DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CCS375 - WEB TECHNOLOGIES

CCS375 - WEB TECHNOLOGIES

COURSE OBJECTIVES:

 To understand different Internet Technologies
 To learn java-specific web services architecture
 To Develop web applications using frameworks

UNIT I WEBSITE BASICS, HTML 5, CSS 3, WEB 2.0 7

Web Essentials: Clients, Servers and Communication – The Internet – World wide web – HTTP Request Message – HTTP Response Message – Web Clients – Web Servers – HTML5 – Tables – Lists – Image – HTML5 control elements – Drag and Drop – Audio – Video controls - CSS3 – Inline, embedded and external style sheets – Rule cascading – Inheritance – Backgrounds – Border Images – Colors – Shadows – Text – Transformations – Transitions – Animations. Bootstrap Framework

UNIT II CLIENT SIDE PROGRAMMING 6

Java Script: An introduction to JavaScript–JavaScript DOM Model-Exception Handling-Validation- Built-in objects-Event Handling- DHTML with JavaScript- JSON introduction – Syntax – Function Files.

UNIT III SERVER SIDE PROGRAMMING 5

Servlets: Java Servlet Architecture- Servlet Life Cycle- Form GET and POST actions- Session Handling- Understanding Cookies- DATABASE CONNECTIVITY: JDBC. 121

UNIT IV PHP and XML 6

An introduction to PHP: PHP- Using PHP- Variables- Program control- Built-in functions- Form Validation. XML: Basic XML- Document Type Definition- XML Schema, XML Parsers and Validation, XSL ,

UNIT V INTRODUCTION TO ANGULAR and WEB APPLICATIONS FRAMEWORKS 6

Introduction to AngularJS, MVC Architecture, Understanding ng attributes, Expressions and data binding, Conditional Directives, Style Directives, Controllers, Filters, Forms, Routers, Modules, Services; Web Applications Frameworks and Tools – Firebase- Docker- Node JS- React- Django- UI & UX.

COURSE OUTCOMES:

CO1: Construct a basic website using HTML and Cascading Style Sheets
CO2: Build dynamic web page with validation using Java Script objects and by applying different event handling mechanisms.
CO3: Develop server side programs using Servlets and JSP.
CO4: Construct simple web pages in PHP and to represent data in XML format.
CO5: Develop interactive web applications.

TEXTBOOKS
1. Deitel and Deitel and Nieto, Internet and World Wide Web - How to Program, Prentice Hall, 5th Edition, 2011.
2. Jeffrey C and Jackson, Web Technologies A Computer Science Perspective, Pearson Education, 2011.
3. Angular 6 for Enterprise-Ready Web Applications, Doguhan Uluca, 1st edition, Packt Publishing
REFERENCES:
1. Stephen Wynkoop and John Burke "Running a Perfect Website", QUE, 2nd Edition,1999.
2. Chris Bates, Web Programming – Building Intranet Applications, 3rd Edition, Wiley Publications, 2009.

CCS375	WEB TECHNOLOGIES
UNIT I	WEBSITE BASICS, HTML 5, CSS 3, WEB 2.0

WEB ESSENTIALS

Web Essentials:
Server:
The software that distributes the information and the machine where the information and softwarereside is called the server.
• provides requested service to client
• e.g., Web server sends requested Web page
Client:
The software that resides on the remote machine, communicates with the server, fetches theinformation, processes it, and then displays it on the remote machine is called the client.
• initiates contact with server (―speaks first‖)
• typically requests service from server
• Web: client implemented in browser
Web server:
Software that delivers Web pages and other documents to browsers using the HTTP protocol
Web Page:
A web page is a document or resource of information that is suitable for the World Wide Web andcan be accessed through a web browser.
Website:
A collection of pages on the World Wide Web that is accessible from the same URL and typicallyresiding on the same server.
URL:
Uniform Resource Locator, the unique address which identifies a resource on the Internet forrouting purposes.
Client-server paradigm:
IThe Client-Server paradigm is the most prevalent model for distributed computing protocols. It is
the basis of all distributed computing paradigms at a higher level of abstraction. It is service-oriented, andemploys a request-response protocol.
A server process, running on a server host, provides access to a service. A client process, running on a client host, accesses the service via the server process.The interaction of the process proceedsaccording to a protocol.
The primary idea of a client/server system is that you have a central repository of information—some kind of data, often in a database—that you want to distribute on

demand to some set ofpeople or machines.

The Internet:

• Medium for communication and interaction in inter connected network.

• Makes information constantly and instantly available to anyone with a connection.

Web Browsers:

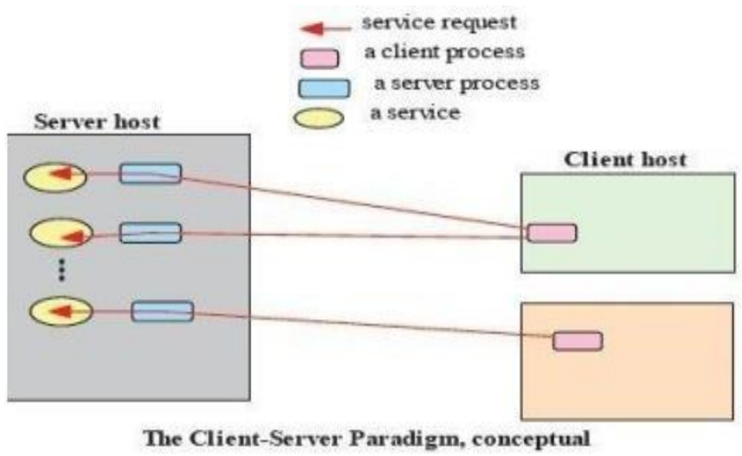
• User agent for Web is called a
browser:o Internet Explorer
o Firefox

Web Server:

• Server for Web is called Web server:
o Apache (public domain)
o MS Internet Information Server

Protocol:

Protocols are agreed formats for transmitting data
between devices.The protocol determines:
i. The error checking required
ii. Data compression method used
iii. The way the end of a message is signalled
iv. The way the device indicates that it has received the message



The Client-Server Paradigm, conceptual

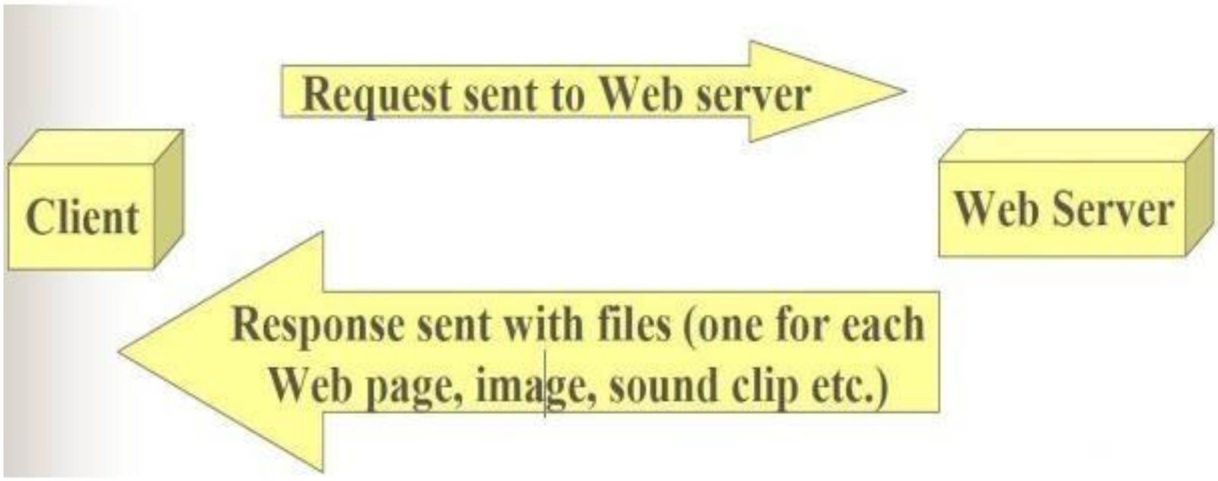
Internet Protocol:

There are many protocols used by the Internet and the WWW:
o TCP/IP
o HTTP
o FTP
o Electronic mail protocols IMAP
o POP

TCP/IP

The Internet uses two main protocols (developed by Vincent Cerf and Robert
Kahn) Transmission control protocol (TCP):Controls disassembly of message into
packets at the originreassembles at the destination
Internet protocol (IP):Specifies the addressing details for each packet Each packet
is labelled withits origin and destination.

Hypertext Transfer Protocol (HTTP)
• The hypertext transfer protocol (HTTP) was developed by Tim Berners-Lee in 1991
• HTTP was designed to transfer pages between machines
• The client (or Web browser) makes a request for a given page and the Server is responsible forfinding it and returning it to the client
• The browser connects and requests a page from the server
• The server reads the page from the file system, sends it to the client andterminates the connection.


Request sent to Web server
Client
Web Server
Response sent with files (one for each Web page, image, sound clip etc.)

Electronic Mail Protocols:
• Electronic mail uses the client/server model
• The organisation has an email server devoted to handling emailo Stores and forwards email messages
• Individuals use email client software to read and send emailo (e.g. Microsoft Outlook, or Netscape Messenger)
• Simple Mail Transfer Protocol (SMTP)
o Specifies format of mail messages
• Post Office Protocol (POP) tells the email server to:
o Send mail to the user's computer and delete it from the server
o Send mail to the user's computer and do not delete it from the server
o Ask whether new mail has arrived.
Interactive Mail Access Protocol (IMAP)
Newer than POP, provides similar functions with additional features.
o e.g. can send specific messages to the client rather than all the messages.A user can view email message headers and the sender's name before downloading the entire message.
Allows users to delete and search mailboxes held on the email server.
The disadvantage of POP: You can only access messages from one PC.
The disadvantage of IMAP : Since email is stored on the email server, there is a need for more and moreexpensive (high speed) storage space.
World Wide Web: comprises software (Web server and browser) and data (Web sites).
Internet Protocol (IP) Addresses:
- Every node has a unique numeric address
- Form: 32-bit binary number

- New standard, IPv6, has 128 bits (1998)
- Organizations are assigned groups of IPs for their computers
- **Domain names**
- Form: host-name. domain-names
- First domain is the smallest (Google)
- Last domain specifies the type of organization (.com)
- Fully qualified domain name - the host name and all of the domain names
- DNS servers - convert fully qualified domain names to IPs

HTTP:

□□Hypertext Transfer Protocol (HTTP) is the communication protocol used by the Internet to transfer hypertext documents.

□□A protocol to transfer hypertext requests and information between servers and browsers

•□Hypertext is text, displayed on a computer, with references (hyperlinks) to other text that the reader can immediately follow, usually by a mouse HTTP is behind every request for aweb document or graph, every click of a hypertext link, and every submission of a form.

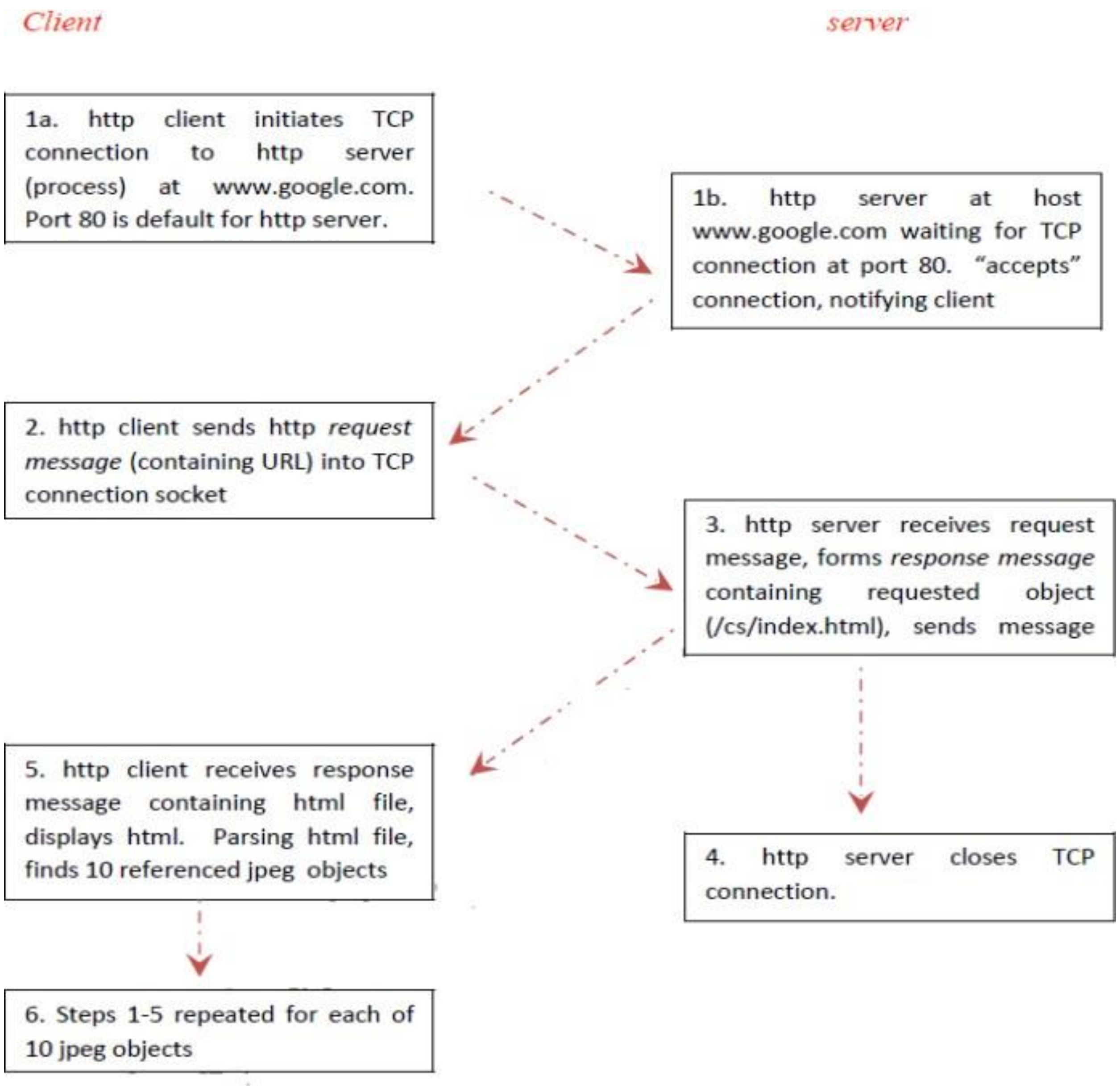
□□HTTP specifies how clients **request** data, and how servers **respond** to these requests.

