scope.prize = 1000000;
});
</script>
```

Lowercase filter

```
Value | lowercase
```

The lowercase filter changes the text to lower case. Suppose a user writes a text in upper case (e.g. RAY YAO) or title case (e.g. Ray Yao) or in mixed case (e.g. rAy or RaY or rAY etc.), and you want the lower case result, then you will have to use lower case filter.

Example 3.2

```
<html >
<head>
   <title>AngularJSfor beginners</title> <script src="js\angular.min.js">
</script> </head>
<body>
<h3>Using Lower Case Filter</h3> <div ng-app=""  ng-init="Username=
'Ray YAO' "> <p>User Name: <input type="text" ng-model="Username">
</p> <p style="color:red" ng-bind="Username | lowercase"></p> </div>
</body>
</html>
```

Currency filter

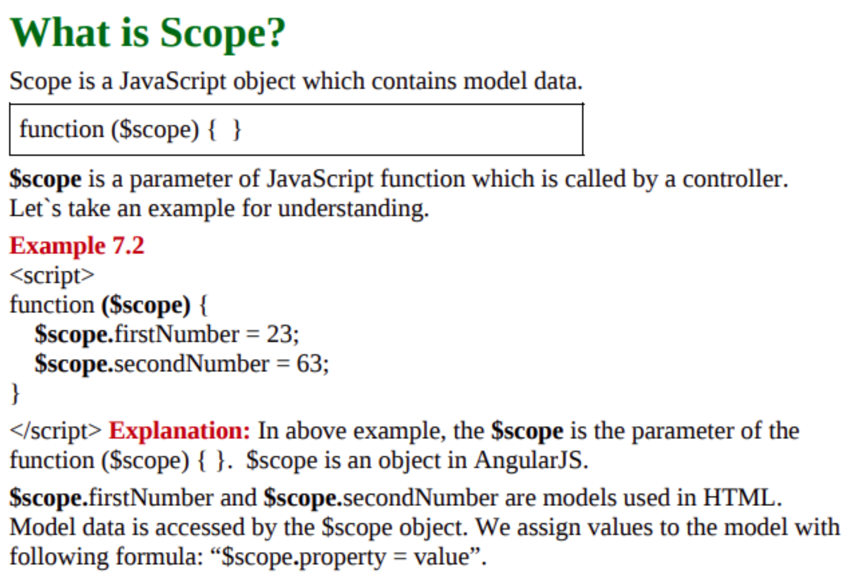
```
Value | currency
```

The currency filter is used to display the result in currency format.

Example 3.5

```
<!DOCTYPE html>
<html >
<head>
   <title>AngularJSfor beginners</title> <script src="js\angular.min.js">
</script> <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/> </head>
<body>
<h1>Using Currency filter</h1> <div ng-app="" ng-init =
"Employees_Monthly_Salary=[{name: 'Jay', salary:8100}, {name: 'Sdwt',
salary:7000},              {name: 'Hao', salary:9000}, {name: 'Luoe',
salary:6300}, {name: 'Fin', salary:9800}]"> <table border="1" > <tr>
<th>Employee Name</th> <th>Employee Salary</th> </tr>
<tr ng-repeat="x in Employees_Monthly_Salary  "> <td ng-bind="x.name
"></td> <td ng-bind="x.salary | currency "></td> </tr>
</table>
</div>
</body>
</html>
```

Q9 c) Demonstrate $ scope with an example (04 Marks)



What is Scope?

Scope is a JavaScript object which contains model data.

```
function ($scope) {  }
```

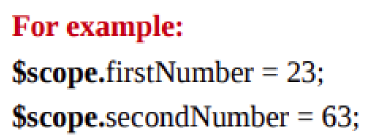
$scope is a parameter of JavaScript function which is called by a controller.
Let`s take an example for understanding.

Example 7.2

```
<script>
function ($scope) {
   $scope.firstNumber = 23;
   $scope.secondNumber = 63;
}
</script>
```

Explanation: In above example, the **$scope** is the parameter of the function ($scope) { }. $scope is an object in AngularJS.

$scope.**firstNumber and **$scope.secondNumber are models used in HTML. Model data is accessed by the $scope object. We assign values to the model with following formula: “$scope.property = value”.



For example:

$scope.firstNumber = 23;

$scope.secondNumber = 63;

Q10 a) Write an Angular JS program to demonstrate client side form validation (10 Marks)

```
<!DOCTYPE html>
<html>

<head>
      <title>AngularJS Form Validation</title>
    <script src=
"https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js">
    </script>
    <style>
        body {
            font-family: Arial, Helvetica, sans-serif;
        }
        h1 {
            color: green;
        }
    </style>
</head>

<body ng-app="">
    <h1>GeeksforGeeks</h1>
    <h3>AngularJS Form Validation</h3>
    <form name="form1">
        <p>Name:
            <input name="username"
                   ng-model="username" required>
            <span ng-show=
"form1.username.$pristine && form1.username.$invalid">
                 The name is required.</span>
        </p>
        <p>Address:
            <input name="useraddress"
                   ng-model="useraddress" required>
        </p>
    </form>
    <p>
        We use the ng-show directive to only
        show the error message <br>if the field
          has not modified yet AND content present
          in the field is invalid.
    </p>
</body>
```

Q10 b) With code snippet, explain the use of controllers in Angular JS (10Marks)

What is Controller?

The controller is basically a JavaScript object that controls the flow of data of an application. So the AngularJS application is controlled by Controller.

How to define Controller?

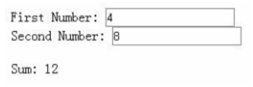
The **ng-controller** directive is used to define the Controller. We know that Controller is a JavaScript object which contains JavaScript function and properties. The syntax of Controller is as following:

```
<div ng-app=""  ng-controller="controllerName">
```

Example 7.1

```
<html>
<script src= "js\angular.min.js"></script> <body>
<div ng-app="Calculation" ng-controller="myController"> First Number:
<input type="number" ng-model="firstNumber"><br> Second Number:
<input type="number" ng-model="secondNumber"><br> <br>
Sum: {{firstNumber + secondNumber}}
</div>
<script>
var app = angular.module('Calculation', [ ]); app.controller('myController',
function($scope) {
   $scope.firstNumber = 4; $scope.secondNumber = 8; });
</script>
</body>
</html>
```

Output:



```
First Number: 4
Second Number: 8

Sum: 12
```

Explanation:

“ng-controller="myController"” defines a controller .

“app = angular.module()” creates a new module.

“app.controller()” adds the controller's constructor function to the module.

“function($scope) ” defines a constructor function, and also defines an object $scope. When page is loaded, the function will run.

“$scope**.**property = value” assigns the value to the property of $scope object.

In the above example, at the first time when the page is loaded, you see there are two textboxes with different numbers, which are 4 and 8; the result area displays the result 12.

angular. module ( )**:** AngularJS module is a collection of various part of an application, such as controllers, services, filters, directives, *etc.* An AngularJS module defines an application, which will be discussed in later chapter in detail.