□□The client makes a request for a given page and the server is responsible for finding it and returning it to the client.

□□The browser connects and requests a page from the server.

•□The server reads the page from the file system and sends it to the client and thenterminates the connection

HTTP Transactions



1a. http client initiates TCP connection to http server (process) at www.google.com. Port 80 is default for http server.

1b. http server at host www.google.com waiting for TCP connection at port 80. "accepts" connection, notifying client

2. http client sends http *request message* (containing URL) into TCP connection socket

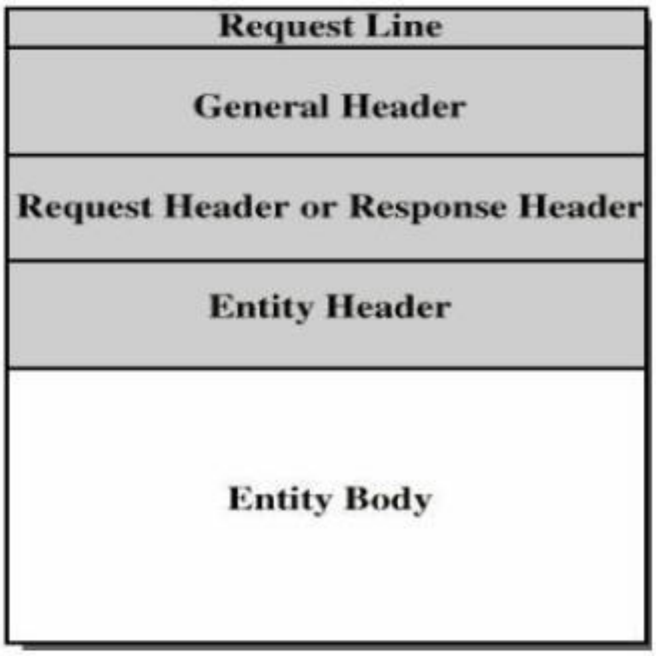
3. http server receives request message, forms *response message* containing requested object (/cs/index.html), sends message

4. http server closes TCP connection.

5. http client receives response message containing html file, displays html. Parsing html file, finds 10 referenced jpeg objects

6. Steps 1-5 repeated for each of 10 jpeg objects

HTTP Message:
HTTP message is the information transaction between the client and server.
Two types of HTTP Message:
1. Requests
a. Client to server
2. Responses
a. Server to client



Fields

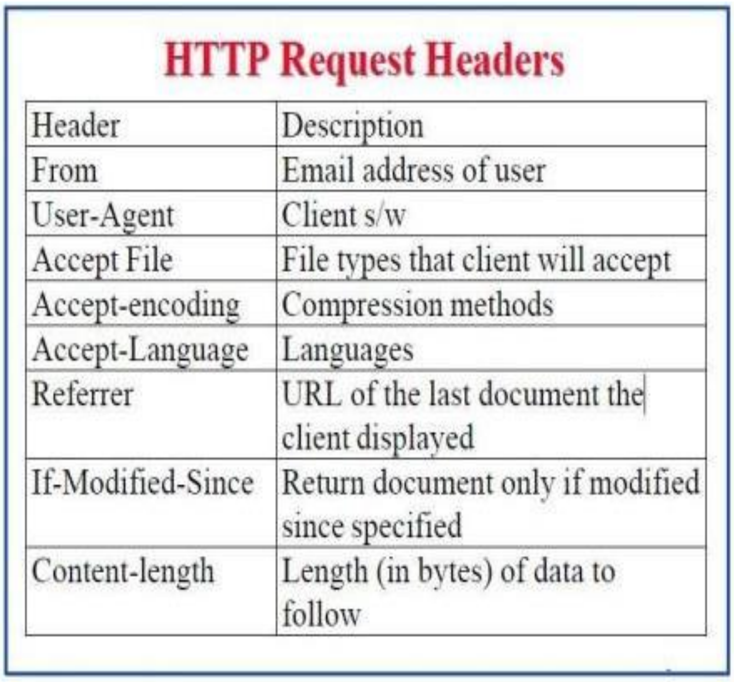
· Request line or Response line
· General header
· Request header or Response header
· Entity header
· Entity body

.10 Request Message:

Request Line:

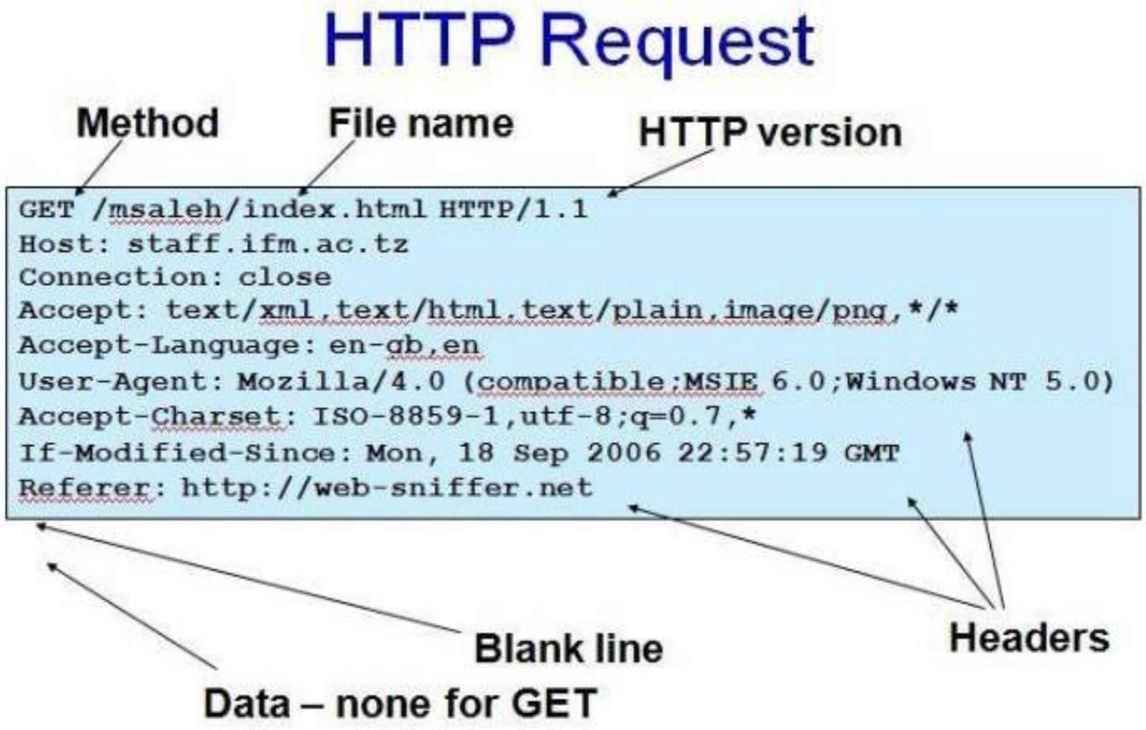
• A request line has three parts, separated by spaceso a *method* name
o the local path of the requested resource
o the version of HTTP being used
• A typical request line is:
o GET /path/to/file/index.html HTTP/1.1
• Notes:
o **GET** is the most common HTTP method; it says "give me this resource". Othermethods include **POST** and **HEAD.** Method names are always uppercase
o The path is the part of the URL after the host name, also called the *request URI*
o The HTTP version always takes the form "**HTTP/x.x**", uppercase.

Request Header:



HTTP Request Headers

Header	Description
From	Email address of user
User-Agent	Client s/w
Accept File	File types that client will accept
Accept-encoding	Compression methods
Accept-Language	Languages
Referrer	URL of the last document the client displayed
If-Modified-Since	Return document only if modified since specified
Content-length	Length (in bytes) of data to follow



```
GET /msaleh/index.html HTTP/1.1
Host: staff.ifm.ac.tz
Connection: close
Accept: text/xml,text/html,text/plain,image/png,*/*
Accept-Language: en-gb,en
User-Agent: Mozilla/4.0 (compatible;MSIE 6.0;Windows NT 5.0)
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*
If-Modified-Since: Mon, 18 Sep 2006 22:57:19 GMT
Referer: http://web-sniffer.net
```

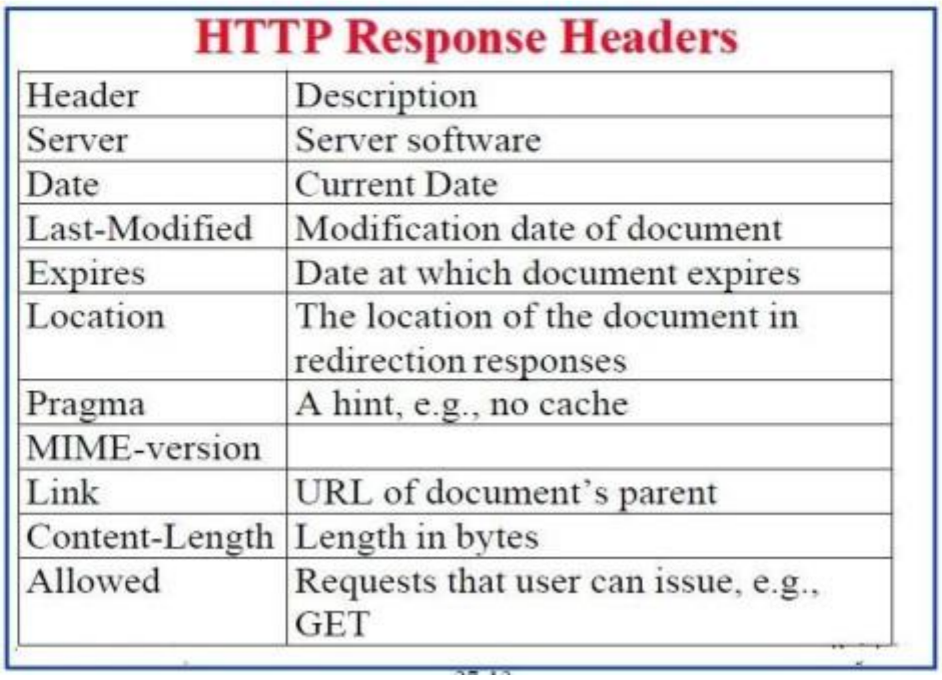
.11 Response Message:

Response Line:

• A request line has three parts, separated by
spaceso the HTTP version,
o a *response status code* that gives the result of the request, and
o an English *reason phrase* describing the status code
• Typical status lines are:
o HTTP/1.0 200 OK or
o HTTP/1.0 404 Not Found
• Notes:
o The HTTP version is in the same format as in the request line, "**HTTP/x.x**".

o The status code is meant to be computer-readable; the reason phrase is meant to behuman-readable, and may vary.

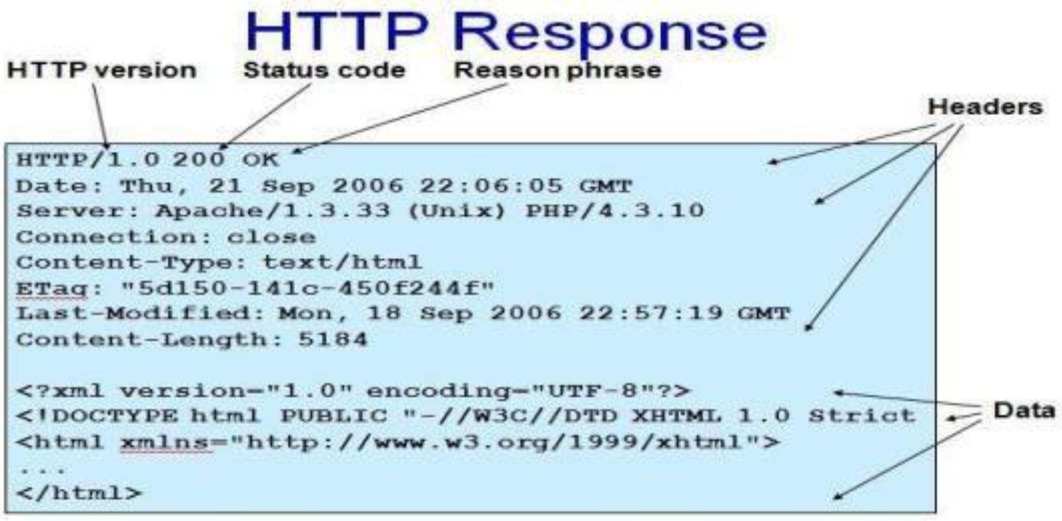
HTTP Request Header:



HTTP Response Headers

Header	Description
Server	Server software
Date	Current Date
Last-Modified	Modification date of document
Expires	Date at which document expires
Location	The location of the document in redirection responses
Pragma	A hint, e.g., no cache
MIME-version	
Link	URL of document's parent
Content-Length	Length in bytes
Allowed	Requests that user can issue, e.g., GET

EXAMPLE



```
HTTP/1.0 200 OK
Date: Thu, 21 Sep 2006 22:06:05 GMT
Server: Apache/1.3.33 (Unix) PHP/4.3.10
Connection: close
Content-Type: text/html
ETag: "5d150-141c-450f244f"
Last-Modified: Mon, 18 Sep 2006 22:57:19 GMT
Content-Length: 5184

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict
<html xmlns="http://www.w3.org/1999/xhtml">
...
</html>
```

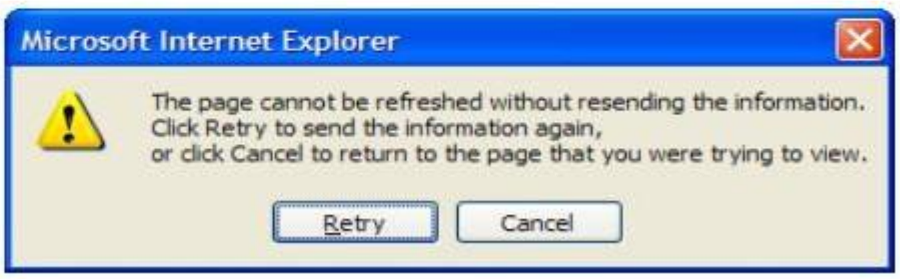
HTTP Method:

• HTTP method is supplied in the request line and specifies the operation that the client hasrequested.

Some common methods:

• Options
• Get
• Head
• Post
• Put
• Move

• Delete
Two methods that are mostly used are the GET and POST:
o **GET** for queries that can be safely repeated
o **POST** for operations that may have side effects (e.g. ordering a book from an on-line store).
The GET Method
• It is used to retrieve information from a specified URI and is assumed to be a safe, repeatableoperation by browsers, caches and other HTTP aware components
• Operations have no side effects and GET requests can be re-issued.
• For example, displaying the balance of a bank account has no effect on the account and can besafely repeated.
• Most browsers will allow a user to refresh a page that resulted from a **GET**, without displayingany kind of warning
• Proxies may automatically retry **GET** requests if they encounter a temporary network connectionproblem.
• GET requests is that they can only supply data in the form of parameters encoded in the URI(known as a **Query String**) – [downside]
• Cannot be unused for uploading files or other operations that require large amounts of data to besent to the server.
The POST Method
• Used for operations that have side effects and cannot be safely repeated.
• For example, transferring money from one bank account to another has side effects and shouldnot be repeated without explicit approval by the user.
• If you try to refresh a page in Internet Explorer that resulted from a **POST**, it displays thefollowing message to warn you that there may


Microsoft Internet Explorer

The page cannot be refreshed without resending the information.
Click Retry to send the information again,
or click Cancel to return to the page that you were trying to view.

The **POST** request message has a content body that is normally used to sendparameters and data
• The IIS server returns two status codes in its response for a **POST** request
o The first is **100 Continue** to indicate that it has successfully received the **POST** request
o The second is **200 OK** after the request has been processed.
HTTP response status codes
• Informational (1xx)
• Successful (2xx)
• Redirection (3xx)
o 301: moved permanently
• Client error (4xx)
o 403 : forbidden
o 404: Not found

• Server error (5xx)
o 503: Service unavailable
o 505: HTTP version not supported

1.12 HTTP

❖ **Features**

• Persistent TCP Connections: Remain open for multiple requests
• Partial Document Transfers: Clients can specify start and stop positions
• Conditional Fetch: Several additional conditions
• Better content negotiation
• More flexible authentication.

Web Browsers :

• Mosaic - NCSA (Univ. of Illinois), in early 1993 First to use a GUI, led to Explosion of Web use Initially for X-Windows, under UNIX, but was ported to other platforms by late 1993
• Browsers are clients - always initiate, servers react (although sometimes servers require responses)
• Most requests are for existing documents, using Hypertext Transfer Protocol
• (HTTP) But some requests are for program execution, with the output being returned as a document.

Browser: A web browser is a software application for retrieving, presenting, and traversing information resources on the World Wide Web.

Web Servers:

- Provide responses to browser requests, either existing documents or dynamically Built documents.
- Browser-server connection is now maintained through more than one request- Response cycle
- All communications between browsers and servers use Hypertext Transfer Protocol
- Web servers run as background processes in the operating system.
- Monitor a communications port on the host, accepting HTTP messages when they appear All current Web servers came from either

1. The original from CERN
2. The second one, from NCSA

- Web servers have two main directories:

1. Document root (servable documents)
2. Server root (server system software)

- Document root is accessed indirectly by clients
- Its actual location is set by the server Configuration file
- Requests are mapped to the actual location
- Virtual document trees
- Virtual hosts
- Proxy servers
- Web servers now support other Internet protocols
- Apache (open source, fast, reliable)
- IIS
- Maintained through a program with a GUI interface.

HTML 5

- ✓ HTML is the main markup language for describing the structure of web pages.
- ✓ HTML stands for **Hypertext Markup Language**.
- ✓ HTML is the basic building block of World Wide Web.
- ✓ Hypertext is text displayed on a computer or other electronic device with references to other text that theuser can immediately access, usually by a mouse click or key press.
- ✓ Apart from text, hypertext may contain **tables, lists, forms, images**, and other presentational elements. It isan easy-to-use and flexible format to share information over the Internet.
- ✓ Markup languages use sets of markup tags to characterize text elements within a document, which givesinstructions to the web browsers on how the document should appear.
- ✓ HTML was originally developed by **Tim Berners-Lee** in 1990.
- ✓ He is also known as the father of the web. In1996, the World Wide Web Consortium (W3C) became the authority to maintain the HTML specifications. HTML also became an international standard (ISO) in 2000. HTML5 is the latest version of HTML.
- ✓ HTML5 provides a faster and more robust approach to web development.

HTML Tags and Elements

HTML is written in the form of HTML elements consisting of markup tags.
These markup tags arethe fundamental characteristic of HTML.
Every markup tag is composed of a keyword, surroundedby angle brackets, such as <html>, <head>, <body>, <title>, <p>, and so on.
HTML tags normally come in pairs like <html> and </html>.
The first tag in a pair is often called the **opening tag (or start tag),** and the second tag is called the **closing tag (or end tag).**
An opening tag and a closing tag are identical, except a slash (/) after the opening angle bracket ofthe closing tag, to tell the browser that the command has been completed.

HTML5 IMAGE:

Inserting Images into Web Pages

Images enhance visual appearance of the web pages by making them more interesting and colorful.
The <img> tag is used to insert images in the HTML documents. It is an empty element and containsattributes only. The syntax of the <img> tag can be given with:

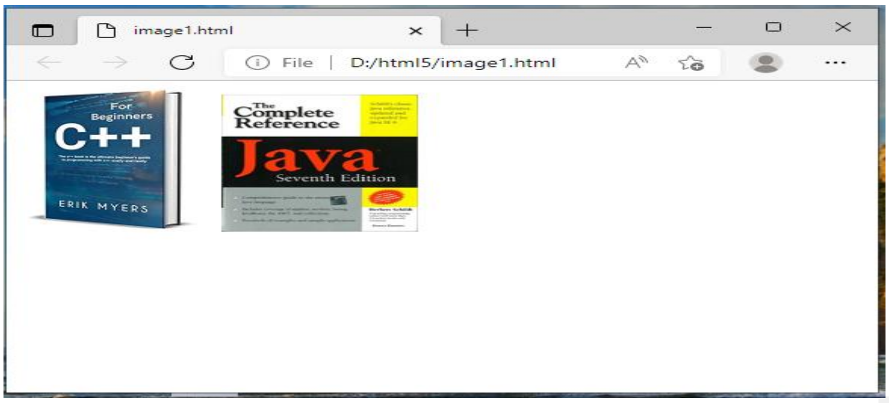
```
<img src="url" alt="some_text">
```

Attributes	**Description**
src	Specifes the path to the image
alt	Specifies an alternate text
height	Specifies the height of an image
width	Specifies the width
ismap	Specifies an image as a server-side image map
usemap	Define a valid map name

The following example inserts three images on the web page:
Example:

```
<html><head></head>
<body>
<img src="c.jpg" height=="160"  width="130"alt="C++ how to Program">
<img src="java.jpg" height="150" width="130"alt="Java how to program">
</body></html>
```

OUTPUT:



Each image must carry at least two attributes: **the src attribute, and an alt attribute.** The src attribute tells the browser where to find the image. Its value is the URL of the image file. Whereas, the alt attribute provides an alternative text for the image, if it is unavailable or cannot bedisplayed for some reason. Its value should be a meaningful substitute for the image.

- **Using Image as Hyperlink:**

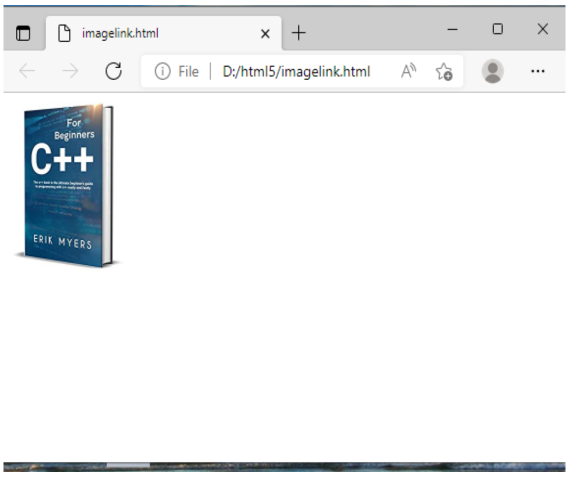
By using image as hyperlinks, web developers can create graphical web pages that link to other resources.
Tag <a> anchor-We can use hyperlinks by using attributes.
Href → specifies the URL.By default links will appear as follows in all browser.

```
<html> <head > </head>
<body>
<a href ="link.html">
<img src="c.jpg" height=="160"  width="130"alt="C++ how to Program">
</a>  </body> </html>
```

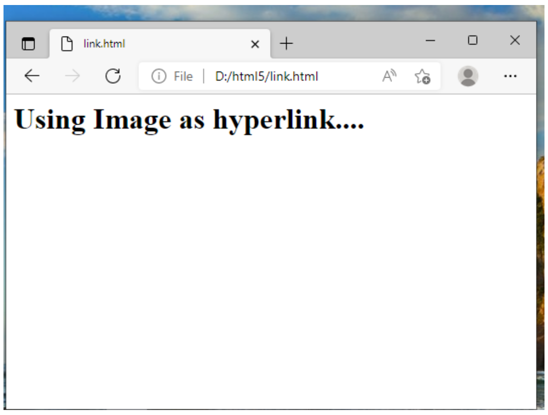
Link.html

```
<html>   <head>  </head>
<body> <h1> Using Image as hyperlink....</h1>
</body>  </html>
```

Output:


imagelink.html

File | D:/html5/imagelink.html


link.html

File | D:/html5/link.html

Using Image as hyperlink....

Hyperlink

Link act as a pointer to some web page or documents and image. Both text and image can be acts as hyperlinks.

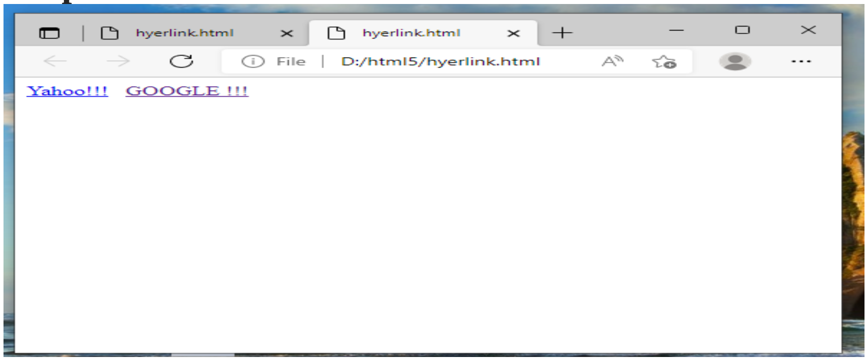
Links are created using the (a) anchor element. Attribute used is href →specifies the URL.

- An unvisited link is blue.
- A visited link is purple.
- An active link is red.

Example:

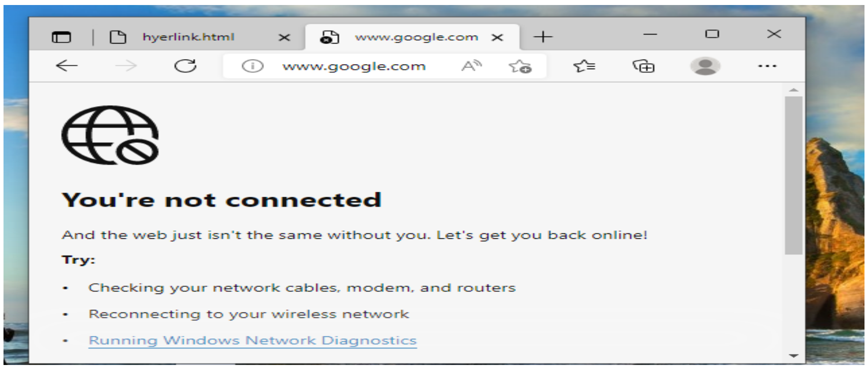
```
<html>
<head></head>
<body>
<a href ="http://www.yahoo.com">Yahoo!!!</a>&nbsp&nbsp
<a href="http://www.google.com">GOOGLE !!!</a>
</body></html>
```

Output:


hyerlink.html | hyerlink.html

File | D:/html5/hyerlink.html

Yahoo!!! GOOGLE !!!


hyerlink.html | www.google.com

www.google.com

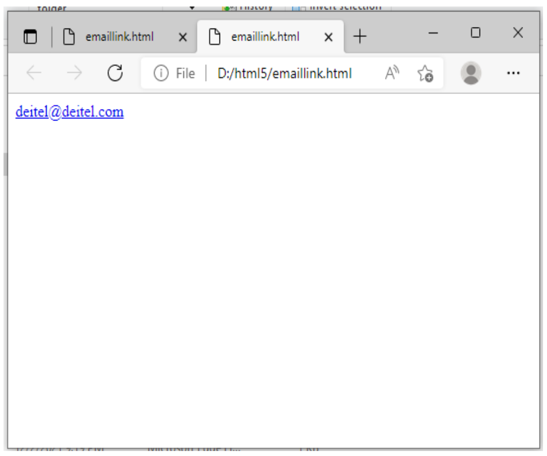
You're not connected

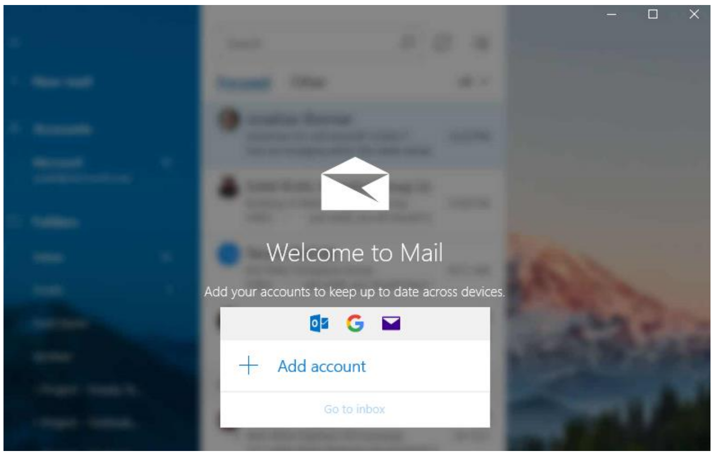
And the web just isn't the same without you. Let's get you back online!

Try:

- Checking your network cables, modem, and routers
- Reconnecting to your wireless network
- Running Windows Network Diagnostics

- Hyperlink to an E-Mail address:Anchor can link to e-mail address using a **mailto:url**
  Format : <a href ="mailto:deitel@deitel.com">deitel@deitel.com</a>


emaillink.html
D:/html5/emaillink.html
deitel@deitel.com



HTML Lists

HTML lists are used to present list of information in well formed and semantic way. There are threedifferent types of list in HTML and each one has a specific purpose and meaning.

- **Unordered list** — Used to create a list of related items, in no particular order.
- **Ordered list** — Used to create a list of related items, in a specific order.
- **Description list** — Used to create a list of terms and their descriptions.

✓ HTML Unordered Lists

An unordered list created using the <ul> element, and each list item starts with the <li> element.The list items in unordered lists are marked with bullets.

<ul>… </ul>	Specifies an unordered list
<li>…</li>	Specifies list item
<ul type=”circle” >	Display the circular bullets
<ul type=”disc”>	Display the solid round bullets
<ul type=”square”>	Display the squared bullets

Here's an example:

```
<ul type=”disc”>
   <li >Chocolate Cake</li>
   <li>Black Forest Cake</li>
   <li>Pineapple Cake</li>
</ul>
```

— The output of the above example will look something like this:

- Chocolate Cake
- Black Forest Cake
- Pineapple Cake

✓ HTML Ordered Lists

An ordered list created using the <ol> element, and each list item starts with the <li> element.Ordered lists are used when the order of the list's items is important.
The list items in an ordered list are marked with numbers.

<ol>… </ol>	Specifies an ordered list
<li>…</li>	Specifies list item
<ul type="A" >	Display the list in the following A,B...
<ul type="I">	Display the list in the following I,II…
<ul type="i">	Display the list in the following i, ii..
<ul type="1">	Display the list in the following 1,2…

Example:

```
<ol type="1">
  <li>Fasten your seatbelt</li>
  <li>Starts the car's engine</li>
  <li>Look around and go</li>
</ol>
```

— The output of the above example will look something like this:

1. Fasten your seatbelt
2. Starts the car's engine
3. Look around and go

✓ HTML Definition list:

Specifically used for lists in which each element is labeled with a word rather than a bullet or number.

Tag	Description
<dl>..</dl>	Specifies a description list
<dt>…</dt>	Specifies the term in a description list
<dd>..</dd>	Specifies description of term a description list

Ex:

```
 <h1> Abbrevation</h1>
<dl>
 <dt>HTML </dt>
  <dd> Hypertext Markup Language….</dd>
 <dt>CSS </dt>
   <dd> Cascading Style sheet</dd>
</dl>
```

Output:

Abbreviation

HTML

Hypertext Markup language

CSS

Cascading Style sheet.

<u>HTML Tables</u>

Creating Tables in HTML

HTML table allows you to arrange data into rows and columns. They are commonly used to displaytabular data like product listings, customer's details, financial reports, and so on. You can create a table using the <table> element. Inside the <table> element, you can use the <tr> elements to create rows, and to create columns inside a row you can use the <td> elements.You can also define a cell as a header for a group of table cells using the <th> element.

The following example demonstrates the most basic structure of a table.

Example

```
<table>
   <tr>
      <th>No.</th>
      <th>Name</th>
      <th>Age</th>
    </tr>
    <tr>
      <td>1</td>
      <td>Peter Parker</td>
      <td>16</td>
    </tr>
    <tr>
      <td>2</td>
      <td>Clark Kent</td>
      <td>34</td>
   </tr>
</table>
```

Tables do not have any borders by default. You can use the CSS border property to add borders tothe tables. Also, table cells are sized just large enough to fit the contents by default. To add more space around the content in the table cells you can use the CSS padding property.

Rowspan and colspan:

Used to categorize the information properly in sub rows and sub columns.

Rowspan → Used to extend the row vertically.

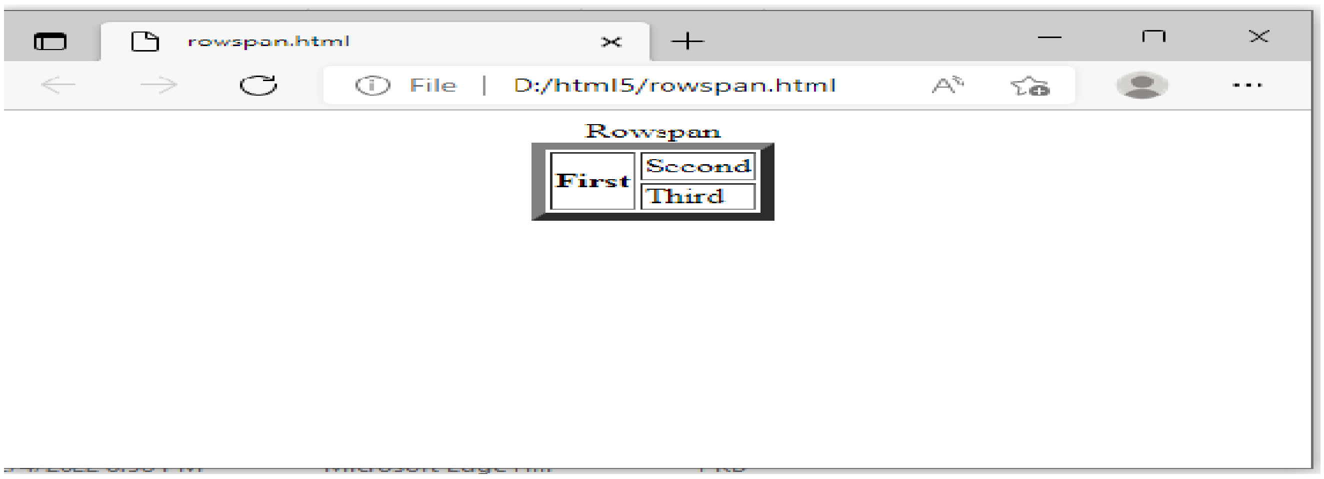
Colspan →Used to extend the column horizontally.

When rowspan =2 then the row can be extended vertically by 2 cells.

Example:

```
<html>
<head>
</head>
<body><center>
<table border="6">
 Rowspan
 <tr>
   <th rowspan="2">First</th>
    <td>Second</td> </tr>
    <tr> <td>Third</td> </tr>
  </table>
</center>
</body>
</html>
```

OUTPUT:



rowspan.html

File | D:/html5/rowspan.html

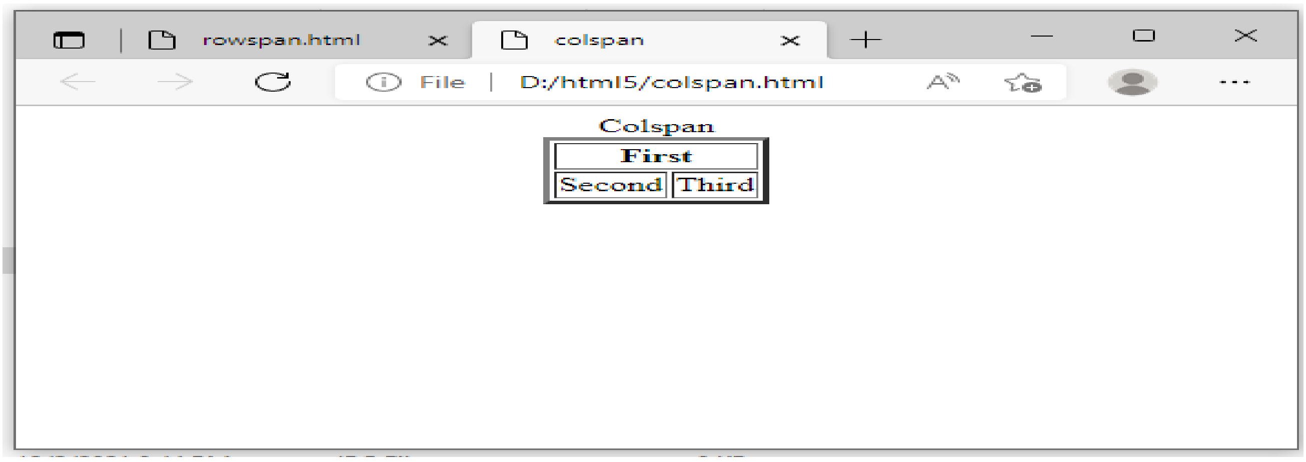
```
Rowspan
First | Second
      | Third
```

When colspan =2 then the row can be extended horizontally by 2 cells.

Example:

```
<html>
 <head>
  <title> colspan </title>
  </head>
 <body>
  <center>
   <table border="3">
  Colspan
    <tr>
     <th colspan="2">First</th>
    </tr>
    <tr>
     <td>Second</td>
     <td>Third</td>
  </tr>
 </table>
 </center>
 </body>
</html>
```

OUTPUT:



Colspan

First	
Second	**Third**

Defining a Table Header, Body, and Footer

HTML provides a series of tags <thead>, <tbody>, and <tfoot> that helps you to create morestructured table, by defining header, body and footer regions, respectively.

The following example demonstrates the use of these elements.

Example

```
<table>
   <thead>
      <tr>
         <th>Items</th>
         <th>Expenditure</th>
      </tr>
   </thead
        >
  <tbody>
     <tr>
         <td>Stationary</td>
         <td>2,000</td>
      </tr>
      <tr>
         <td>Furniture</td>
         <td>10,000</td>
     </tr>
   </tbody
        >
  <tfoot>
      <tr>
          <th>Total</th>
          <td>12,000</td>
      </tr>
  </tfoot>
</table>
```

HTML5 Image

HTML Images Syntax

In HTML, images are defined with the `<img>` tag.

The `<img>` tag is empty, it contains attributes only, and does not have a closing tag.

The `src` attribute specifies the URL (web address) of the image:

```
<img src="url">
```

EXAMPLE

```
<!DOCTYPE html>
<html>
<body>
<h2>HTML Image</h2>
<img src="html5.gif" alt="HTML5 Icon" style="width:128px;height:128px;">
</body>
</html>
```

OUTPUT

HTML Image

HTML Form

HTML Forms are required to collect different kinds of user inputs, such as contact details likename, email address, phone numbers, or details like credit card information, etc.

Forms contain special elements called **controls** like ***inputbox, checkboxes, radio-buttons, submit buttons***, etc. Users generally complete a form by modifying its controls e.g. entering text, selectingitems, etc. and submitting this form to a web server for further processing.

The <form> tag is used to create an HTML form. Here's a simple example of a login form:

Example

```
<form>
   <label>Username: <input type="text"></label>
   <label>Password: <input type="password"></label>
   <input type="submit" value="Submit">
</form>
```

The following section describes different types of controls that you can use in your form.

Input Element
This is the most commonly used element within HTML forms.
It allows you to specify various types of user input fields, depending on the type attribute. An inputelement can be of type *text field*, *password field*, *checkbox*, *radio button*, *submit button*, *reset button*, *file select box*, as well as several new input types introduced in HTML5.
The most frequently used input types are described below.

Text Fields
Text fields are one line areas that allow the user to input text.
Single-line text input controls are created using an <input> element, whose type attribute has a valueof text. Here's an example of a single-line text input used to take username:

Example

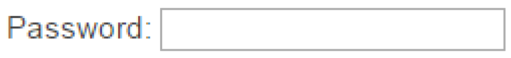
```
<form>
   <label for="username">Username:</label>
   <input type="text" name="username" id="username">
</form>
```

— The output of the above example will look something like this:

Password Field
Password fields are similar to text fields. The only difference is; characters in a password field aremasked, i.e. they are shown as asterisks or dots. This is to prevent someone else from reading the password on the screen. This is also a single-line text input controls created using
an <input> element whose type attribute has a value of password.

Example

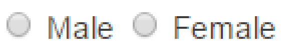
```
<form>
   <label for="user-pwd">Password:</label>
   <input type="password" name="user-password" id="user-pwd">
</form>
```
— The output of the above example will look something like this:



Radio Buttons
Radio buttons are used to let the user select exactly one option from a pre-defined set of options. Itis created using an <input> element whose type attribute has a value of radio.

Example
Try this code »

```
<form>
   <input type="radio" name="gender" id="male">
   <label for="male">Male</label>
   <input type="radio" name="gender" id="female">
   <label for="female">Female</label>
</form>
```
— The output of the above example will look something like this:



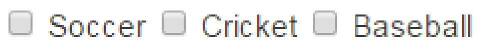
Checkboxes
Checkboxes allows the user to select one or more option from a pre-defined set of options. It iscreated using an <input> element whose type attribute has a value of checkbox.

Example

```
<form>
   <input type="checkbox" name="sports" id="soccer">
   <label for="soccer">Soccer</label>
   <input type="checkbox" name="sports" id="cricket">
```

```
    <label for="cricket">Cricket</label>
    <input type="checkbox" name="sports" id="baseball">
    <label for="baseball">Baseball</label>
</form>
```

— The output of the above example will look something like this:



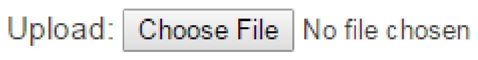
File Select box

The file fields allow a user to browse for a local file and send it as an attachment with the form data.Web browsers such as Google Chrome and Firefox render a file select input field with a Browse button that enables the user to navigate the local hard drive and select a file. This is also created using an `<input>` element, whose `type` attribute value is set to `file`.

Example

```
<form>
    <label for="file-select">Upload:</label>
    <input type="file" name="upload" id="file-select">
</form>
```

— The output of the above example will look something like this:



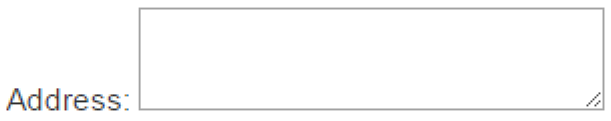
Textarea

Textarea is a multiple-line text input control that allows a user to enter more than one line of text.Multi-line text input controls are created using an `<textarea>` element.

Example

```
<form>
    <label for="address">Address:</label>
    <textarea rows="3" cols="30" name="address" id="address"></textarea>
</form>
```

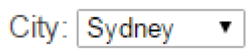
— The output of the above example will look something like this:



Select Boxes
A select box is a dropdown list of options that allows user to select one or more option from a pull-down list of options. Select box is created using the <select> element and <option> element.The <option> elements within the <select> element define each list item.

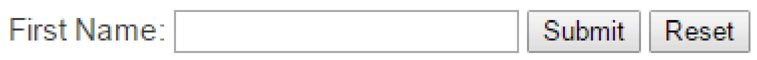
Example

```
<form>
   <label for="city">City:</label>
   <select name="city" id="city">
      <option value="sydney">Sydney</option>
      <option value="melbourne">Melbourne</option>
      <option value="cromwell">Cromwell</option>
   </select>
</form>
```
— The output of the above example will look something like this:



Submit and Reset Buttons
A submit button is used to send the form data to a web server. When submit button is clicked theform data is sent to the file specified in the form's action attribute to process the submitted data.
A reset button resets all the forms control to default values. Try out the following example by typingyour name in the text field, and click on submit button to see it in action.

Example

```
<form action="action.php" method="post">
   <label for="first-name">First Name:</label>
   <input type="text" name="first-name" id="first-name">
   <input type="submit" value="Submit">
   <input type="reset" value="Reset">
```


```
</form>
```

<u>HTML5 Colors</u>

```
<!DOCTYPE html>   <html>
<body>
<h1 style="background-color:Tomato;">Tomato</h1>
<h1 style="background-color:Orange;">Orange</h1>
<h1 style="background-color:DodgerBlue;">DodgerBlue</h1>
<h1 style="background-color:MediumSeaGreen;">MediumSeaGreen</h1>

<h1 style="background-color:Gray;">Gray</h1>
<h1 style="background-color:SlateBlue;">SlateBlue</h1>
<h1 style="background-color:Violet;">Violet</h1>
<h1 style="background-color:LightGray;">LightGray</h1>
</body>
</html>
```

OUTPUT

```
Tomato
Orange

DodgerBlue

MediumSeaGreen

Gray

SlateBlue

Violet


LightGray
```

<u>HTML5 Audio</u>

Embedding Audio in HTML Document

Inserting audio onto a web page was not easy before, because web browsers did not have a uniformstandard for defining embedded media files like audio.

Using the HTML5 audio Element

The newly introduced HTML5 `<audio>` element provides a standard way to embed audio in webpages. However, the audio element is relatively new but it works in most of the modern web browsers.

The following example simply inserts an audio into the HTML5 document, using the browserdefault set of controls, with one source defined by the `src` attribute.

Example

```
<audio controls="controls" src="media/birds.mp3">
  Your browser does not support the HTML5 Audio element.
</audio>
```

An audio, using the browser default set of controls, with alternative sources.

Example

```
<audio controls="controls">
  <source src="media/birds.mp3" type="audio/mpeg">
  <source src="media/birds.ogg" type="audio/ogg">
  Your browser does not support the HTML5 Audio element.
</audio>
```

HTML5 Video

Embedding Video in HTML Document

Inserting video onto a web page was not relatively easy, because web browsers did not have auniform standard for defining embedded media files like video.

Using the HTML5 video Element

The newly introduced HTML5 `<video>` element provides a standard way to embed video in webpages. However, the video element is relatively new, but it works in most of the modern web browsers.
The following example simply inserts a video into the HTML document, using the browser defaultset of controls, with one source defined by the `src` attribute.

Example

```
<video controls="controls" src="media/shuttle.mp4">
  Your browser does not support the HTML5 Video element.
</video>
```

A video, using the browser default set of controls, with alternative sources.

Example

```
<video controls="controls">
  <source src="media/shuttle.mp4" type="video/mp4">
  <source src="media/shuttle.ogv" type="video/ogg">
```

Your browser does not support the HTML5 Video element.
</video>

New HTML5 Elements

The most interesting new HTML5 elements are:

New **semantic elements** like <header>, <footer>, <article>, and <section>. New **attributes of form elements** like number, date, time, calendar, and range.
New **graphic elements**: <svg> and <canvas>.

New **multimedia elements**: <audio> and <video>.

What are Semantic Elements?

A semantic element clearly describes its meaning to both the browser and the developer.

Examples of **non-semantic** elements: <div> and <span> - Tells nothing about its content.

Examples of **semantic** elements: <form>, <table>, and <article> - Clearly defines its content.

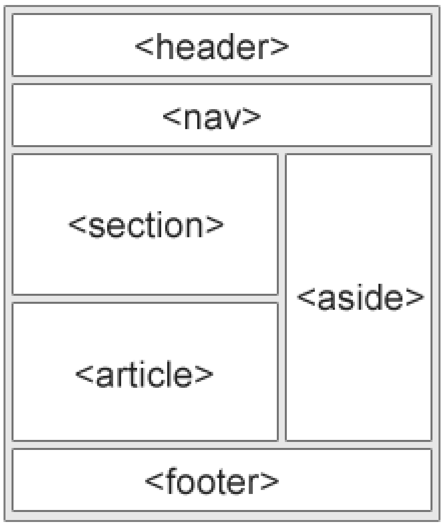
New Semantic Elements in HTML5

Many web sites contain HTML code like:

<div id="nav"> <div class="header"> <div id="footer">to indicate navigation, header, and footer.

HTML5 offers new semantic elements to define different parts of a web page:

- <article>
- <aside>
- <details>
- <figcaption>
- <figure>
- <footer>
- <header>
- <main>
- <mark>
- <nav>
- <section>
- <summary>
- <time>



HTML5 <section> Element

The <section> element defines a section in a document.

According to W3C's HTML5 documentation: "A section is a thematic grouping of content, typically with aheading."

A home page could normally be split into sections for introduction, content, and contact information.

Example

```
<section>
 <h1>WWF</h1>
 <p>The World Wide Fund for Nature (WWF) is       </p>
</section>
```

HTML5 <article> Element

The <article> element specifies independent, self-contained content.

An article should make sense on its own, and it should be possible to read it independently from the rest of theweb site.

Examples of where an <article> element can be used:

- Forum post
- Blog post
- Newspaper article

Example

```
<article>
 <h1>What Does WWF Do?</h1>
 <p>WWF's mission is to stop the degradation of our planet's
 natural environment,and build a future in which humans live in
```

```
  harmony with nature.</p>
</article>
```

HTML5 <header> Element

The <header> element specifies a header for a document or section.

The <header> element should be used as a container for introductory content.

You can have several <header> elements in
one document. The following example defines
a header for an article:

Example

```
<article>
  <header>
    <h1>What Does WWF Do?</h1>
    <p>WWF's mission:</p>
  </header>
  <p>WWF's mission is to stop the degradation of our planet's
  natural environment, and build a future in which humans live in
  harmony with nature.</p>
</article>
```

HTML5 <footer> Element

The <footer> element specifies a footer for a document or section.

A <footer> element should contain information about its containing element.

A footer typically contains the author of the document, copyright information, links to terms of use, contactinformation, etc.

You may have several <footer> elements in one document.

Example

```
<footer>
  <p>Posted by: Hege Refsnes</p>
  <p>Contact information: <a
  href="mailto:someone@example.com">
  someone@example.com</a>.</p>
</footer>
```

HTML5 <figure> and <figcaption> Elements

The purpose of a figure caption is to add a visual explanation to an image.

In HTML5, an image and a caption can be grouped together in a <figure> element:

Example

```
<figure>
  <img src="pic_trulli.jpg" alt="Trulli">
  <figcaption>Fig1. - Trulli, Puglia, Italy.</figcaption>
</figure>
```

OUTPUT

Places to Visit

Puglia's most famous sight is the unique conical houses (Trulli) found in the area aroundAlberobello, a declared UNESCO World Heritage Site.

Fig.1 - Trulli, Puglia, Italy.

Semantic Elements in HTML5

Below is an alphabetical list of the new semantic elements in HTML5. The links go to our complete HTML5 Reference.

Tag	Description
<article>	Defines an article
<aside>	Defines content aside from the page content

<details>	Defines additional details that the user can view or hide
<figcaption>	Defines a caption for a <figure> element
<figure>	Specifies self-contained content, like illustrations,diagrams, photos, code listings, etc.
<footer>	Defines a footer for a document or section
<header>	Specifies a header for a document or section
<main>	Specifies the main content of a document
<mark>	Defines marked/highlighted text
<nav>	Defines navigation links
<section>	Defines a section in a document
<summary>	Defines a visible heading for a <details> element
<time>	Defines a date/time

HTML5 Drag and Drop

Drag the W3Schools image into the rectangle.

Drag and Drop

Drag and drop is a very common feature. It is when you "grab" an object and drag it to a different location. In HTML5, drag and drop is part of the standard: Any element can be draggable.

HTML Drag and Drop Example

The example below is a simple drag and drop example:

Example

```
<!DOCTYPE HTML>
<html>
<head>
<script>
function
 allowDrop(ev) {
 ev.preventDefault()
 ;
}

function drag(ev) {
 ev.dataTransfer.setData("text",
 ev.target.id);
}

function drop(ev)
 {
 ev.preventDefau
 lt();
 var data = ev.dataTransfer.getData("text");
 ev.target.appendChild(document.getElementById(data));
}
</script>
</head>
<body>

<div id="div1" ondrop="drop(event)" ondragover="allowDrop(event)"></div>

<img id="drag1" src="img_logo.gif" draggable="true" ondragstart="drag(event)" width="336"
height="69">

</body>
</html>
```

OUTPUT

Drag the W3Schools image into the rectangle:

HTML5 <nav> Element

The <nav> element defines a set of navigation links.

Notice that NOT all links of a document should be inside a <nav> element. The <nav> element is intended only for major block of navigation links.

Example

```
<nav>
  <a href="/html/">HTML</a> |
  <a href="/css/">CSS</a> |
  <a href="/js/">JavaScript</a> |
  <a href="/jquery/">jQuery</a>
</nav>
```

What Is CSS3 And Why Is It Used?

To help build highly interactive online pages, CSS3 is invariably used due to its importance in providing greater options in the design process. When marketing products and services, web design plays a vital part; a site should be created in a manner that will draw potential customers to explore and revisit a website more often. Many **web design firm**s are developing and enhancing websites through the use of CSS3 as this is a great form of web development. This article will help define CSS3 and will point out its advantages.

Definition

The acronym CSS stands for Cascading Style Sheets which is used to augment the functionality, versatility.and efficient performance of site content. It allows for the creation of content-rich websites that do not require much weight or codes; this translates into more interactive graphics and animation, superior user- interface, and significantly more organization and rapid download time.

It is used with HTML to create content structure, with CSS3 being used to format structured content. It is responsible for font properties, colors, text alignments, graphics, background images, tables and other components. This tool provides extra capabilities such as absolute, fixed and relative positioning of various elements. The increasing popularity of CSS3 when used by **web design firm**s stimulates major browsers suchas Google Chrome, Firefox, Safari, and IE9 to adopt and embrace this programming language.

Advantages

Although CSS3 is not the only web development solution, it does allow provide greater advantages for severalreasons.

- **Customization** – A web page can be customized and alterations created in the design by simplychanging a modular file.

- **Bandwidth Requirements** – It decreases server bandwidth requirements, giving rapid download timewhen a site is accessed with desktop or hand-held devices, providing an improved user experience.
- **Consistency** – It delivers consistent and accurate positioning of navigational elements on the website.
- **Appealing** – It makes the site more appealing with adding videos and graphics easier.
- **Viewing** – It allows online videos to be viewed without the use of third-party plug-ins.
- **Visibility** – It delivers the opportunity to improve brand visibility by designing effective online pages.
- **Cost Effective** – It is cost-effective, time-saving, and supported by most browsers.

Since the introduction of CSS3, there is greater control of the presentation of content and various elements on a website; however it is not really responsible for overall design as it only specifies the structure and contentpresentation of certain web pages.

External, internal, and inline CSS styles

Cascading Style Sheets (CSS) are files with styling rules that govern how your website is presented onscreen. CSS rules can be applied to your website's HTML files in various ways. You can use an **external stylesheet**, an **internal stylesheet**, or an **inline style**. Each method has advantages that suitparticular uses.

An **external stylesheet** is a standalone .css file that is linked from a web page. The advantage of external stylesheets is that it can be created once and the rules applied to multiple web pages. Should you need to make widespread changes to your site design, you can make a single change in the stylesheet and it will beapplied to all linked pages, saving time and effort.

An **internal stylesheet** holds CSS rules for the page in the **head** section of the HTML file. The rules only apply to that page, but you can configure CSS classes and IDs that can be used to style multiple elements inthe page code. Again, a single change to the CSS rule will apply to all tagged elements on the page.

Inline styles relate to a specific HTML tag, using a **style** attribute with a CSS rule to style a specific pageelement. They're useful for quick, permanent changes, but are less flexible than external and internal stylesheets as each inline style you create must be separately edited should you decide to make a design change.

Using external CSS stylesheets

An HTML page styled by an external CSS stylesheet must reference the .css file in the document head. Oncecreated, the CSS file must be uploaded to your server and linked in the HTML file with code such as:

```
<link href="style.css" rel="stylesheet" type="text/css">
```

You can name your stylesheet whatever you wish, but it should have a .css file extension.

Using internal CSS stylesheets

Rather than linking an external .css file, HTML files using an internal stylesheet include a set of rules intheir **head** section. CSS rules are wrapped in <style> tags, like this:

```
<head>
<style type="text/css">

   h1 {
         color:#fff
         margin-left: 20px;
      }

   p{
      font-family: Arial, Helvetica, Sans Serif;
      }
</style>
</head>
```

Using inline styles

Inline styles are applied directly to an element in your HTML code. They use the **style** attribute, followed byregular CSS properties.

For example:

```
<h1 style="color:red;margin-left:20px;">Today’s Update</h1>
```

Rule Cascading

Cascade and inheritance

Conflicting rules

- CSS stands for **Cascading Style Sheets**, and that first word *cascading* is incredibly important to understand

— the way that the cascade behaves is key to understanding CSS.

- At some point, we will find that the CSS have created two rules which could potentially apply to the same element.
- The **cascade**, and the closely-related concept of **specificity**, is mechanisms that control which ruleapplies when there is such a conflict.
- Which rule is styling your element may not be the one you expect, so you need to understand how these mechanisms work.
- Also significant here is the concept of **inheritance**, which means that some CSS properties by default inheritvalues set on the current element's parent element, and some don't. This can also cause some behavior that you might not expect.

The cascade

Stylesheets **cascade** — at a very simple level this means that the order of CSS rules matter; when two rulesapply that have equal specificity the one that comes last in the CSS is the one that will be used.

EXAMPLE

In the below example, we have two rules that could apply to the h1. The h1 ends up being colored blue — these rules have an identical selector and therefore carry the same specificity, so the last one in the sourceorder wins.

```
h1 {
   color: red;
   }
h1 {
   color: blue;
}
   <h1>This is my heading.</h1>
```

OUTPUT

This is my heading.

Specificity

Specificity is how the browser decides which rule applies if multiple rules have different selectors, but couldstill apply to the same element. It is basically a measure of how specific a selector's selection will be:

- An element selector is less specific — it will select all elements of that type that appear on a page — so willget a lower score.
- A class selector is more specific — it will select only the elements on a page that have a specific class attributevalue — so will get a higher score.

Example time! Below we again have two rules that could apply to the h1. The below h1 ends up being coloredred — the class selector gives its rule a higher specificity, and so it will be applied even though the rule withthe element selector appears further down in the source order.

EXAMPLE

```
main-heading {

   color: red;

}

      h1 {

   color: blue;

}
```

```
.   <h1 class="main-heading">This is my heading.</h1>
```

OUTPUT

```
This is my heading.
```

Inheritance

Inheritance also needs to be understood in this context — some CSS property values set on parent elementsare inherited by their child elements, and some aren't.

For example, if you set a color and font-family on an element, every element inside it will also be styled withthat color and font, unless you've applied different color and font values directly to them.

Some properties do not inherit — for example if you set a width of 50% on an element, all of its descendantsdo not get a width of 50% of their parent's width. If this was the case, CSS would be very frustrating to use!

```
body {
   color: blue;
}

span {
   color: black;
}
```

```
<p>As the body has been set to have a color of blue this is inherited through the
descendants.</p>

<p>We can change the color by targetting the element with a selector,
 such as this

<span>span</span>.</p>
```

OUTPUT

As the body has been set to have a color of blue this is inherited through the descendants.

We can change the color by targetting the element with a selector, such as this **span**.

BACKGROUND:

CSS provides control over the backgrounds of block-level elements. CSS can set a back- ground color or add background images to HTML5 elements.

- ***background-image Property***

The **background-image** property specifies the image URL for the image flower.png in the format url(*fileLocation*).

- ***background-position Property***

The **background-position** property places the image on the page. The keywords top, bottom, center, left and right are used individually or in combination for vertical and horizontal positioning. You can position an image using lengths by specifying the hor-izontal length followed by the vertical length.

For example, to position the image as *hori- zontally centered* (positioned at 50 percent of the distance across the screen) and 30 pixels from the top, use

background-position: 50% 30px;

- ***background-repeat Property***

✓ The **background-repeat** proper controls background image **tiling**, which place-es *multiple copies* of the image next to each other to fill the background. Here, we set the tiling to no-repeat to display only one copy of the background image.

✓ Other values in- clude repeat (the default) to tile the image *vertically and horizontally*,

✓ repeat-x to tile the image only *horizontally* or

✓ repeat-y to tile the image only *vertically*.

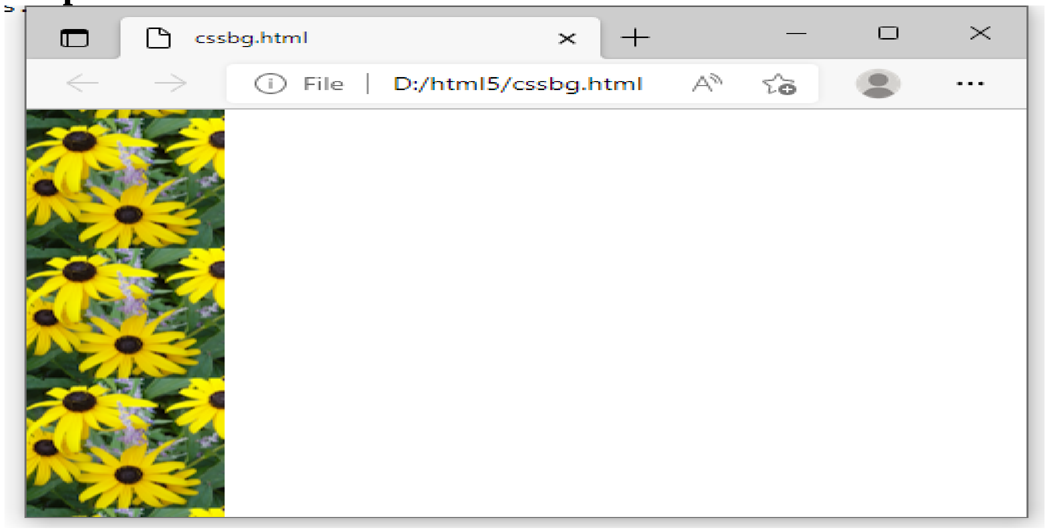
background-attachment: fixed Property

The next property setting, **background-attachment: fixed** fixes the image in the position specified by **background-position**.

Example:

```
<html>
  <head>
    <style type="text/css">
     body
      {
        background-image:url(yellowflowers.png);
        background-position:left;
        background-attachment:fixed;
        background-repeat:repeat-y;
      }
  </style>
 </head>
 <body>
 </body>
</html>
```

Output:



BORDER IMAGES:

The CSS3 **border-image property** uses images to place a border around *any* block-level element.

- Stretching an Image Border

The border-image property has six values:

border-image-source—the URL of the image to use in the border (in this case, url(border.png)).

border-image-slice—expressed with four space-separated values in pixels (in this case, 80808080). These values are the *inward offsets* from the top, right, bottom and left sides of the image. Since our original image is square, we used the same value for each.

- The border-image-slice divides the image into nine *regions*: four corners, four sides and middle, which is transparent unless other- wise specified. These regions may overlap.
- If you use values that are larger than the actual image size, the border-image-slice values will be interpreted as 100%. *You may not use negative values*.
- We could express the border-image-slice in *two* values—8080—in which case the first value would represent the top and bottom, and the second value the left and right.
- The border-image-slice may also be ex-pressed in percentages.

border-image-repeat—specifies how the regions of the border image are scaled and *tiled* (repeated). By indicating stretch just *once*, we create a border that will stretch the top, right, bottom and left regions to fit the area.

- You may specify *two* values for the border-image-repeat property.

Stretch ,repeat, the top and bottom regions of the image border would be *stretched*, and the right and left regions of the border would be repeated (i.e., *tiled*) to fit the area.

- Other possible values for the border-image-repeat property in- clude round and space.
- If you specify round, the regions are repeated using only whole tiles, and the border image is scaled to fit the area. If you specify space, the regions are repeated to fill the area using only whole tiles, and any excess space is distributed evenly around the tiles.

✓ **Repeating an Image Border**

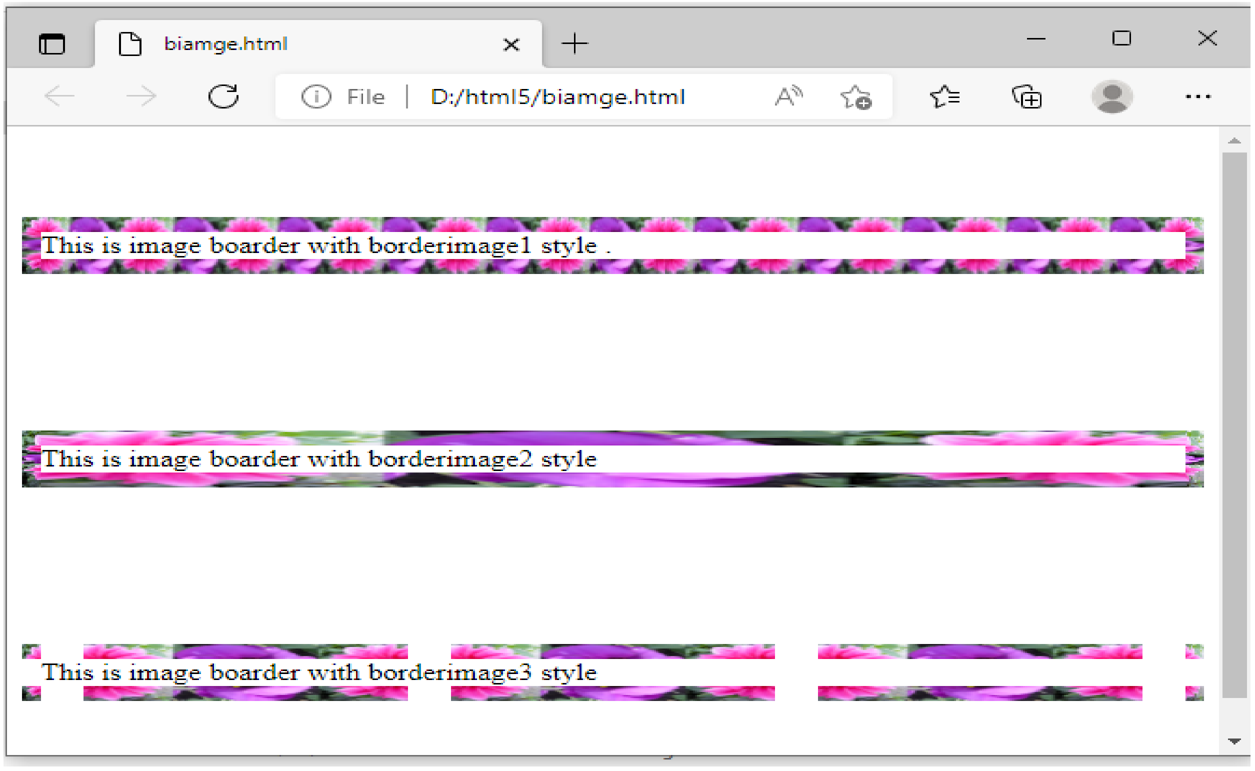
We create an image border by repeating the regions to fit the space. The border-image property

includes four values:
border-image-source—the URL of the image to use in the border (once again,url(border.png)).
border-image-slice—in this case, we provided *two* values expressed in percent-ages (34% 34%) for the top/bottom and left/right, respectively.
border-image-repeat—the value repeat specifies that the tiles are repeated to fit the area, using partial tiles to fill the excess space.

Example:

```
<html>
  <head>
    <style>
      #borderimg1 {
          border: 10px solid transparent;
        border-image-source: url(border.png);
        border-image-repeat: round;
        border-image-slice: 30;
        border-image-width: 10px;
      }
      #borderimg2 {
        border: 10px solid transparent;
        border-image-source: url(border.png);
        border-image-repeat: stretch;
        border-image-slice: 20;
        border-image-width:10px;
      }
      #borderimg3 {
        border: 10px solid transparent;
        border-image-source: url(border.png);
        border-image-repeat: space;
        border-image-slice: 30;
        border-image-width: 10px;
      }
    </style>
  </head>
  <body>
    <br> <p id = "borderimg1">This is image boarder with borderimage1 style  .</p> <br>
     <br><p id = "borderimg2">This is image boarder with borderimage2 style</p><br>
     <br><p id = "borderimg3">This is image boarder with borderimage3 style</p><br>
  </body>
</html>
```

Output:


biamge.html

File | D:/html5/biamge.html

This is image boarder with borderimage1 style .

This is image boarder with borderimage2 style

This is image boarder with borderimage3 style

COLORS:

CSS3 allows you to express color in several ways in addition to standard **color names** (suchas Aqua) or hexadecimal RGB values (such as #00FFFF for Aqua).

- ✓ **RGB** (Red, Green, Blue) or **RGBA** (Red, Green, Blue, Alpha) gives you greater control over the exact colors in yourweb pages.
    - The value for each color—red, green and blue—can range from 0 to 255. The *alpha* value—which represents *opacity*—can be any value in the range 0.0 (fully transparent) through 1.0(fully opaque). **For example**, if you were to set the backgroundcolor as follows:

background: rgba(**255**, **0**, **0**, **0.5**);

the resulting color would be a half-opaque red.

    - Using RGBA colors gives you far more op- tions than using only the existing HTML color names—there are over 140 HTML colornames, whereas there are 16,777,216 different RGB colors (256 x 256 x 256) and varyingopacities of each.
- ✓ CSS3 also allows you to express color using **HSL (hue, saturation, lightness)** or **HSLA (hue, saturation, lightness, alpha)** values.
    - The ***hue*** is a color or shade expressed asa value from 0 to 359 representing the degrees on a color wheel (a wheel is 360 degrees).
    - The colors on the wheel progress in the order of the colors of the rainbow—red, orange, yellow, green, blue, indigo and violet.
    - The value for red, which is at the beginning of the wheel, is 0. Green hues have values around 120 and blue hues have values around 240. A hue value of 359, which is just left of 0 on the wheel, would result in a red hue.
    - The ***satu ration***—the intensity of the hue—is expressed as a percentage, where 100%is fully saturated(the full color) and 0% is gray**.** ***Lightness***—the intensity of light or luminance of the hue—is also expressed as a percentage.

- A lightness of 50% is the actual hue. If you *decrease* the amount of light to 0%, the color appears completely dark (black). If you *increase* the amount of light to 100%, the color appears completely light (white).

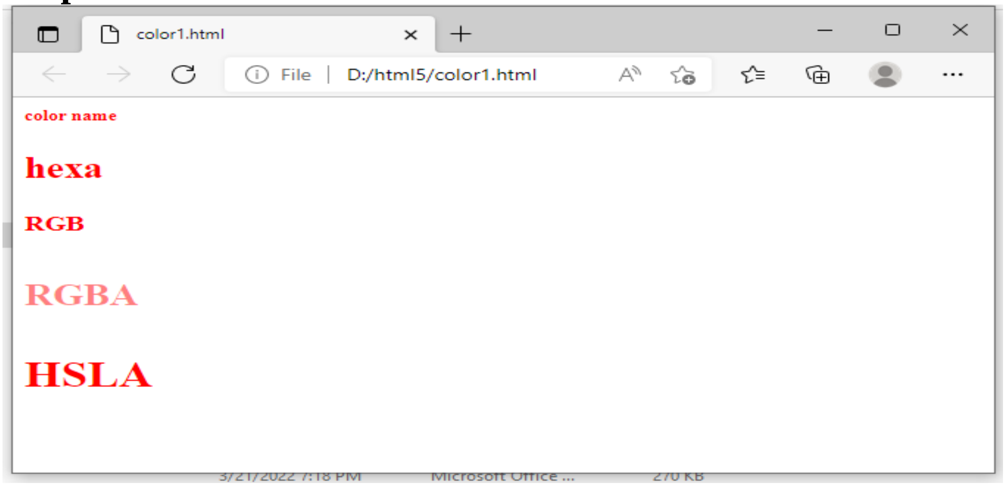
For example, if you wanted to use an hsla value to get the same color red as in our example of an rgba value, you would set the background property as follows:

background: hsla(**0**, **100%**, **50%**, **0.5**);

Example:

```
<!DOCTYPE html>
<html>
  <head>
   <style type="text/css">
     h5{color:red;}
     h2{color:FF0000;}
     h3{color:rgb(255,0,0);}
     h4{color:rgba(255,0,0,0.5);}
     h1{color:hsla(0,100%,50%,1.5);}
  </style>
  </head>
  <body>
     <h5> color name</h5>
     <h2 style="font-size:20pt"> hexa </h2>
     <h3>RGB </h3>
     <h4 style="font-size:20pt">RGBA </h4>
     <h1> HSLA</h1>
   </body>
</html>
```

Output:



color name

hexa

RGB

RGBA

HSLA

SHADOWS:

➢ **Text shadow:**

- The CSS3 **text-shadow property** makes it easy to add a **text shadow** effect to *any* text . First we add a text-shadow property to our styles . The property has four values: -4px, 4px, 6px and DimGrey, which represent:

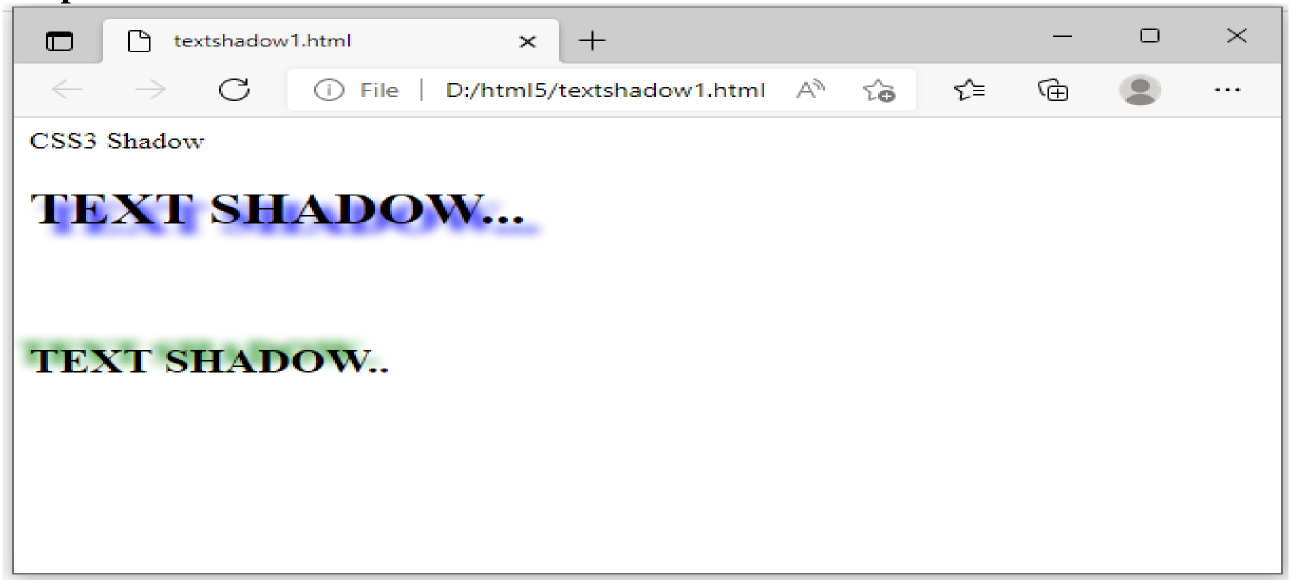
- **Horizontal offset of the shadow**—the number of pixels that the text-shadow will appear to the *left* or the *right* of the text. In this example, the horizontal offset of the shadow is -4px. A *negative* value moves the text-shadow to the *left*; a *positive* value moves it to the *right*.
- **Vertical offset of the shadow**—the number of pixels that the text-shadow will be shifted *up* or *down* from the text. In this example, the vertical offset of the shadow is 4px. A *negative* value moves the shadow *up*, whereas a *positive* value moves it *down*.
- **blur radius**—the blur (in pixels) of the shadow. A blur-radius of 0px would result in a shadow with a sharp edge (no blur). The greater the value, the greater the blurring of the edges. We used a blur radius of 6px.
- **color**—determines the color of the text-shadow. We used dimgrey.

Example:

```
<!DOCTYPE html>
<html>
  <head>CSS3 Shadow
   <style type="text/css">
     h1
        {

          text-shadow:7px 7px 6px blue;
         }
      h2
        {
          text-shadow:-6px -6px 6px green;
         }
   </style>
 <head>
 <body>
   <h1> TEXT SHADOW...</h1> <br> <br>
   <h2>TEXT SHADOW..</h2>
  </body>
</html>
```

Output:



CSS3 Shadow

TEXT SHADOW...

TEXT SHADOW..

➢ **<u>Box shadow</u>:**

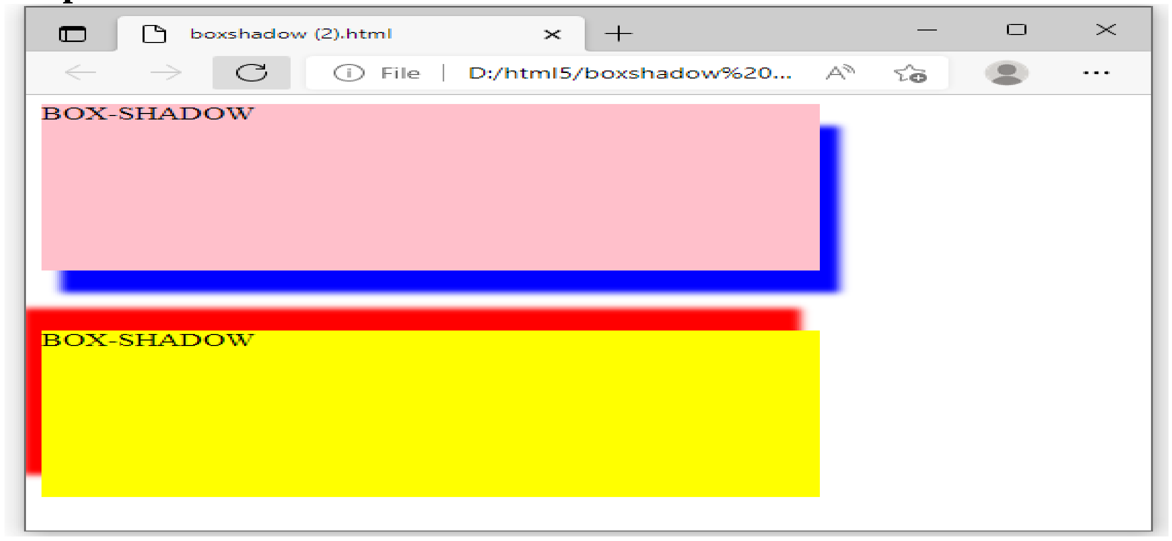
You can shadow *any* block-level element in CSS3. Next, we add the **box-shadow property** with four values :

- **Horizontal offset of the shadow (**25px)—the number of pixels that the box-shadowwill appear to the left or the right of the box. A *positive* value moves the box-shadow to the *right.A negative values moves the box-shadow to the left.*
- **Vertical offset of the shadow** (25px)—the number of pixels the box-shadowwill be shifted up or down from the box. A *positive* value moves the box-shadow*down. A negative values moves the box-shadow to the up.*
- **Blur radius**—A blur-radius of 0pxwould result in a shadow with a sharp edge (noblur). The greater the value, the more the edges of the shadow are blurred. Weused a blur radius of 10px.
- **Color**—the box-shadow's color .

Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
 #bs1
 {
 width:400px;
 height:150px;
 background-color:pink;
 box-shadow:10px 20px 5px blue;
   }
 #bs2
   {
    width:400px;
   height:150px;
   background-color:yellow;
   box-shadow:-10px -20px 5px red;
    }
</style>
</head>
 <body>
  <div id="bs1"> BOX-SHADOW</div><br><br><br>
  <div id="bs2">BOX-SHADOW</div>
 </body>
</html>
```

Output:



TRANSFORMATION:

It is a property by which the object can be rotated, scaled or skewed.

The 2D transformation:

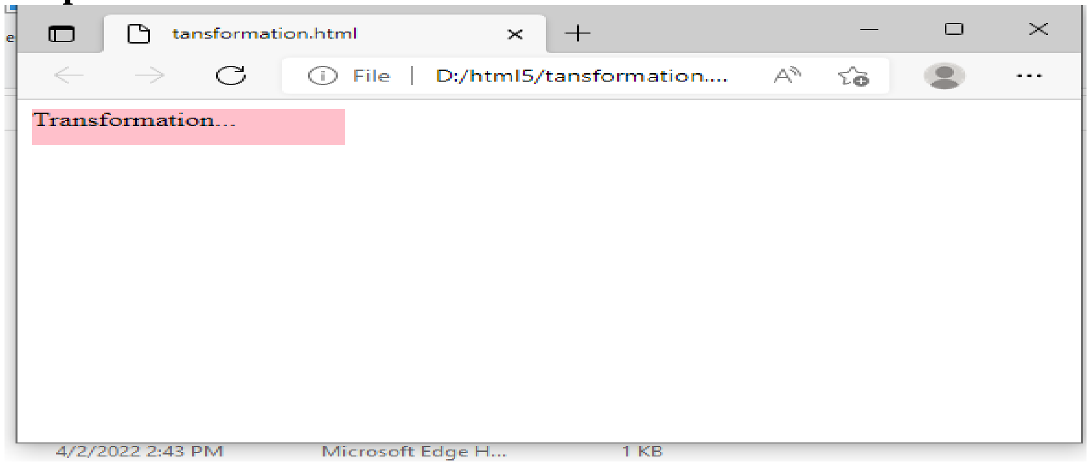
Translate Property:

- translate () : It moves an element from its current position.
  Syntax : transform: translate (30px,100px);
- rotate () : It is used to rotate the element in clockwise or anticlockwise manner to given degree.
  Syntax : transform : rotate(45deg);
- scale () : It used to increment or decrement the size of the element.
  scaleX () : Method increases or decreases the width of element.
  Syntax : scaleX(3);
  scaleY () : Method increases or decreases the height of element.
  Syntax : scaleY (4);
- skew () :Method skew the element along X and Y axis.
  Syntax : transform: skew (30deg,45deg);

Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
  div
  {
  width:170px;
  height:30px;
  background-color:pink;
      }
   div:hover
   {
    transform:translate(30px,100px);
   }
</style>
</head>
  <body>
   <div> Transformation...</div>
  </body>
</html>
```

Output:



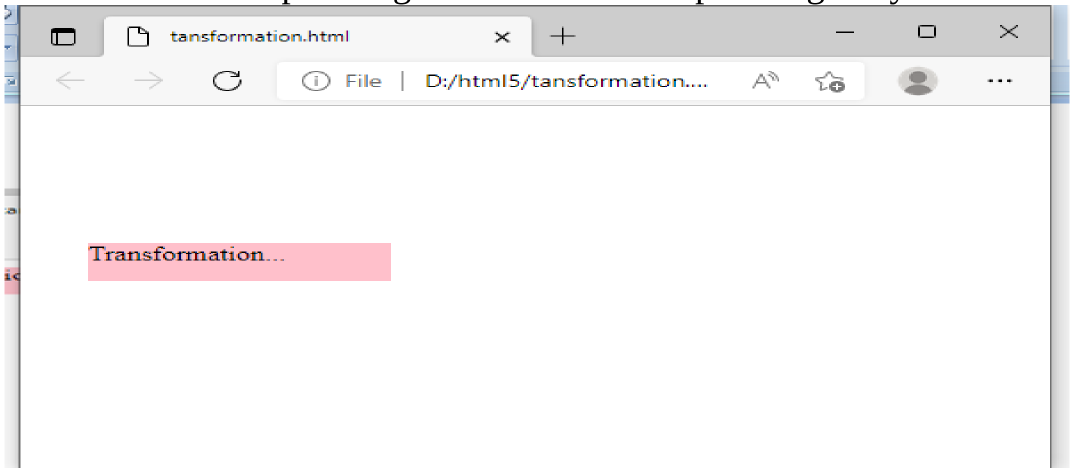
tansformation.html

File | D:/html5/tansformation....

Transformation...

4/2/2022 2:43 PM Microsoft Edge H... 1 KB

-→ div is hover 30px along the x-axis and 170px along the y-axis.



tansformation.html

File | D:/html5/tansformation....

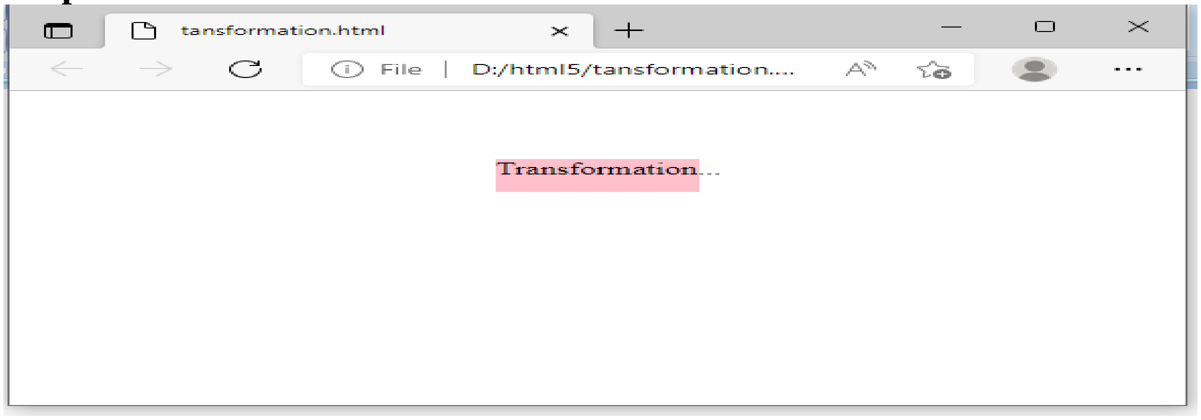
Transformation...

Example: scale()

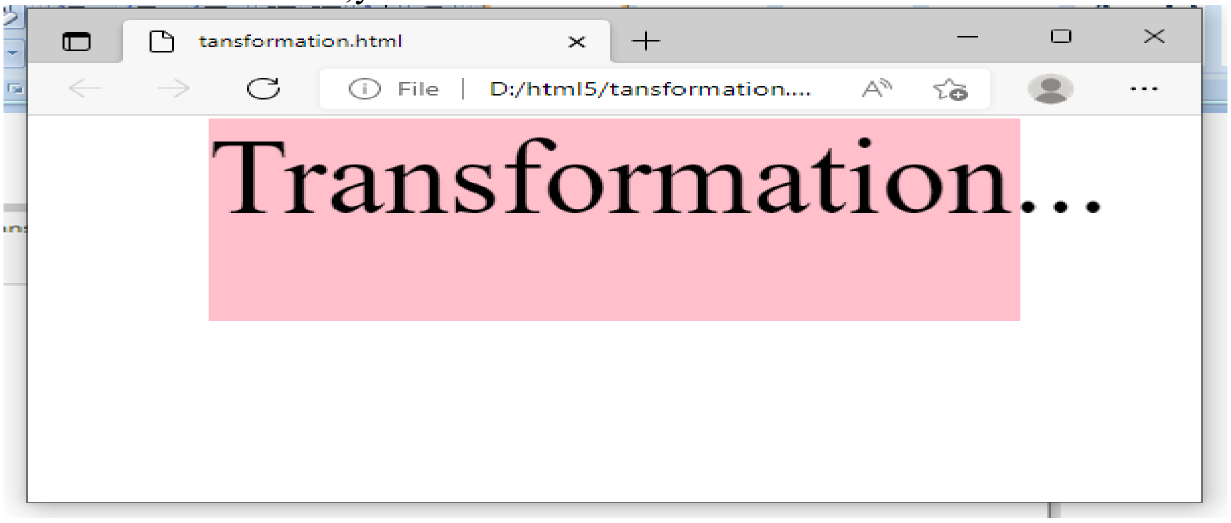
```
<!DOCTYPE html>
<html>
<head>
<style>
  div
  {
  width:100px;
  height:30px;
  background-color:pink;
      }
```

```
   div:hover
   {
    transform:scale(4,5); /*  scaled to x=4 and y=5 */
   }
</style>
</head>
  <body>
   <br> <br> <br><center>
    <div> Transformation...</div></center>
  </body>
</html>
```

Output:



→ div is scaled to x=4,y=5…



TRANSITIONS:

CSS Transitions is a module of CSS that lets you create gradual transitions between the values of specific CSS properties. The behavior of these transitions can be controlled by specifying their timing function, duration, and otherattributes.

Properties

- transition
- transition-delay
- transition-duration
- transition-property
- transition-timing-function

transition –timing-function → Specifies the speed curve of the transition effect.

- ease – Specifies the transition effect slow start, then fast end slowly.
- linear – same speed from start to end.
- ease-in – slow start.
- ease-out – slow end.
- ease-in-out –slow start and end.

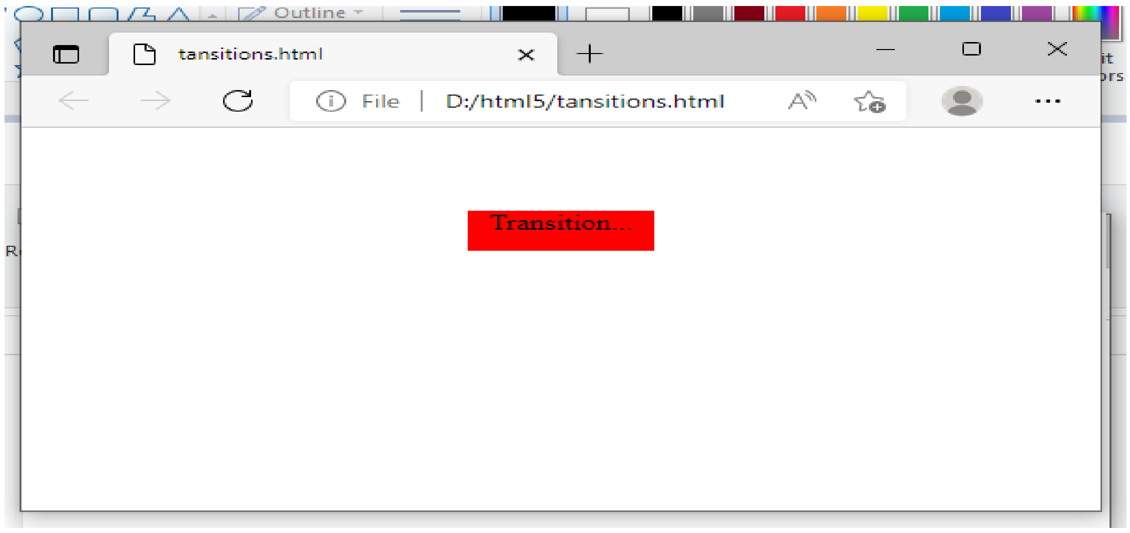
Example:

```
#div1
 { transition-timing-function: linear;}
#div2
{ transition-timing-function: ease;}
```

Example:

```
<! DOCTYPE html>
<html>
<head>
<style>
  div
  {
  width:100px;
  height:30px;
  background-color:red;
  transition:width 2s;
      }
   div:hover
   {
     width 300px;
   }
</style>
</head>
  <body>
   <br> <br> <br><center>
    <div> Transition...</div></center>
  </body>
</html>
```

OUTPUT:



ANIMATION:

CSS allows animation of HTML elements without using JavaScript or Flash!

What are CSS Animations?

- An animation lets an element gradually change from one style to another. You can change as many CSS properties you want, as many times you want.
- To use CSS animation, you must first specify some keyframes for the animation. Keyframes hold what styles the element will have at certain times.
- The @keyframes Rule

When you specify CSS styles inside the @keyframes rule, the animation will gradually change from the current style to the new style at certain times.

To get an animation to work, you must bind the animation to an element.

CSS Animation Properties

The following table lists the @keyframes rule and all the CSS animation properties:

Property	**Description**
@keyframes	Specifies the animation code
animation	A shorthand property for setting all the animation properties
animation-delay	Specifies a delay for the start of an animation
animation-direction	Specifies whether an animation should be played forwards,backwards or in alternate cycles
animation-duration	Specifies how long time an animation should take to completeone cycle
animation-fill-mode	Specifies a style for the element when the animation is notplaying (before it starts, after it ends, or both)
animation-iteration-count	Specifies the number of times an animation should be played
animation-name	Specifies the name of the @keyframes animation
animation-play-state	Specifies whether the animation is running or paused

animation-timing-function	Specifies the speed curve of the animation

Example:

```
/* The animation
code */@keyframes
example {
   from {background-
 color:red;} to {background-
 color: yellow;}
}

/* The element to apply the
animation to */div {
 width:
 100px;
 height:
 100px;
 background-color:
 red; animation-name:
 example;animation-
 duration: 4s;
}
```

Note: The animation-duration property defines how long time an animation should take to complete. If
the animation-duration property is not specified, no animation will occur, because the default value is 0s (0seconds).

In the example above we have specified when the style will change by using the keywords "from" and "to"(which represents 0% (start) and 100% (complete)).

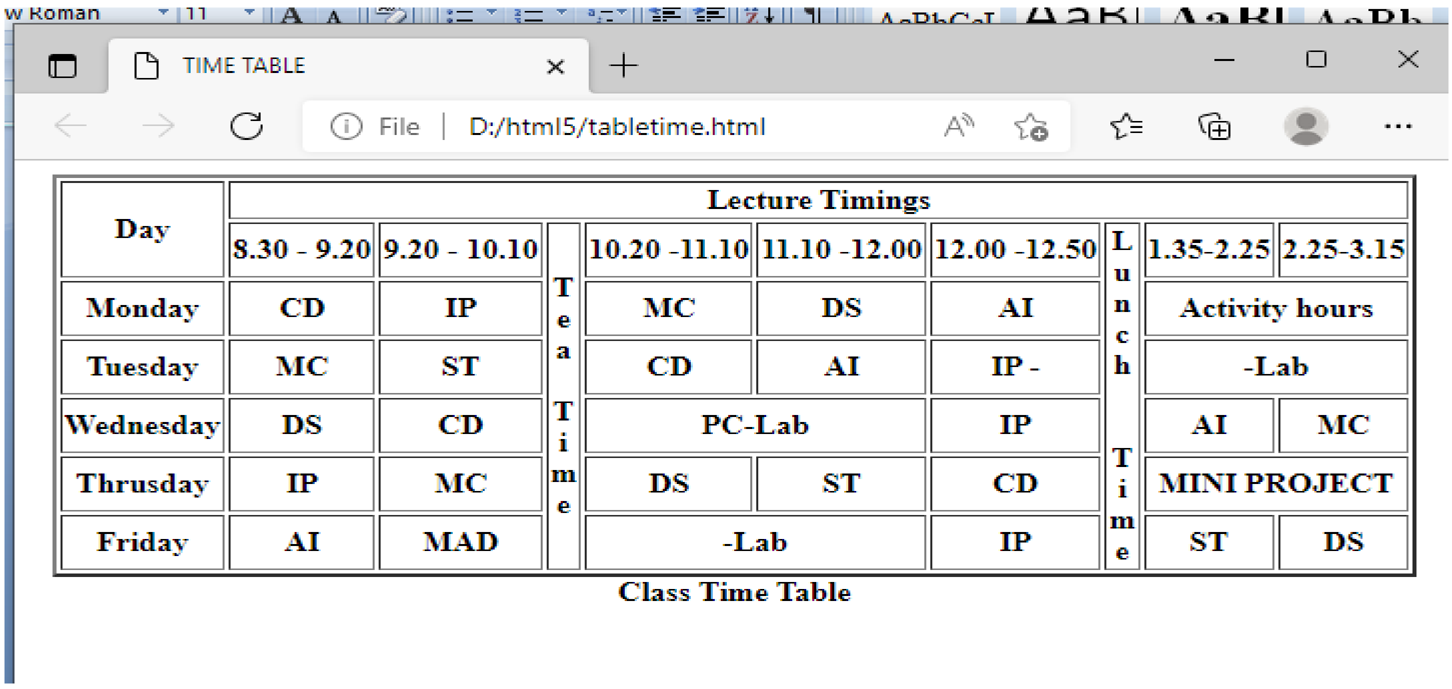
It is also possible to use percent. By using percent, you can add as many style changes as you like.

The following example will change the background-color of the <div> element when the animation is 25%complete, 50% complete, and again when the animation is 100% complete:

Example

```
/* The animation
code */@keyframes
example {
 0%  {background-color: red;}
 25%  {background-color: yellow;}
 50%  {background-color: blue;}
 100% {background-color: green;}
}
/* The element to apply the
animation to */div {
 width:
 100px;
 height:
 100px;
 background-color:
 red; animation-name:
 example;animation-
 duration: 4s;
}
```

Example: Create the following time table using HTML tags.



Day	Lecture Timings								
	8.30 - 9.20	9.20 - 10.10	Tea Time	10.20 -11.10	11.10 -12.00	12.00 -12.50	Lunch Time	1.35-2.25	2.25-3.15
Monday	CD	IP		MC	DS	AI		Activity hours	
Tuesday	MC	ST		CD	AI	IP -		-Lab	
Wednesday	DS	CD		PC-Lab		IP		AI	MC
Thrusday	IP	MC		DS	ST	CD		MINI PROJECT	
Friday	AI	MAD		-Lab		IP		ST	DS

Class Time Table

TIME TABLE:

```
<html>
 <head>
  <title> TIME TABLE </title>
 </head>
 <body>
  <table border="2" align=center>
   <caption align=bottom>
   <b>Class Time Table</b>
   </caption>

   <tr algin=center>
    <th rowspan=2>Day</th>
    <th colspan=9>Lecture Timings</th>
   </tr>
   <tr>
     <th> 8.30 - 9.20</th>
     <th> 9.20 - 10.10</th>
    <th rowspan=6>T<br>e <br>a<br> <br>T<br> i<br>m<br> e<br>
       </th>
     <th>10.20 -11.10</th>
     <th>11.10 -12.00</th>
     <th>12.00 -12.50</th>
<th rowspan=6>L<br> u<br>n<br>c<br>h<br><br><br> T<br>i<br>m<br>e<br></th>
     <th>1.35-2.25</th>
     <th>2.25-3.15</th>
       </tr>

   <tr align=center>
     <th>Monday </th>
     <th>CD</th>
     <th>IP </th>
     <th>MC </th>
     <th>DS</th>
     <th>AI</th>
<th colspan=2> Activity hours
   </tr>

   <tr align=center>
     <th>Tuesday </th>
     <th> MC</th>
     <th>ST </th>
     <th>CD</th>
     <th>AI</th>
     <th>IP -</th>
    <th colspan=2>-Lab </th>
   </tr>

   <tr align=center>
     <th>Wednesday </th>
     <th> DS</th>
     <th>CD </th>
      <th colspan=2>PC-Lab</th>
      <th>IP</th>
      <th>AI</th>
      <th>MC</th>
   </tr>

<tr align=center>
     <th>Thrusday </th>
     <th> IP</th>
     <th>MC </th>
     <th>DS</th>
     <th>ST</th>
     <th>CD</th>
     <th colspan=2>MINI PROJECT</th>
   </tr>

<tr align=center>
     <th>Friday </th>
     <th> AI</th>
     <th> MAD</th>
     <th colspan=2>-Lab </th>
     <th>IP</th>
     <th>ST</th>
     <th>DS</th>
   </tr>
 </body>
</html>
```